*République Algérienne Démocratique et Populaire*

*Ministère de l'Enseignement Supérieur*

*et de la Recherche Scientifique*

**Institut d'Aéronautique et des Etudes Spatiale**

**Département de Construction**

**Filière : Aéronautique**

**Option** : Avionique

**Mémoire de Fin d'Etudes**
**En vue de l'obtention du diplôme:**

# MASTER

Thème

# Contribution à la navigation sûre de systèmes robotiques autonomes

***Présenté par :***

***M^elle^ MAHFOUFI Fella Fadia***

***M^elle^ BENRAHMANI Hanane***

***Dirigé par:***

***M^me^ BOURAINE SARA : Encadreur***

***M^r^ BENAISSA RACHID: Promoteur***

**2019/2020**

# *Résumé*

Ce travail traite le problème de la navigation autonome pour un robot mobile de type différentiel avec un champ de vision limité dans un environnement dynamique inconnu. La garantie de la sureté de mouvement est de maintenir le robot loin des états de collision et aussi le garder loin des états qui le conduisent à une collision. C'est pourquoi, nous proposons un algorithme de navigation réactive, raisonnant sur un pas de temps qui garantit la sûreté passive car il est impossible d'assurer la sûreté absolue en respectant les conditions du monde réel. Cette méthode est un niveau plus élevé que l'évitement d'obstacles qui laisse le robot se déplacer loin des états de collision inévitables dans un environnement dynamique et inconnu en utilisant les capteurs embarqués, et en prenant en compte non seulement les objets statiques et dynamiques mais aussi les obstacles inattendus et non-perçus ainsi que leur comportement futur.



***Mots clés* :** Robot mobile – Environnement dynamique – Navigation autonome – Sûreté de mouvement – Evitement d'obstacles – obstacle inattendu – Etats de collision inévitable.

# *Abstract*

This work addresses the problem of autonomous navigation for a differential type mobile robot with a limited field of view in an unknown dynamic environment. The guarantee of the motion safety is to keep the robot away from the collision states and also keep it away from the states which lead to a collision. This is why we propose a reactive navigation algorithm, reasoning on a time step that guarantees passive safety because it is impossible to ensure absolute safety by respecting the criterias applied in the real world. This method is a higher level than obstacle avoidance which lets the robot move away from inevitable collision states in dynamic and unknown environment using on-board sensors, taking into account not only static and dynamic objects but also unexpected and unperceived obstacles also their future behavior.



**Keywords** Mobile robots – Dynamic environments – Autonomous navigation – Motion safety– Collision avoidance– unexpected obstacles – Inevitable collision state

# ملخص

يعالج هذا العمل مشكلة الملاحة المستقلة لروبوت متحرك من النوع التفاضلي مجال رؤيته محدود في بيئة ديناميكية غير معروفة. لضمان سلامة الحركة يجب إبقاء الروبوت بعيدًا عن حالات الاصطدام وأيضًا إبعاده عن الحالات التي تؤدي إليها. مما دفعنا لتطوير خوارزمية تضمن السلامة الكامنة للحركة لأنه لا يمكن ضمان السلامة المطلقة . هذه الطريقة هي مستوى متقدم من تجنب العوائق التي تسمح للروبوت بالابتعاد عن حالات الاصطدام الحتمية في بيئة ديناميكية وغير مألوفة باستخدام أجهزة استشعار مع مراعاة الأشياء الثابتة والديناميكية وأيضًا العقبات الغير متوقعة والغير مرئية .وبالتالي، بغض النظر عما يحدث في البيئة في وقت التنفيذ ، فلن يحدث التصادم أبدًا مع عقبة غير متوقعة وإذا كان الأمر كذلك ،سيكون الروبوت متوقفا

**الكلمات الرئيسية :**الروبوتات المتحركة -البيئات الديناميكية الملاحة المستقلة - سلامة الحركة مواضع الاصطدام المحتوم-العقبات الغير متوقعة - تجنب الاصطدام

## *REMERCIMENT*

Nous adressons toute notre gratitude en premier lieu à notre Seigneur «ALLAH» pour la volonté, la santé, et la force qu'Il nous a procuré pour la réalisation de ce projet. En second lieu, nous tenons à exprimer notre respect et notre reconnaissance à notre Encadreur madame : SARA BOURAINE qui a toujours été disponible malgré ses nombreuses occupations, et dont les encouragements et les conseils nous furent d'une très grande utilité. Nous la remercions encore une fois pour le fait qu'elle nous à fait aimer ce domaine. Nous ne saurons trop lui témoigner notre gratitude. Sans oublié d'adresser nos sincères remerciements au directeur de CDTA d'avoir nous accepté comme stagiaires, ainsi que monsieur DJEKOUN OUALID et monsieur SEDDIKI FAYSEL. Nous tenons également à remercier notre Promoteur Monsieur BENAISSA RACHID de nous avoir consacré une partie de son temps et nous a pousser à choisir ce sujet. Nous remercions aussi monsieur ZABOT AMAR pour sa présence et son aide, surtout pour le fait qu'il a cru à nos compétences dés le début. Nous rendons grâce à la directrice de l'IAES pour son écoute et sa compréhension. Nous remercions vivement les membres du jury pour avoir accepté d'évaluer ce modeste mémoire. Enfin, nous tenons à exprimer nos vifs et sincères gratitudes à toute personne ayant contribué matériellement ou moralement, de prés ou de loin, à la confection de ce présent mémoire.

# *Dédicace*

*Je dédie ce mémoire à :*

*À ma mère, la plus belle créature que Dieu m'a donné*

*À cette source de tendresse, de patience et de générosité*

*À celle qui a toujours sacrifié pour mon bonheur et mon succès*

*À mes frères KHALED, HAMZA, WALID ET AKRAM qui ont été toujours à mes cotés, qui me favorisent toujours et cherchent mon bonheur*

*À la mémoire de ma grand- mère qui a toujours attendu le jour de ma soutenance*

*À mon oncle EL HACHMI pour ses prières et sa présence ainsi que tous mes oncles et mes tantes, surtout ma tante KHIRA.*

*A ma deuxième mère qui a été toujours la à mon écoute MAMA YOUSFEN RABIA et à mon professeur que je respecte et j'aime beaucoup madame CHAHBOUNIA MOUNIRA. Ces deux merveilleuses femmes qui m'ont toujours soutenir et m'ont donné de la force pour donner à fond et ne jamais perdre l'espoir*

*A la mémoire de mon papa RAHMANI, celui qui m'a toujours supporté et qui a toujours rêvé de vivre ce jour*

*À mes sœurs ABLA, SARRA et ROMAISSA pour leurs soutient et aide*

*Surtout, à ma sœur et binôme FELLA que j'aime et sa famille qui m'ont hébergé et m'ont traité comme une membre de la famille*

*À mes amis MAWIA, IMANE et REDHA qui n'ont jamais refusé de m'aider.*

*Je remercie celui qui a été mon professeur d'avionique, même si de loin, il m'explique ce que je ne comprends pas et me cherche de l'aide si besoin monsieur AHMED.*

*À tous ceux et celles qui me sont chers*

*Et à tous ceux qui, même par un mot, m'ont donné la force de continuer et arriver à ce jour*

*..... Hanane*

# *Dédicace*

*Je dédie ce mémoire ...*

*À MES CHERS PARENTS, Aucune dédicace ne saurait exprimer mon respect, mon amour éternel et ma considération pour les sacrifices que vous avez consenti pour mon instruction et mon bien être. Je vous remercie pour tout le soutien et l'amour que vous me portez depuis mon enfance, pour vos conseils et encouragements qui m'ont toujours poussé d'avancer et donner à fond et ils m'ont fait ce que je suis aujourd'hui. Vous êtes ma source de force.*

*J'espère que votre bénédiction m'accompagne toujours. Que ce modeste travail soit l'exaucement de vos vœux tant formulés, le fruit de vos innombrables sacrifices, bien que je ne vous en acquitterai jamais assez.*

*A Ma chère sœur CHANEZ que j'aime beaucoup. Merci d'être avec moi dans les moments difficiles d'avoir m'aider et me supporter, tu étais toujours mon meilleur cadeau que dieu m'a offrir.*

*A ma famille, mes proches et à ceux qui me donnent de l'amour et de la vivacité*

*A mes grands-parents, pour leurs prières et leur soutien tout au long de mes études.*

*A mes tantes HASSINA, SAMIRA, RADIA, FAIROUZ, FOUZIA, NASSIMA et mes oncles REDHA, KARIM qui m'ont toujours encouragé et soutenu.*

*A mes cousines FERIEL, LYNA ainsi que tous mes cousins.*

*A mes chers copines HANANE et FETTOUMA. Je les remercie d'être à l'écoute, elles m'ont chaleureusement supporté, conseillé et encouragé à tout moment.*

*A tous les gens qui m'ont aidé de prés ou de loin dans ce travail.*

*A tous ceux que j'aime. Merci d'être toujours là pour moi*

*Puisse dieu, vous donne santé, bonheur, courage, et surtout réussite.*

*.....FELLA*

# Sommaire

# Liste des figures

# Liste des acronymes

**ASV:** autonomous surface vessel.

**CBT :** espace temps de configuration.

**CC :** collision cône.

**CDTA:** centre de développement des techniques avancées.

**CIR:** Centre instantané de rotation.

**$C_{libre}$ :** espace libre de configuration.

**CPA:** closest point approach.

**CTRV**: constant turning rate velocity.

**CV:** constant velocity.

**DDMR:** differential drive mobile robot.

**DRVO**: dynamic reciprocal velocity obstacles.

**DW:** dynamic window.

**Espace-C ou C:** espace de configuration.

**FNN**: Fuzzy neural networks.

**FoV:** field of view.

**GPS:** global positionnig system.

**HOG:** Histograms of Oriented Gradient.

**ICO:** inevitable collision obstacles.

**ICS :** inevitable collision states.

**$ICS^b$:** braking inevitable collision states.

**$ICS^d$:** distance based inevitable collision states.

**IMO:** international maritime organization.

**IVO:** Inverse Velocity Obstacles.

**MAV :** micro UAV.

**MC:** cellule mobile dans la grille d'occupation.

**NCRM :** navigation et control des robots mobiles autonomes.

**Obstacle-C ou CB:** la région des obstacles de configuration.

**ORCA:** optimal Reciprocal Collision avoidance.

**OS:** operating system.

**PC:** personal computer.

**PDG :** premier directeur général.

**PFS:** passively friendly safe.

**PS:** passivement sûr.

**RAV:** reachable avoidance velocity.

**ROS:** robot operating system.

**RV:** reachable velocity.

**S:** espace d'état.

**SLAM:** Simultaneous Localization and Mapping.

**ST:** espace d'état-temps.

**SVO:** security velocity obstacles.

**TBO:** trajectory based operation.

**TCAS:** traffic_alert collision avoidance system.

**UGV:** unmanned ground vehicle.

**UAV:** unmaned aerial vehicles.

**VK:** viability kernel.

**VO** : velocity obstacles.

# *Introduction générale*

De nos jours, l'une des technologies à la croissance la plus rapide est la technologie robotique. Avec son développement, nous pouvons trouver des solutions aux problèmes dont nous ne pouvions rêver dans le passé. Les robots font partie de notre quotidien, ils sont de plus en plus performants et autonomes. L'utilisation des robots peut encourager et simplifier le travail d'ingénierie et peut aider partout dans le monde.

Les robots sont utilisés dans différents domaines comme l'industrie manufacturière, le secteur médical (chirurgie de précision, robots infirmiers), le secteur militaire (sauvetage, surveillance, robots démineurs, drones), l'exploration spatiale et sous marine, le transport, etc.

Les robots de service comme leur nom l'indique sont généralement destinés à aider l'être humain ; assister les handicapés (comme la chaise roulante intelligente qui peut être utilisée aussi bien en environnement d'intérieur ou d'extérieur). Pour les personnes âgées et les malades, ces robots sont d'une grande aide, ils sont aptes à se déplacer dans des environnements encombrés d'obstacles mobiles. Dans ces conditions, il est primordial de garantir qu'un robot tel que la chaise roulante ne cause pas de dégâts ou qu'il rentre en collision avec une personne ou un véhicule. Ces robots utilisent des fonctionnalités d'intelligence artificielle qui adapte en conséquence leurs mouvements.

Un robot a toujours une tâche qui consiste à exécuter un mouvement entre le début et la position souhaitée. Différents types d'obstacles peuvent se produire dans l'espace de travail du robot, on peut parler d'environnement dynamique, s'il y a non seulement des obstacles statiques, mais aussi mobiles. Le robot doit naviguer et atteindre la position cible sans heurter aucun des obstacles. Il est également important de savoir à quelle vitesse et de quelle manière la tâche peut être accomplie.

La navigation autonome a attiré beaucoup d'attention ces dernières années, elle est applicable dans plusieurs domaines tels que les voitures autonomes, les opérations de sauvetage, etc. Toutes ces applications nécessitent un système d'évitement de collision pour une navigation sûre du système vers le but.

Depuis les débuts de la robotique mobile, la capacité de se déplacer tout en évitant les collisions est une préoccupation majeure. La navigation (planification des mouvements et évitement d'obstacles) est passée de modèles simples de l'environnement (statiques et connus) à des modèles complexes et réalistes (dynamiques).De nombreuses approches couvrant toute la gamme peuvent être trouvées dans la littérature. Les techniques de planification de mouvement sont bien adaptées aux environnements contrôlés et

immuables. Les méthodes réactives sont des approches d'évitement d'obstacles et elles sont populaires en raison de leur réactivité aux objets imprévus dans des environnements statiques et dynamiques.

Bien que la majorité des approches réactives puissent être appliquées en environnement dynamique, il ne peut pas être garanti qu'aucune collision n'aura jamais lieu. Les roboticiens ont essayé de concevoir un système de navigation sûr. Cependant, la notion de la sûreté de mouvement a été souvent mal définie.

En sureté, le problème est de calculer des mouvements pour lesquels il est garanti que, quoi qu'il se passe au moment de l'exécution, le système robotique ne se trouve jamais dans une situation où il n'ya aucun moyen d'éviter la collision avec un obstacle inattendu donc c'est un niveau plus élevé que l'évitement d'obstacles. Une navigation sûre dans des environnements dynamiques nécessite des critères. Le non-respect de ces derniers donne une sûreté de mouvements insuffisante.

Plusieurs travaux ont été proposés pour résoudre le problème de sureté de mouvement comme le concept ICS. Un ICS pour un système robotique est un état pour lequel une collision se produit finalement quelle que soit la trajectoire future du système, cet état représente du point de vue sureté un état qui doit être évité par le robot. En conséquence, ICS peut être utilisé pour garantir la sécurité des mouvements. Les ICS sont définis avec une perspective de sûreté de mouvement absolue. La sureté absolue nécessite une connaissance complète du futur jusqu'à l'infinie ce qui est impossible dans le cas réel, c'est pourquoi la sureté passive est apparue. Elle garantit que si une collision aura lieu, le robot sera à l'arrêt.

L'objectif principal de ce mémoire est de développer un système de navigation d'un robot mobile permettant la navigation autonome en toute sécurité un environnement inconnu avec un champ de vision limité en tenant compte les objets perçus et non-perçus, statiques et dynamiques, aussi les contraintes cinématiques et dynamiques de robot. Pour ce faire, un algorithme est développé pour mener le robot d'un passivement état sûr à un autre état qui doit être passivement sûr aussi par le calcul des ICS (régions interdites) et les éviter. Ainsi, la sûreté passive est garantie.

Ce travail est réalisé au sein du Centre de Développement des Techniques Avancées (CDTA), et plus particulièrement dans la division productique et robotique au sein de l'équipe NCRM. Où notre algorithme est destiné à être utilisé sur une chaise roulante robotisée qui fait partie des types de robots différentiels.

Ce travail est organisé en 4 chapitres comme suit. Le chapitre 1 définit la robotique et les différents types d'un robot mobile, une description du robot utilisé dans ce travail est présentée ainsi que ses propriétés en donnant son modèle et ses capteurs embarqués. Dans le chapitre2, un état de l'art sur les méthodes de navigation réactives est présenté montrant ceux qui garantissent la sûreté et ceux qui ne la garantissent pas en se basant sur des

critères. Le chapitre3 traite la méthode choisie qui est le cœur de notre travail et un algorithme de vérification des états de collision inévitables vérifie si un état est un ICS ou non. Un algorithme général de la navigation sûre est présenté. Dans le dernier chapitre l'implémentation de l'approche développée est montrée en expliquant la méthode de travail et les résultats discutés ainsi que l'outil informatique utilisé qui est « ROS ».

Nous terminons notre travail par une conclusion générale récapitulant ce qui a été fait et exposant les perspectives de notre travail.

# CHAPITRE 1

# Les robots mobiles

## 1-1 Introduction :

Les robots mobiles présentent aujourd'hui un formidable potentiel technologique, ils deviennent indispensables dans notre vie quotidienne.ils peuvent remplacer, assister l'être humain et le servir.

Ce chapitre a d'abord pour objectif de définir la robotique et présenter les différents types de robots mobiles à roues à savoir les plus utilisés en robotique mobile. Comme cas particulier, une étude cinématique de notre plateforme le robot mobile utilisé dans ce travail qui est la chaise roulante robotisée de CDTA ainsi que ses caractéristiques.

Ensuite, Comme cas particulier, une étude cinématique du robot mobile utilisé dans ce travail qui est la chaise roulante robotisée de CDTA ainsi que ses caractéristiques sont présentées.

Enfin, les capteurs embarqués sont définis dans ce chapitre en montrant leurs principes de fonctionnement et ses équations de calcule.

## 1.2 Définition de la robotique :

La robotique est un domaine de recherche interdisciplinaire à l'interface de l'informatique et de l'ingénierie. Elle implique la conception, la construction, l'exploitation et l'utilisation des robots. Son objectif est de concevoir des machines intelligentes qui peuvent aider et assister les humains dans leur vie quotidienne et assurer la sécurité de chacun. Ce domaine s'appuie sur les recherches de l'ingénierie informatique, de l'ingénierie mécanique, de l'ingénierie électronique et autres.

La robotique développe des machines qui peuvent se substituer aux humains et reproduire les actions humaines. Les robots peuvent être utilisés dans de nombreuses situations et à de nombreuses fins, mais aujourd'hui, beaucoup sont utilisés dans des environnements dangereux (y compris l'inspection des matières radioactives, la détection et la désactivation de bombes), les processus de fabrication ou lorsque les humains ne peuvent pas survivre (par exemple dans l'espace, sous l'eau, en chaleur, nettoyage et confinement des matières dangereuses et des rayonnements). Les systèmes robotisés peuvent prendre n'importe quelle forme, mais certains sont faits pour ressembler à des humains en apparence. On dit que cela aide à les accepter dans certains comportements réplicatifs généralement pratiqués par des personnes. Ces robots tentent de reproduire la marche, la

parole, la cognition ou toute autre activité humaine. Beaucoup d'entre eux d'aujourd'hui sont inspirés par la nature, contribuant au domaine de la robotique bio-inspirée.

## 1-3 Définition d'un robot mobile :

Un robot mobile est un système mécanique, électronique et informatique qui peut effectuer des actions physiques sur son environnement pour atteindre les objectifs qui lui sont assignés. Cette machine a une large gamme d'utilisations et peut s'adapter à certaines modifications de ses conditions de travail. Il a les fonctions de perception, de prise de décision et d'action. Par conséquent, même lorsqu'il rencontre de nouvelles situations inattendues, le robot doit être en mesure d'exécuter diverses tâches de différentes manières et d'exécuter ses tâches correctement.

L'appellation Robot mobile regroupe tous les types de robots qui ont la capacité de déplacement qui est la caractéristique commune entre eux, la différence réside dans la manière, qui dépend du domaine d'utilisation de robot, par laquelle le robot va atteindre cette faculté de mouvement. La mobilité par les roues est la structure mécanique la plus communément appliquée. Cette technique assure selon l'agencement et les dimensions des roues un déplacement dans toutes les directions avec une accélération et une vitesse importante.

Dans ce mémoire, on s'intéresse aux robots mobiles à roues.

## 1-4 Disposition des roues et centre de rotation instantané :

La combinaison de la sélection des roues et de leurs dispositions donne aux robots la possibilité de se déplacer à sa propre façon. Sur les robots mobiles, on rencontre principalement quatre types de roues [1, 2]:

– les roues fixes dont l'axe de rotation, de direction constante, passe par le centre de la roue, et l'axe d'orientation est constant (figure 1.1 (a)) ;
– les roues centrées orientables, dont l'axe d'orientation passe par le centre de la roue (figure 1.1 (b));

– les roues décentrées orientables, souvent appelées roues folles, pour lesquelles l'axe d'orientation ne passe pas par le centre de la roue (figure 1.1 (c));

– les roues suédoises dont la bande de roulement a été remplacées par des galets inclinés par rapport à la normale au plan de la roue. C'est la combinaison de la rotation de la roue avec la rotation libre du galet en contact avec le sol qui permet un déplacement sans glissement dans toutes les directions (figure 1.1 (d)).

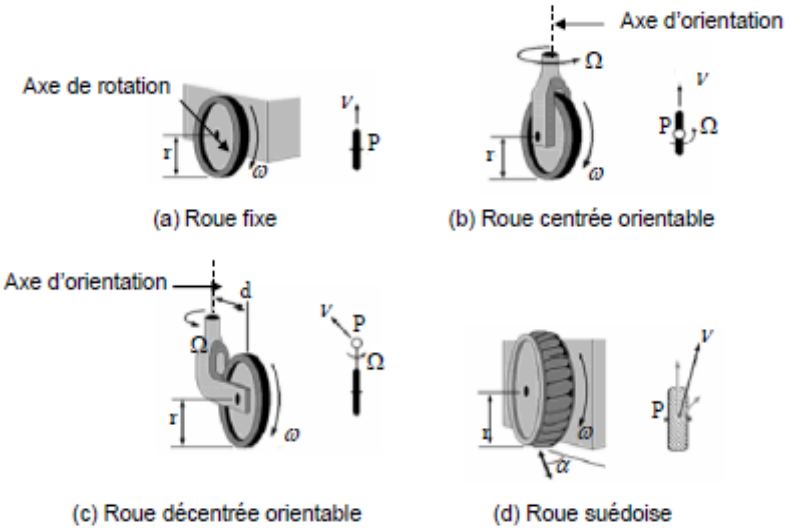



**Figure 2.1 : Les principaux types de roues pour robots mobiles**

On rencontrera aussi des systèmes particuliers, tels que les roues à plusieurs directions de roulement, etc.

Bien évidemment, pour un ensemble de roues donné, toute disposition ne conduit pas à une solution viable. Un mauvais choix peut limiter la mobilité du robot ou causer un blocage. Par exemple, un robot équipé de deux roues fixes non parallèles ne pourrait pas aller en ligne droite ! Pour qu'une disposition de roues soit viable et n'entraine pas de glissement des roues sur le sol, il faut qu'il existe pour toutes ces roues un unique point de vitesse nulle autour duquel tourne le robot de façon instantanée. Ce point, lorsqu'il existe, est appelé centre instantané de rotation (CIR).

Les points de vitesse nulle liés aux roues se trouvant sur leur axe de rotation, il est donc nécessaire que le point d'intersection des axes de rotation des différentes roues soit unique. Pour cette raison, il existe en pratique plusieurs catégories de robots mobiles à roues.

## 1-5 Les grandes classes de robots mobiles [1, 2, 3]. :

### 1-5-1 Robots mobiles de type différentiel :

On désigne par différentiel un robot actionne par deux roues fixes indépendantes et possédant éventuellement un certain nombre de roues folles (généralement deux) assurant sa stabilité. Le schéma des robots de type différentiel est donné à la figure 1.2. On y a omis les roues folles, qui n'interviennent pas dans la cinématique.

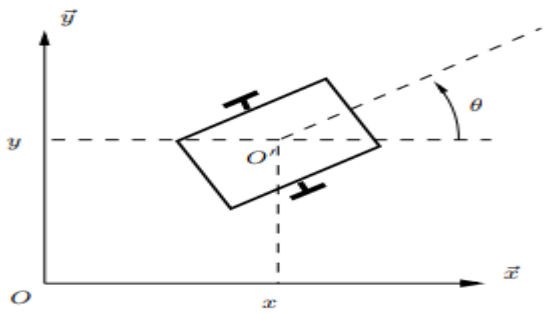



**Figure 1.2 : Robot mobile de type différentiel**

Ce type de robot est très répandu en raison de sa simplicité de construction et de propriétés cinématiques. La figure 1.3 présente différents robots de type différentiel


**Figure 1.3 : Exemple de robots mobiles différentiels : (a) Robot mobile Pioneer P3-DX, (b) Robot mobile ATRV2 du CDTA.**

## 1-5-2 Robots mobiles de type tricycle :

Un robot de type tricycle est constitué de deux roues fixes placées sur un même axe et d'une roue centrée orientable placée sur l'axe longitudinal. Le mouvement du robot est donné par la vitesse des deux roues fixes et par l'orientation de la roue orientable. Son centre de rotation est situé à l'intersection de l'axe contenant les roues fixes et de l'axe de la roue orientable.

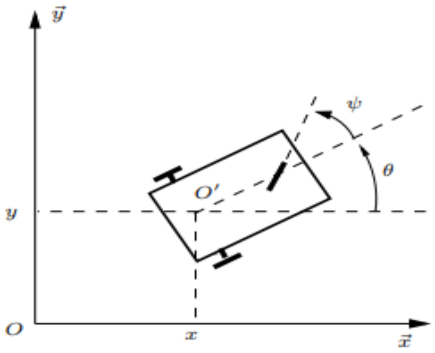



**Figure 1.4 : Robot mobile de type tricycle**

Il est impossible de le déplacer dans une direction perpendiculaire aux roues fixes. Sa commande est plus compliquée. Il est en général impossible d'effectuer des rotations simples à cause d'un rayon de braquage limité de la roue orientable.

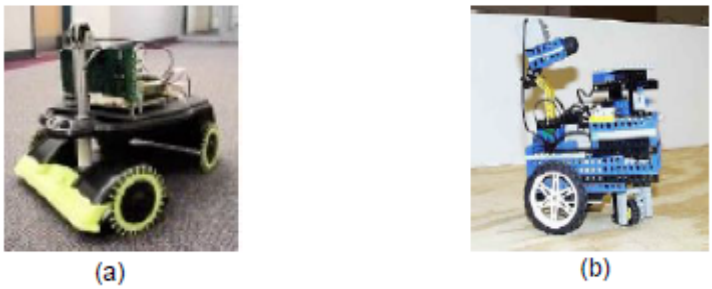



**Figure 1.5 : Exemple de robots mobiles de type tricycle**

## 1-5-3-Robots mobiles de type voiture :

Un robot de type voiture est semblable au tricycle, il est constitué de deux roues fixes placées sur un même axe et de deux roues centrées orientables placées elles aussi sur un même axe (figure 1.6).

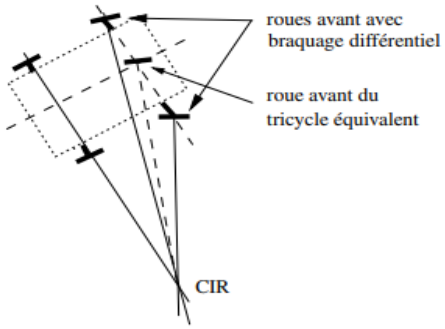



**Figure 1.6 : Un robot mobile de type voiture et son CIR**

Le robot de type voiture est cependant plus stable puisqu'il possède un point d'appui supplémentaire.

Toutes les autres propriétés du robot voiture sont identiques au robot tricycle, le deuxièmes pouvant être ramené au premier en remplaçant les deux roues avant par une seule placée au centre de l'axe, et ceci de manière à laisser le centre de rotation inchangé.

(a) Robot Kanade

(b) Robot Cycab

**Figure 1.7 : Exemple de robots mobiles de type voiture**

## 1-5-4 Robots mobiles de type omnidirectionnel :

Un robot omnidirectionnel est un robot qui peut se déplacer librement dans toutes les directions. Il est en général constitué de trois roues décentrées orientables ou de trois roues suédoises placées en triangle équilatéral.

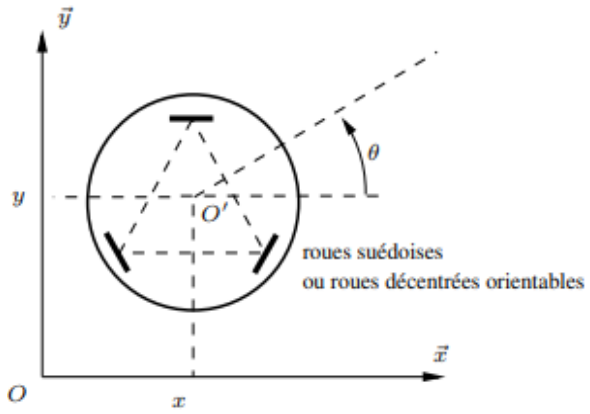



**Figure 1.8 : Représentation d'un robot mobile omnidirectionnel**

**Figure 1.9 : Robot mobile omnidirectionnel Nomadic XR4000**

## 1-6 Description du robot utilisé:

**Figure 1.10 : Photo de la chaise roulante FAURSA développé au CDTA**

La plateforme expérimentale utilisée dans ce travail est un fauteuil roulant robotique appelé FAURSA, développé au CDTA. La chaise roulante robotisée est du type différentiel DDMR. Ce type est devenu l'un des plus utilisés en raison de sa structure simple et de ses

caractéristiques de mouvement intéressantes (telles que sa capacité à tourner sur lui même).

Elle est équipée de deux roues de moteur sans balais, d'un télémètre laser, de capteurs à ultrasons, d'encodeurs (odomètres), d'un système embarqué basé sur des contrôleurs Arduino pour les implémentations de bas niveau et d'un ordinateur portable pour les implémentations de haut niveau. Le Pc utilisé est un HP EliteBook doté d'un processeur Intel i5-2540M CPU 2.6 GHz, RAM 4 Go.

### 1-6-1 Les propriétés de la chaise roulante :

- Longueur: 750 mm
- Largeur: 720 mm
- Hauteur: 940 mm
- Poids du nouveau châssis: 20kg
- Poids total de la chaise: 35 kg
- Charge: 100 kg


**Figure 1.11 : Les dimensions de la chaise FAURSA**

### 1-6-2 Description des capteurs embarqués [1] :

#### 1-6-2-1 Odomètre :

La question de la trajectométrie en robotique est très importante. Elle correspond à déterminer à chaque instant : la position, la vitesse et l'accélération de l'engin, mais la question qui se pose, comment estimer sa position à partir de la mesure du trajet qu'il a accompli depuis une position initiale ? La réponse est l'utilisation de l'odométrie.

L'odométrie est une technique permettant d'estimer la position d'un véhicule en mouvement. Cette mesure est présente sur quasiment tous les robots mobiles, grâce à des capteurs embarqués permettant de mesurer le déplacement du robot (de ses roues).

Les systèmes odométriques peuvent fournir l'information concernant le changement de la position du robot mobile, ces informations sont extraites au moyen de capteurs qui comptent le nombre de rotations pour les axes des roues et leurs axes d'orientation, pour cela des codeurs à haute résolution sont utilisés. Dans la plupart des cas ce sont des codeurs optiques incrémentaux, il existe néanmoins d'autre codeurs (magnétiques, inductifs, capacitifs…).

La mesure des déplacements des roues permet de reconstituer le mouvement global du robot. En partant d'une position initiale connue et en intégrant les déplacements mesurés, on peut ainsi calculer à chaque instant la position courante du véhicule. À chaque instant, les mesures odométriques nous donnent les déplacements des roues $\delta d$ et $\delta \varphi$ depuis l'instant précédent.

Comme tout capteur, le modèle d'odométrie à bien des limitations du moment que l'idée fondamentale de celui-ci est l'intégration d'information incrémentale du mouvement à travers le temps, ce qui mène inévitablement à l'accumulation d'erreurs, et particulièrement l'accumulation d'erreurs d'orientation qui causera une grande erreur dans la position .cette erreur croît proportionnellement avec la distance traversée par le robot.

Ces erreurs sont classées en deux types :

- Erreurs déterministes (systématiques) comme mauvais alignement des roues, incertitude sur le diamètre de la roue et/ou diamètre non constant, etc.... La résolution de ce type est faite par calibrage du système.

- Non-déterministes (aléatoires) sont des erreurs résiduelles qui mènent à des incertitudes de position au fur et à mesure comme un contact variable avec le sol (glisse, bosse, sol mou, déformation).

**1-6-2-2 Le télémètre laser :**

Un télémètre laser, également connu sous le nom laser rangefinder, est un télémètre qui utilise un faisceau laser pour déterminer la distance à un objet.

Les télémètres laser (LRF) fonctionnent tous selon le même concept de base. Le télémètre émet des faisceaux laser ; Ces faisceaux rebondissent sur des objets éloignés et l'horloge rapide du télémètre mesure le temps total écoulé entre le moment où les faisceaux ont quitté l'appareil et leur retour. Puisque nous savons à quelle vitesse le faisceau se déplaçait (vitesse de la lumière), l'appareil peut simplement utiliser cette mesure du temps pour calculer la distance parcourue, puis il affiche la distance jusqu'à l'utilisateur. Le faisceau d'onde émis est très concentré, ce qui permet d'avoir un cône d'émission très étroit et donc une bonne précision de mesure.

Le télémètre laser utilisé dans ce mémoire est un Hokuyo4 UST-10LX placé à l'avant de la chaise roulante, il scanne le plan horizontalement avec un champ de vision de 270 ° et a une limite de perception de 10 mètres, ce laser émet des ondes tout les 0.25°


**Figure 1.12 : Le télémètre laser Hokuyo UST-10LX**

## 1-7 Modélisation du robot :

Le robot que nous avons utilisé est un robot différentiel. Soit le rayon de courbure R de la trajectoire du robot, c'est-à-dire la distance du *CIR* au point O' (figure 1.13). Soit 2L la distance entre les deux roues, et la vitesse angulaire du robot par rapport au *CIR*. Alors les vitesses des roues droite et gauche, respectivement notées $v_d$ et $v_g$ vérifient :

$$v_d = (R+L)\,\Omega \qquad (1.1)$$

$$v_g = (R-L)\,\Omega \qquad (1.2)$$

On déduit R et Ω :

$$\Omega = (v_d - v_g)/2L \qquad (1.3)$$

$$R = L\,\frac{vd+vg}{vd-vg} \qquad (1.4)$$

La vitesse linéaire v du robot au point O' est donnée par :

$$V = (v_d + v_g)/2 \qquad (1.5)$$

La vitesse angulaire est égale à la vitesse de rotation autour du CIR :

$$\Omega = \frac{vd-vg}{2L} = \dot{\varphi} \qquad (1.6)$$

Ces équations expliquent deux propriétés particulières du mouvement des robots différentiels :
si $v_d = v_g$, la vitesse angulaire Ω sera nulle et le rayon de courbure *R* est infinie donc le robot se déplace en ligne droite.
si $v_d = -v_g$, $\Omega \neq 0$ et *R* est nulle alors le robot effectue une rotation sur lui-même.
Dans le cas où $v_d \neq v_g$ le déplacement du robot est un virage à gauche ou à droite (dans une direction qui correspond à la vitesse inférieur).

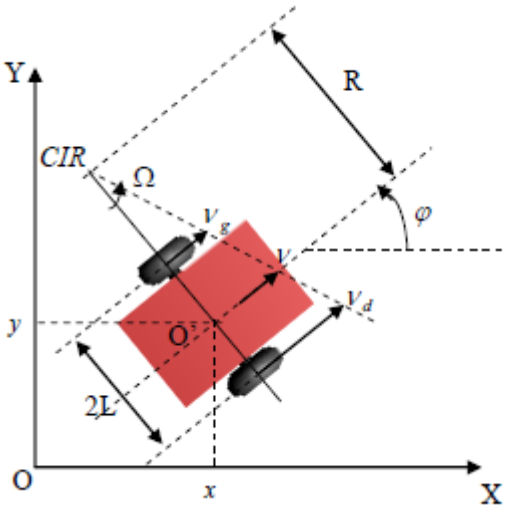



**Figure 1.13 : Centre instantanée de rotation CIR d'un robot mobile différentiel.**

## 1-7-1 Modèle cinématique :

Le modèle cinématique du robot différentiel est donné par les équations suivantes :

$$\dot{x}=v\cos\varphi \qquad (1.7)$$

$$\dot{y}=v\sin\varphi \qquad (1.8)$$

$$\dot{\varphi}=\Omega \qquad (1.9)$$

En intégrant ces équations on obtient la position de notre robot :

$$x(t)=\int_0^t v(\sigma)\cos\big(\varphi(\sigma)\big)\,d\sigma \qquad (1.10)$$

$$y(t)=\int_0^t v(\sigma)\sin\big(\varphi(\sigma)\big)\,d\sigma \qquad (1.11)$$

$$\varphi(t)=\int_0^t \Omega(\sigma)d\sigma \qquad (1.12)$$

N.B : Les roues folles n'interviennent pas dans la cinématique, mais ils assurent juste l'équilibre.

## 1-7-2 Le modèle odométrique d'un système robotique :

L'odométrie est une technique de localisation qui permet de déterminer la position du robot $(x, y, \varphi)$ par rapport au repère univers en intégrant des translations et rotations élémentaires. Pour un déplacement entre deux instants d'échantillonnages *k-1* et *k*, qui leurs corresponds respectivement les mesures odométriques ($xr_{k-1}$, $yr_{k-1}$, $\theta r_{k-1}$) et ($xr_k$, $yr_k$ , $\theta r_k$).

Le déplacement élémentaire en translation est donné par :

$$\delta d = \sqrt{(xr_k - xr_{k-1})^2 + (yr_k - yr_{k-1})^2} \qquad (1.13)$$

Le déplacement élémentaire angulaire est donné par :

$$\delta\varphi = \theta r_k - \theta r_{k-1} \qquad (1.14)$$

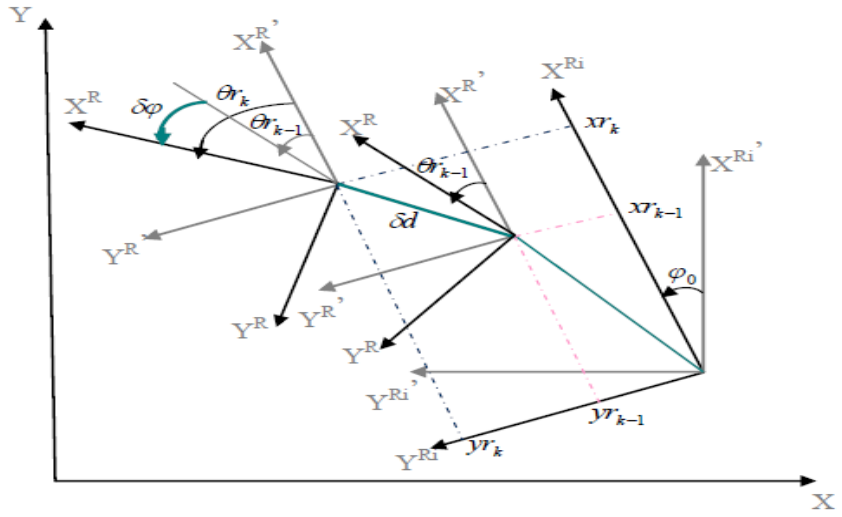



**Figure 1.14** : **Position du robot dans le repère univers $R^u$ en fonction des données odométriques.**

$$\dot{x}=(x_k - x_{k-1})/Te \qquad (1.15)$$

La discrétisation du modèle donne :

$$x_k=x_{k-1}-v_kTe \sin \varphi_{k-1} \qquad (1.16)$$

$$y_k=y_{k-1}-v_kTe \cos \varphi_{k-1} \qquad (1.17)$$

$$\varphi_k= \varphi_{k-1}+\Omega_{k-1}Te \qquad (1.18)$$

Modèle (1)

On faisant apparaitre les mesures odométriques $\delta d$ et $\delta\varphi$ on obtient le modèle :

$$x_k=x_{k-1}-\delta d \sin \varphi_{k-1} \qquad (1.19)$$

$$y_k=y_{k-1}- \delta d \cos \varphi_{k-1} \qquad (1.20)$$

$$\varphi_k= \varphi_{k-1}+ \delta\varphi \qquad (1.21)$$

Modèle (2)

Une autre méthode, plus précise lorsque la trajectoire est une courbe, consiste à supposer que le mouvement se fait localement suivant un arc de cercle de longueur $\delta d$.

Sous l'hypothèse d'un mouvement circulaire, on a :

$$\delta d = \rho \delta \varphi \qquad (1.22)$$

Où$\rho$Le rayon de courbure.

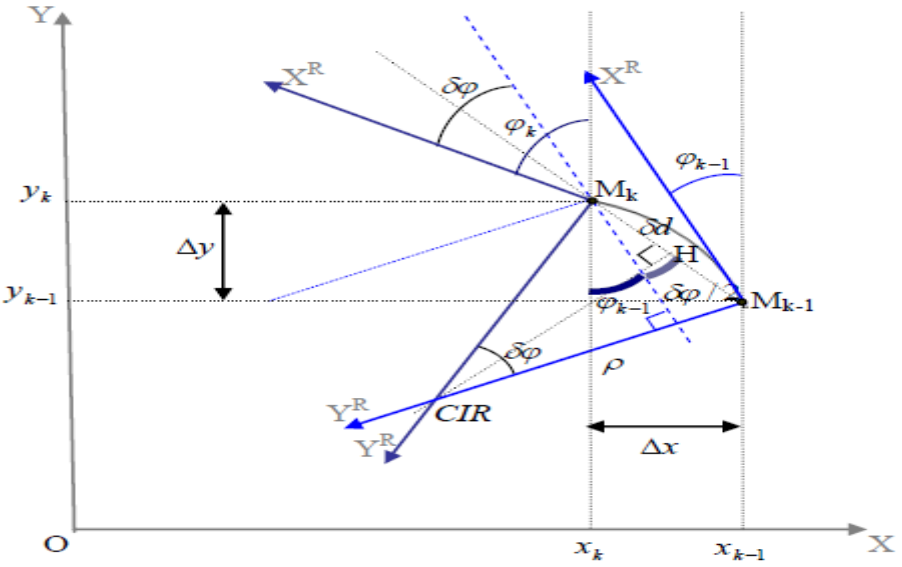



**Figure 1.15 : Déplacement circulaire du robot entre deux instants d'échantillonnage**

Le modèle odométrique correspondant est :

$$x_k = x_{k-1} - \delta d \sin(\varphi_{k-1} + \delta\varphi/2) \qquad (1.23)$$

$$y_k = y_{k-1} - \delta d \cos(\varphi_{k-1} + \delta\varphi/2) \qquad (1.24)$$

$$\varphi_k = \varphi_{k-1} + \delta\varphi \qquad (1.25)$$

Modèle (3)

Ce dernier modèle odométrique est l'un des plus utilisés. Par rapport au modèle (2), il apparaît une correction "$\delta\varphi/2$" dans les termes trigonométriques.

## 1-8 Conclusion :

Au cours de ce chapitre, nous avons présenté les différents types de robots mobiles à roues les plus utilisés. On a donné plus de détails sur le robot utilisé dans ce travail.

Le but est de permettre à la chaise roulante autonome de se déplacer dans un environnement dynamique inconnu en toute sécurité. Pour la navigation autonome d'un robot mobile, plusieurs approches ont été proposées dans ce domaine. Un état de l'art est présenté dans le chapitre suivant.

# CHAPITRE 2

# LA NAVIGATION REACTIVE D'UN ROBOT MOBILE :

# TRAVAUX CONNEXES

## 2-1 Introduction :

Le problème dans la navigation autonome des robots était toujours comment les conduire vers leur but en toute sécurité. Parmi les approches proposées pour résoudre ce problème les méthodes réactives, mais éviter les obstacles est insuffisant pour assurer la sûreté.

Notre objectif consiste à développer un système de navigation capable de conduire le robot vers son but dans un environnement dynamique inconnu, en utilisant uniquement les données sensorielles, tout en évitant les collisions et en considérant le comportement futur des obstacles mobiles de l'environnement, qu'ils soient perçus ou non perçus, ainsi que les contraintes cinématiques et dynamiques du robot. Ce système évite non seulement les états de collisions, mais il garantit aussi la sûreté du mouvement.

Ce chapitre présente un état de l'art sur les travaux connexes dans ce domaine. L'objectif de ce chapitre est de présenter les différentes approches de la navigation réactive concernant l'évitement des obstacles et la sûreté de mouvement, puis évaluer ces méthodes par rapport à plusieurs critères en montrant les avantages et les incontinents de chaque une d'elles .Enfin, choisir une méthode qui répond à nos besoins.

## 2-2 Définition de la navigation :

La navigation du robot signifie la capacité du robot à déterminer sa propre position dans son cadre de référence, puis à planifier un chemin vers un emplacement de but. Pour naviguer dans son environnement, le robot ou tout autre appareil de mobilité nécessite une représentation, c'est-à-dire une carte de l'environnement et la capacité d'interpréter cette représentation.

La navigation peut être définie comme la combinaison de quatre compétences fondamentales :

- localisation
- Planification de trajectoire
- Création de cartes et interprétation de cartes (modélisation de l'environnement)
- évitement d'obstacles

La localisation du robot indique la capacité du robot à établir sa propre position et son orientation dans le cadre de référence. La planification de trajectoire est en fait une extension de la localisation, en ce qu'elle nécessite la détermination de la position actuelle du robot et de la position d'un emplacement de but, tous deux dans le même cadre de référence ou de coordonnées. La construction de la carte peut prendre la forme de toute notation décrivant des emplacements dans le cadre de référence du robot. L'évitement d'obstacles utilise les données sensorielles, tout en évitant les collisions.

Plusieurs approches ont été proposées dans la littérature pour résoudre le problème de la navigation d'un robot mobile, mais généralement, nous distinguons deux grandes catégories d'approches : les approches délibératives qui représentent les approches de planification de mouvement et les approches réactives qui représentent les approches d'évitement d'obstacles [4]. Dans ce travail on s'intéresse à l'évitement d'obstacles.

## 2-3 Navigation réactive :

La navigation réactive est un paradigme bien connu pour contrôler un robot mobile autonome, qui suggère de prendre toutes les décisions de contrôle pour éviter tous les obstacles grâce à un traitement léger des données de capteur actuelles / récentes. Parmi les nombreux avantages de ce paradigme sont:

-la possibilité de l'appliquer à des robots avec des ressources matérielles limitées,

-le fait de pouvoir naviguer en toute sécurité avec un robot dans des environnements totalement inconnus contenant des obstacles mobiles imprévisibles.

Néanmoins, le paradigme réactif peut occasionnellement entraîner le piégeage des robots dans certaines zones de l'environnement - généralement, ces zones en conflit ont une grande forme concave et /ou sont pleines d'obstacles rapprochés.

À ce dernier égard, un effort énorme a été consacré pour surmonter un inconvénient aussi grave au cours des deux dernières décennies. À la suite de cet effort, un nombre important de nouvelles approches de navigation réactive ont été proposées. Certaines de ces approches ont clairement amélioré la façon dont un robot commandé de manière réactive peut se déplacer parmi des obstacles densément encombrés; certaines autres approches se sont essentiellement concentrées sur l'augmentation de la variété de formes et de tailles d'obstacles qui pourraient être contournées avec succès; etc.

## 2-4 Les méthodes de navigation :

### 2-4-1 La fenêtre dynamique (DW) :

Un robot mobile doit effectuer des tâches ciblées dans des environnements dynamiques et inconnus. Les algorithmes d'évitement d'obstacles réactifs locaux doivent être mis en œuvre et intégrés dans un module de commande de mouvement afin de fournir au robot cette capacité

L'approche par fenêtre dynamique [5] est une technique d'évitement réactif local basée sur l'espace de vitesse où la recherche de commandes contrôlant le robot est effectuée directement dans l'espace des vitesses. La trajectoire d'un robot peut être décrite

par une séquence d'arcs de lignes circulaires et droites. L'espace de recherche est réduit par les contraintes cinématiques et dynamiques du robot à une certaine plage de vitesses autour de vecteur actuel des vitesses ($v_c$, $\omega_c$) qui peuvent être atteintes dans l'intervalle d'échantillonnage $\Delta t$. La fenêtre dynamique $V_d$ contenant les vitesses atteignables possibles est définie comme

$$V_d = \left\{ (v,\omega) \mid \begin{array}{l} v \in [v_c - \dot{v}_b \Delta t, v_c + \dot{v}_a \Delta t] \wedge \\ \omega \in [\omega_c - \dot{\omega}_b \Delta t, \omega_c + \dot{\omega}_a \Delta t] \end{array} \right\}, \quad (2.1)$$

Où les accélérations $\acute{v}_a$ et $\acute{\omega}_a$b sont les accélérations maximales de translation et de rotation exercées par les moteurs et $\acute{v}_b$ et $\acute{\omega}$b sont les décélérations maximales de rupture en translation et en rotation. Un couple de vitesse (v,ω) du Vd est considéré comme sûr si le robot est capable de s'arrêter le long de la trajectoire définie par cette vitesse avant de heurter tout objet qui peut être rencontré le long de ce chemin.

Pour rendre la recherche de vitesses réalisable et appropriée pour une réponse réactive rapide, l'approche de fenêtre dynamique considère exclusivement le premier intervalle d'échantillonnage pour choisir le vecteur de vitesse optimal et suppose que les vitesses dans les n-1 intervalles d'échantillonnage restants sont constantes. Une capture des trajectoires possibles instantanées du robot à un moment donné et de la configuration locale d'obstacles déterminée par un espace de vitesse réduit est représentée sur la figure 2.1.

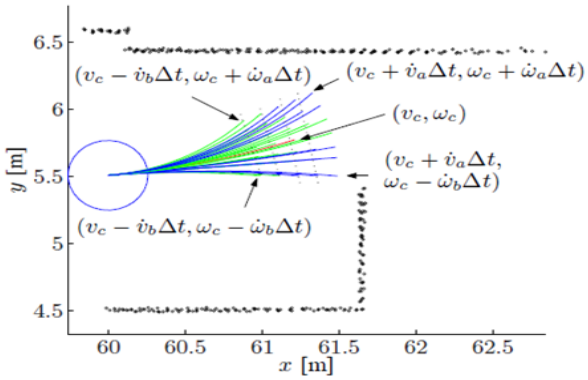



**Figure 2.1 capture des trajectoires possibles du robot**

## 2-4-2 Une méthode d'évitement d'obstacles utilisant les réseaux de neurones (neural network)[6] :

Au cours des dernières années, il y a eu des progrès significatifs dans la navigation de robot mobile en extérieur appliquée dans des environnements de plus en plus complexes. Cependant, il est encore nécessaire d'étudier plus en profondeur les problématiques liées aux systèmes autonomes pour les fiabiliser en milieu urbain.

Cette partie met en œuvre un système de transport automatisé permettant aux passagers d'un robot semblable à une voiture d'atteindre leur destination en toute sécurité, de manière intelligente et autonome [7]. Différents composants du système sont décrits comme la localisation basée sur Hector SLAM, la détection d'humains basée sur le descripteur HOG (Histograms of Oriented Gradient) et la navigation intelligente basée sur l'approche FNN (Fuzzy neural networks).

De nombreux projets utilisant ces véhicules autonomes ont été développés, Parmi eux : DARPA urban 2007 one [8], The Tsukuba challenge [9, 10, 11], Dans [7], une navigation sans conducteur pour un micro-bus est présentée. La méthode permet au micro-bus de passer d'un point de latitude-longitude à un autre tout en évitant les obstacles à l'aide d'un capteur lidar. L'itinéraire prévu est modifié pour éviter l'obstacle détecté et le bus continue son chemin vers la destination dans un circuit privé.

Le but de cette méthode est de présenter les différents composants et de mettre en évidence les possibilités réalisées par le système de transport.

Ce robot exécute un cadre de navigation autonome, qui est composé de trois modules différents: localisation, détection visuelle des humains et navigation locale intelligente (figure 2.2)

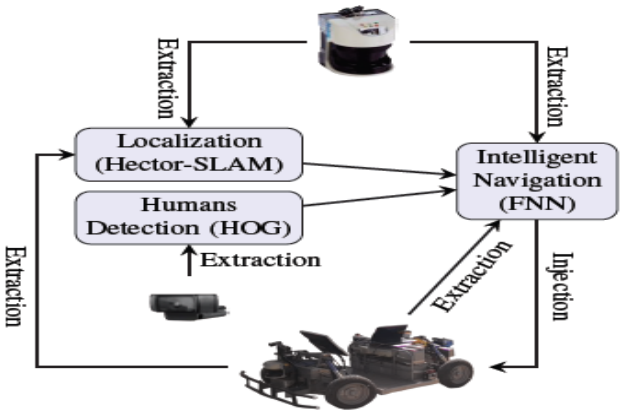



**Figure 2.2 : diagramme de la navigation**

Le robot utilise ses capteurs externes (télémètre laser et caméra) dans leurs différents composants.

Il extrait la vitesse et l'angle de braquage et les envoie à la localisation et à la navigation intelligente. Le module de navigation calcule la bonne vitesse et l'angle de braquage et les réinjecte au robot. Le robot applique la méthode Hector SLAM (Simultaneous Localization and Mapping) pour estimer la position 2D du robot, une méthode basée sur HOG (Histograms of Oriented Gradient) pour détecter visuellement les humains et les voitures et une méthode basée sur FNN (Fuzzy neural networks) pour rechercher localement la destination tout en évitant les obstacles rencontrés après l'apprentissage et l'adaptation.

## 2-4-3 La méthode SURF pour les UAV [12] :

Les véhicules aériens sans pilote (UAV) sont utilisés dans plusieurs applications telles que la cartographie, le journalisme, le transport, les applications militaires de sauvetage et les environnements auxquels un humain ne peut pas accéder. Parfois, il est difficile de manipuler l'UAV en raison d'une perte de visibilité ou du fait que le système de position globale (GPS) n'est pas disponible.

Cette méthode permit la détection d'obstacles connus en temps réel basé sur des points caractéristiques, et une modélisation hors ligne de la MAV (micro UAV) pour concevoir un contrôleur pour l'évitement d'obstacles fixes en environnement inconnu.

Deux images sont utilisées pour la détection des objets, l'une est située dans une base de données contenant des obstacles et l'autre est capturée avec la caméra embarquée. Afin de trouver une correspondance entre ces images, la détection, la description et la correspondance des points caractéristiques sont utilisées. De plus, la zone d'obstacle et le centre de masse à utiliser comme cible sont calculés dans le contrôleur. Pour la détection de point de caractéristique, cette proposition utilise SURF car son coût de calcul est inférieur sans réduire la robustesse.

Il est nécessaire de trouver la correspondance entre l'image de la base de données et l'image capturée avec le drone (figure 2.3) et puis comparer les points caractéristiques avec le même type de contraste, ce qui permet d'obtenir un coût inférieur sans réduire les performances du descripteur.

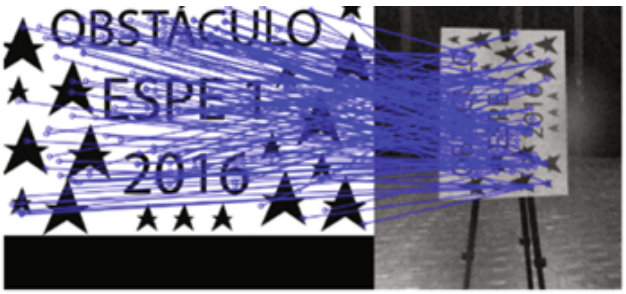



**Figure 2.3 : la correspondance entre une image de la base de données et une photo prise en temps réel**

La zone d'obstacle est inversement proportionnelle à la distance entre l'obstacle et le drone, mais la perspective de la caméra déforme la géométrie de l'obstacle. Il est nécessaire de compenser la déformation de la perspective de l'obstacle à l'aide d'une transformation géométrique. De plus, le centre de masse est estimé par la valeur moyenne des coordonnées x et y de chaque sommet pour le rectangle, étant une figure régulière. Le système de contrôle du MAV manipule quatre actions de contrôle différentes: tangage, roulis, lacet et altitude ou les actions de contrôle sont les entrée (figure 2.4a) et les vitesses sont les sorties (figure 2.4b). Deux mouvements dans le plan x et y étaient proposés pour le système d'évitement, dans l'axe x le mouvement est uniforme, c'est-à-dire que la vitesse linéaire x est

constante. Le mouvement dans l'axe y dépend de l'emplacement de l'obstacle, donc la loi de commande sera appliquée sur cet axe.

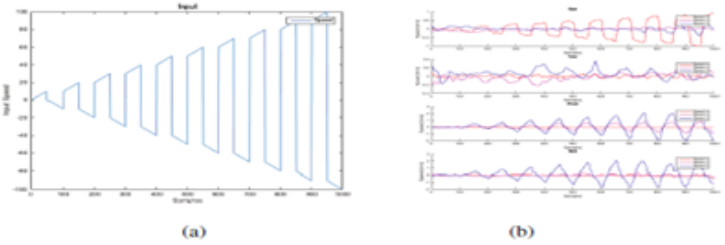



**Figure2.4:comportement du système envers différentes entrées (a)entrée vitesse (b) sorties sur les axes ( x,y,z)**

## 2-4-4 Control de permutation pour l'évitement d'obstacles sur les UAVs [13] :

Le risque de collision spatiale inattendue des aéronefs augmente lorsqu'ils partagent le même espace aérien avec d'autres véhicules. Cette méthode présente une approche réaliste d'évitement de collision en 3D pour les véhicules aériens sans pilote (UAV) à voilure fixe dans un environnement non coopératif qui fait référence à un environnement dans lequel un aéronef hôte n'a aucune information préalable sur la trajectoire de vol d'un intrus et une communication coopérative entre eux n'est pas disponible. Les approches non coopératives sont les aspects les plus difficiles du problème étant donné la grande incertitude dans l'état de l'intrus. La stratégie proposée vise à atteindre le relèvement relatif souhaité dans le plan horizontal et l'élévation relative dans le plan vertical afin que l'aéronef hôte puisse éviter une collision avec l'aéronef intrus en 3D.

L'aéronef hôte suivra une trajectoire souhaitée dans la trajectoire d'évitement de collision et reprendra la trajectoire préétablie une fois la collision évitée. La condition d'arrêt en approche est déterminée pour que l'aéronef hôte déclenche une manœuvre d'évasion pour éviter une collision en termes de cap mesuré. Un contrôleur de commutation est conçu pour réaliser la stratégie d'évitement de collision spatiale.

L'évitement de collision pour les aéronefs a fait l'objet d'une enquête approfondie dans un nombre considérable d'articles, et des efforts importants ont été faits pour résoudre différents problèmes en divers scénarios.

La présente recherche fait partie des efforts consacrés à la conception d'un système de détection et d'évitement (SA) pour éviter les collisions dans l'espace aérien.

Ce système ne repose que sur la détection de signaux émanant des cibles elles-mêmes.

Récemment, la détection de mouvement à l'aide de plusieurs caméras offre un moyen intéressant de développer un système SA en raison du coût, de la taille et de la puissance relativement faibles des capteurs.

Les résultats démontrent que l'approche proposée peut effectivement éviter les collisions spatiales, ce qui la rend adaptée à l'intégration dans les systèmes de commande de vol des UAV.

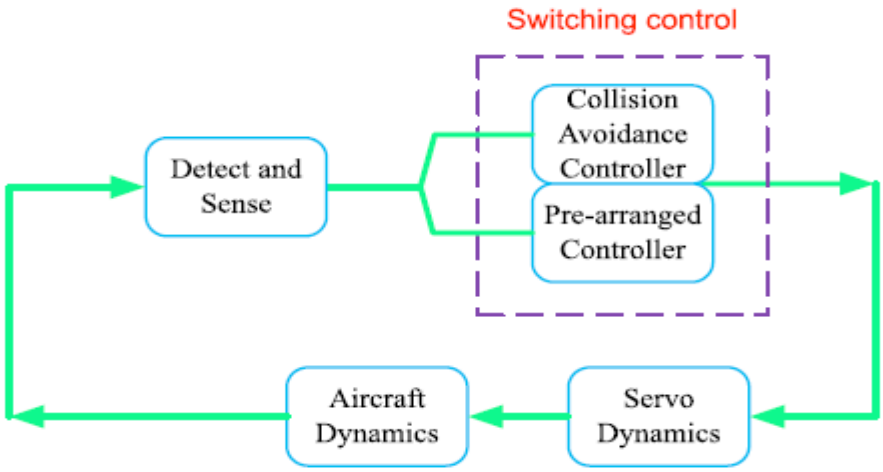



**Figure 2.5 control de permutation pour l'évitement d'obstacles**

## 2-4-5 Représentation des obstacles dans l'espace des vitesses :

La méthode des obstacles de vitesse (VO) [14] définit un ensemble de vitesses, notées $VO_{A/B}$ pour l'agent A par rapport à l'agent B, qui conduisent à une collision dans le futur, en supposant que la vitesse de l'agent B est constante dans le temps.

Par conséquent, s'il est possible de sélectionner la vitesse de l'agent A en dehors de $VO_{A/B}$, l'agent A n'entrera pas en collision avec l'agent B. Les agents de ce travail sont considérés comme des objets dynamiques de forme circulaire dont les limites décrivent des zones de sécurité typiquement définies autour des robots**.** Il faut spécifier une distance de séparation minimale au point d'approche (CPA) le plus proche de l'obstacle.

Par souci de simplicité, la définition de VO est présentée en utilisant le Collision Cône $CC_{A/B}$ qui considère la vitesse relative ($v_{A/B}$), au lieu de $VO_{A/B}$, qui représente une condition équivalente sur la vitesse absolue, $v_A$.

Autrement dit, $VO_{A/B}$ est obtenu en translatant $CC_{A/B}$ par $v_B$.

$$VO_{A/B} = CC_{A/B} \oplus v_B \qquad (2.2)$$

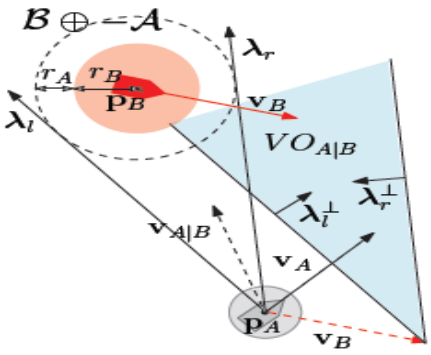



**Figure 2.6 : cône de collision $CC_{A/B}$ translaté de $v_B$**

Des extensions de VO ont été proposées pour améliorer ses performances et remédier ses limitations, tels que la méthode RVO [15,16] qui suppose que les obstacles utilisent le même protocole d'évitement, le robot dans cette méthode doit tenir compte les vitesses de chaque objet afin de les éviter, la méthode DRVO [17] qui intègre le comportement interactif des agents et il est proactif face à l'incertitude du comportement futur des obstacles.

Les méthodes SVO [18,19] et IVO [20,21] sont établies afin de rendre l'évitement plus sûr.

Van den Berg [22] a introduit l'algorithme ORCA qui considère le comportement réactif entre les UAV et applique l'algorithme à l'évitement de collision du robot, Afin de surmonter les limitations du concept VO et RVO.

## 2-5 La sureté de mouvement [1]:

Dans un proche avenir, on s'attend à ce que les systèmes robotiques partagent les espaces de vie et de travail humains: les robots de service et les systèmes de transport cybernétique sont des exemples de deux domaines d'application principaux. La technologie est maintenant mature; nous avons déjà vu des robots mobiles autonomes guidant les gens dans les musées et des voitures automatisées circulant sur le réseau routier.

En déplacement (en particulier à grande vitesse), les véhicules automatisés, les manipulateurs mobiles et les robots humanoïdes peuvent être potentiellement dangereux en cas de collision. Avant de laisser de tels systèmes robotiques transporter ou partager l'espace avec des personnes de manière réellement autonome, il est essentiel d'affirmer et de caractériser leur sécurité de mouvement, c'est-à-dire leur capacité à éviter les collisions. Il ne suffit pas de démontrer qu'un système robotique fonctionne correctement sur un ensemble limité d'expériences. Si jamais des robots autonomes doivent être déployés chez l'homme à grande échelle, il est nécessaire de caractériser le niveau de sécurité de mouvement qui peut être atteint et/ou de préciser les conditions dans lesquelles il peut être garanti. Les collisions se produisent pour des raisons qui appartiennent généralement à l'une des catégories suivantes:

-Pannes matérielles, par ex. défaillance des freins.

-Bogues logiciels, par exemple erreur de troncature.

[1] https://link.springer.com/

-Erreurs de perception, c'est-à-dire toutes les erreurs liées au système de détection du robot et qui conduisent le robot à une mauvaise compréhension de son environnement (par exemple, un faux négatif).

-Erreurs de raisonnement, c'est-à-dire à un moment donné, une mauvaise décision a été prise.

Les roboticiens connaissent depuis longtemps le problème de la sureté de mouvement. La sureté des mouvements des systèmes robotiques fonctionnant dans le monde réel est essentielle (en particulier lorsque leur taille et leur dynamique les rendent potentiellement nocifs pour eux-mêmes ou pour leur environnement). Elle est une notion prise pour acquise et mal définie. Dans la littérature sur la robotique Il existe une riche littérature sur l'évitement des collisions à partir des travaux pionniers de Moravec (1981). La question de la sécurité des mouvements a été récemment explorée à un niveau abstrait par Fraichard [2] ou il a mis en évidence un certain nombre d'exigences dont la violation est susceptible de mettre un système robotique en danger et de provoquer des collisions

### 2-5-1 La différence entre évitement d'obstacle et sureté :

La sureté est un niveau plus élevé de l'évitement d'obstacles, la sureté de mouvement implique qu'il n y est jamais collision quelque soit la trajectoire prise par le robot. Pour savoir si une méthode s'agit de sureté de mouvement ou de l'évitement d'obstacle on vérifie des critères précis qu'on va les expliquer dans la section suivante.

### 2-5-2 Les types de sureté :

### 2-5-2-1 Sureté absolue[1] :

La sûreté de mouvement nécessite non seulement que la trajectoire du robot soit sans collision mais également doit être sans ICS (inévitable collision states), i.e. le robot doit être toujours dans un état où une trajectoire d'évitement est disponible et elle doit être toujours définie par rapport au modèle du futur qui est a priori connu, une connaissance complète de l'environnement dans un horizon temporel infini est demandée. Ceci n'est pas le cas du monde réel, on n'a aucune information sur l'environnement ou son évolution dans le futur et le robot se base que sur les informations qu'il collecte de ses captures.la sureté absolue dans ce cas devient impossible, et on se contente d'un niveau plus faible de sûreté de mouvement ; étant raisonnable, mieux vaut garantir moins que ne rien garantir. Le plus important est que la sûreté est garantie malgré les contraintes sévères imposées par le champ de vision limité.

### 2-5-2-2 La sureté passive [23]:

Un état s est sûr ou PS (p- sûr) si et seulement si il existe au moins une manœuvre de freinage à partir de s et sans collision jusqu'à Tb, avec Tb l'heure où R est au repos (le temps de freinage).

### 2-5-2-3 La sureté amicale [23]:

un état s est en sécurité ou PFS (ou pf- sûr) si et seulement si il existe au moins une manœuvre de freinage à partir de s et sans collision jusqu'à Tb + Tob, avec Tb le temps de freinage de R, et Tob le freinage maximal heure des objets en mouvement présents dans l'environnement.

## 2-6 Les méthodes ICS :

### 2-6-1 Inevitable collision states ICS [24]:

En raison de sécurité un système de robot ne doit jamais se trouver dans un état où il n'y a pas de trajectoire possible pour éviter une collision avec un obstacle. Un tel état est un état de collision inévitable (ICS).

ICS est un état de collision inévitable pour un système robotique, il peut être défini comme un état pour lequel, quelle que soit la trajectoire future suivie par le système, une collision avec un obstacle se produit finalement, quel que soit le comportement du robot. Le concept ICS est particulièrement utile pour la navigation dans des environnements dynamiques car il prend en compte le comportement futur des objets en mouvement. En conséquence, il nécessite un modèle de l'évolution future de l'environnement. Dans le monde réel, les trajectoires futures des obstacles sont généralement inconnues et seules des estimations sont disponibles.

### 2-6-2 ICS probabiliste [25] :

Le but de cette méthode est précisément d'étudier dans quelle mesure le concept de ICS peut être étendu pour gérer l'incertitude inhérente à l'avenir. Les ICS probabilistes permettent de caractériser la probabilité de sécurité de mouvement d'un état donné, une probabilité qui peut ensuite être utilisée pour concevoir des stratégies de navigation sûres dans des situations réelles. Cette méthode présente également deux vérificateurs probabilistes ICS, c'est-à-dire des algorithmes qui déterminent la probabilité de sécurité de mouvement d'un état donné. La première est la version probabiliste de l'ICS-Checker présentée dans [26] (backward version). Le second PIC-Checker (forward version) est nouveau et plus efficace.

Un ICS probabiliste peut être défini comme la probabilité d'être dans un ICS ou de manière équivalente comme la probabilité de collision minimale de toutes les trajectoires de collision possibles.

Les résultats obtenus avec PICS-CHECK englobent ceux d'ICS-CHECK démontrant la capacité des algorithmes probabilistes à intégrer l'incertitude du modèle du futur.

### 2-6-3 ICS de freinage [4] :

La navigation d'un robot mobile avec un champ de vision limité dans un environnement dynamique inconnu présente un problème dans la robotique. Apres plusieurs approches qui ont été développées pour résoudre ce problème, en criant des variantes du concept ICS tels que :

- Approximations de l'ICS: de telles approximations n'étant pas conservatrices, la garantie de la sureté de mouvement est perdue.

- ζ-Sureté: le robot est garanti de rester dans des états où il est en sécurité pendant une durée donnée (espérons qu'elles sont suffisantes pour calculer une trajectoire de sécurité mise à jour).

-Trajectoires évasives : ils garantissent que le robot ne peut être que dans des états où il est possible d'exécuter une trajectoire évasive, par ex. une manœuvre de freinage pour une voiture ou une manœuvre indirecte pour un avion ou un navire.

L'ICS$^b$ est développé pour résoudre ce problème d'une façon plus optimale. Un ICS de freinage (ICS$^b$) est défini de manière informelle comme un état pour lequel quelle que soit la future trajectoire de freinage suivie par le robot A, une collision se produit avant qu'A ne soit au repos.

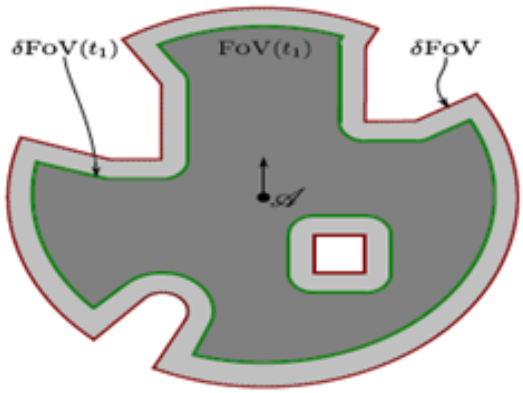



**Figure 2.7: modèle conservatif de futur (le rétrécissement de FoV)**

Un certain nombre de propriétés importantes sont établies pour ce concept [4]. Ces propriétés sont ensuite utilisées pour concevoir ICS$^b$ checker, c'est-à-dire qu'un algorithme qui vérifie si un état donné est un ICS$^b$ ou non. Pour valider le concept ICS$^b$ et démontrer son utilité, l'algorithme ICS$^b$ –checker est intégré dans un schéma de navigation réactive appelé PASSAVOID [4,27].

Le modèle de futur utilisé dans ce cas est le modèle conservatif ou tous les mouvements futurs possibles des obstacles sont pris en considération.

Cela est fait comme suit :

∂FOVu ∪ FOVC (les objets non perçus): chaque point de cet ensemble est modélisé comme un disque qui grandit avec le temps (c'est-à-dire un cône dans l'espace × temps).

- ∂FOVf (les objets fixes): chaque point de cet ensemble reste constant dans le temps (c'est-à-dire une ligne verticale dans l'espace × temps).

- ∂FOVm (les objets en mouvement): si l'information sur leur comportement futur est disponible et fiable, chaque point de cet ensemble est modélisé en conséquence (c'est-à-dire une courbe dans l'espace × temps), sinon il est traité comme un objet non perçus et modélisé comme un disque en croissance.

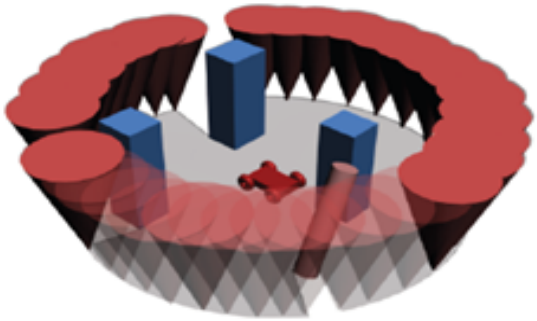


**Figure 2.8: modèle conservatif de futur (représentation partiel pour des buts de visualisation)**

C'est bien sûr le cas lorsque les capteurs de A peuvent différencier les objets fixes et mobiles. Si ce n'est pas le cas, alors chaque point de ∂FOV est modélisé comme un disque qui grandit avec le temps (c'est-à-dire un cône dans l'espace × temps).

## 2-6-4 Sureté des mouvements des navires basée sur les états de collision inévitables de distance [28] :

Beaucoup de recherches ont été faites dans le domaine de l'évitement d'obstacles dans les rencontre rapprochée des navires générant des trajectoires évasives pour améliorer la sureté de mouvement dans le domaine maritime, entre eux on distingue : la méthode de la navigation électronique [29] présenté par l'IMO. La validation de ces trajectoires vérifiant les états de collision inévitables pour maintenir un état pour lequel une trajectoire évasive est disponible est proposée dans cette méthode.

L'approche de la sureté de mouvement des navires utilise les distances et ses progressions à tous les autres navires pour s'assurer qu'une manœuvre indirecte peut être effectuée à tout moment pour une trajectoire donnée. La sureté est garantie, s'il existe, pour un état actuel, une trajectoire évasive qui conduit à des distances constantes ou croissantes à tous les autres navires sans enfreindre des limites de sécurité, ce qui est définit comme le concept des états de collision inévitables basés sur la distance ($ICS^d$). Pour simplifier la modélisation de futur, les obstacles sont considérés comme des pts, leurs

dimensions sont utilisées pour le calcul de la distance minimale de sureté. Deux modèles de mouvement sont utilisés pour prédire les positions futures des autres navires. le modèle de vitesse constante (CV) est appliqué pour prédire les mouvements en lignes droites ou les positions, longitudinale et latérale, $p = (x, y)^T$ et les vitesses $v = (\dot{x}, \dot{y})^T$ sont utilisées plus la direction de mouvement qui est utilisée comme orientation du navire $\theta$ et le modèle de la vitesse de rotation constante (CTRV) pour les mouvements circulaires qui ajoute la vitesse de piste constante $v$ et la vitesse de rotation non nulle $\omega$ plus les paramètres considérés dans le modèle CV, dans le cas de $v=0$ la méthode CV est utilisé. Pour le propre navire, un troisième modèle de suivi de trajectoire est utilisé. Le modèle de suivi de trajectoire est utilisé pour prédire le mouvement futur du navire voyageant le long de la trajectoire donnée R. En plus de la position, de l'orientation et de la vitesse, ce modèle comprend les points de cheminement W de la trajectoire.

Pour un navire voyageant le long d'une trajectoire, deux classes de sureté de mouvement sont définies. Le premier est appelé la sureté évasive et passive de mouvement et peut être appliquée à des scénarios où le voyage se termine dans un endroit sûr, par ex. une jetée ou un point d'ancrage comme illustré à la figure 2.9a. Dans ce cas, ce concept utilisant des trajectoires de freinage et $ICS^b$ peut être appliqué pour une zone autour de la position de destination. Cette zone est également appelée zone de freinage et est représentée par un cercle orange sur la figure 2.9a. Ainsi, pour un voyage sûr le long de la trajectoire, la sureté d'un navire en mouvement continu doit être garantie jusqu'à un certain point lorsqu'il atteint cette zone. Par conséquent, le concept $ICS^d$ est appliqué à la trajectoire depuis un point de départ jusqu'à cette zone de freinage. En raison du rétrécissement du champ de vision, une position de repos doit être atteinte avant que l'espace de travail ne soit réduit à un point. Cela signifie que le temps $t_h$ nécessaire pour atteindre une position de repos doit être inférieur au temps $t_{FoV}$ jusqu'à ce que le champ de vision soit réduit à un point. Pour les scénarios où cela ne peut être garanti ou pour des trajectoires ne menant pas à un endroit sûr, le deuxième niveau de sureté plus faible appelé sureté évasive conditionnelle est définie. Pour ces trajectoires, l'arrêt n'est pas une solution réalisable et, par conséquent, le modèle de mouvement CV est appliqué au point où le propre navire atteint le dernier point de cheminement pour prédire son mouvement futur comme le montre la figure 2.9b. Pour ce cas, le champ de vision rétrécissant est négligé pour garantir à un navire qu'il existe une trajectoire d'évitement circulaire, en considérant uniquement les objets actuellement connus. L'apparition de nouveaux objets n'est pas prise en compte. Le concept d'$ICS^d$ est appliqué à tous les objets actuellement connus en considérant un horizon temporel fini.

(a) Une trajectoire qui se termine dans un endroit sur (b) trajectoire qui continue avec un modèle CV

(Flèche rouge) dans le dernier point de cheminement

**Figure 2.9 : une trajectoire typique pour une navigation en sureté d'un navire**

## 2-6-5 Une sureté de mouvement basée sur les nœuds de viabilité [30]

Éviter les états de collision inévitables (ICS) est la condition d'un mouvement sûr d'un robot. Cependant, la caractérisation de l'ensemble ICS est un défi. Plusieurs méthodes d'approximation ont été proposées, dont la plupart sont soit trop conservatrices, soit ne fournissent pas de garanties de sécurité de mouvement appropriées. Afin d'améliorer les garanties de sécurité, cette méthode est basée sur la théorie de la viabilité et un algorithme conçu est adapté pour approcher viability kernel, un concept similaire à l'ICS. L'algorithme est appliqué en premier à un scénario d'environnement statique difficile. Il est ensuite étendu pour gérer les environnements dynamiques. S'il n'est en général pas possible d'assurer indéfiniment la sécurité, nous parvenons néanmoins à atteindre une sécurité de mouvement infinie dans deux cas particuliers.

La théorie de la viabilité [31] étudie l'évolution des systèmes dynamiques et leur capacité à satisfaire les contraintes de viabilité. Les contraintes de viabilité définissent généralement un sous-ensemble de l'espace d'états du système, l'espace admissible. Un état viable est garanti d'avoir au moins une séquence de commandes qui, lorsqu'elles sont appliquées à partir du l'état déclaré, empêchent le système de tomber en panne, c'est-à-dire le maintiennent dans l'espace admissible. À l'inverse, les états non viables sont ceux où l'échec n'est plus évitable. Notez que lorsque l'évitement de collision est la contrainte de viabilité, les états non viables sont des états de collision inévitables. Le calcul VK et ICS sont donc liés et comme il existe des algorithmes qui calculent des approximations de VK, il est intéressant d'explorer s'ils peuvent être utilisés dans un contexte d'évitement de collision.

L'algorithme VK de [32] est exploité pour fournir une bonne approximation de l'ensemble ICS (bon dans le sens où il garantit la sécurité de mouvement de manière moins conservatrice). Premièrement, l'algorithme est adapté pour être conservateur dans les environnements statiques. L'algorithme est évalué dans le cas d'un robot imparable naviguant dans des passages étroits. Ensuite, l'algorithme est étendu pour gérer les environnements dynamiques. Bien qu'il soit en général impossible de garantir indéfiniment

la sécurité des mouvements, nous établissons et démontrons qu'elle peut être réalisée pour deux classes d'environnements dynamiques: la classe de congélation où l'environnement devient statique à un moment donné, et la classe périodique où les obstacles mobiles ont un comportement périodique.

L'algorithme de viabilité fonctionne en deux étapes. Il se rapproche d'abord du problème continu d'origine en le discrétisant dans le temps et dans l'espace. Ensuite, il calcule le VK exact pour le problème discrétisé de manière itérative.

L'algorithme de viabilité d'un système robotique est implémenté dans différents scénarios d'espace de travail. Pour chaque scénario, nous avons calculé le VK approximatif et la carte de régulation correspondante. Ensuite, nous avons utilisé cette carte de régulation pour naviguer en toute sécurité dans l'espace de travail. Le schéma de navigation est assez simple. A partir d'un état du VK, le robot choisit à chaque pas de temps, une commande à appliquer parmi les commandes viables disponibles à l'état présent. Cet ensemble de contrôles viables est déterminé selon la carte de régulation. Le choix de la commande dépend de l'objectif que le robot est sur le point d'atteindre, et cela maximiserait une fonction d'utilité définie.

## 2-7 Discussion :

Une approche originale intégrée du contrôle de robot mobile traitée ici est étendue avec une capacité de mouvement de robot sans collision en présence d'obstacles en mouvement. L'approche est basée sur l'intégration de l'algorithme de DW pour la génération de trajectoires robotiques admissibles sans collision. Le mouvement d'un obstacle est considéré comme le mouvement des cellules mobiles occupées dans une carte quadrillée. La trajectoire prédite de chaque cellule en mouvement est utilisée pour le calcul de collision avec les trajectoires possibles du robot. Cet algorithme nécessite un vecteur de vitesse et un cap de mouvement pour chaque cellule mobile mais cette approche suppose une connaissance a priori des trajectoires des obstacles en mouvement, cela n'est pas toujours possible dans le monde réel disant impossible a être réalisé. En plus les paramètres nécessaires dans l'algorithme sont considérés connus ce qui peut conduire a une collision. Cette méthode ne prend pas en considération les ICS ni la notion de l'horizon temporel.

Ainsi, le concept qui décrit un système de transport autonome extérieur. Dans ce système, trois composants importants ont été introduits: la localisation Hector-SLAM, la détection visuelle des humains basée sur HOG et la navigation locale FNN. Des expériences sur un robot ressemblant à une voiture ont montré que ce robot est capable de se déplacer de manière autonome et intelligente dans une zone piétonne confinée. Il peut également transporter des passagers à bord et les conduire à destination.

La méthode proposant un système d'évitement d'obstacles pour les drones utilisant une caméra monoculaire L'approche la plus simple traitée dans ce chapitre. Cette méthode garantit la flexibilité de l'UAV sans toucher au coût, l'évitement d'obstacle est optimal,

rapide et avec des performances plus élevées. Mais pour que cela soit réalisable une base de données qui contient les images de tous les obstacles qui pourrons être face a l'UAV, cela peut être uniquement fait dans un environnement d'intérieure ou on a une connaissance complète de l'environnement, le système n'a que détecté puis évité ces obstacles.

Une stratégie d'évitement de collision spatiale réalisable est proposée en 3D. L'UAV hôte est contrôlé pour maintenir une distance relative sûre par rapport à l'intrus en gardant le cap et l'élévation relatifs souhaités. Le système de commande de commutation est également conçu pour déterminer le moment de déclenchement de la stratégie d'évitement de collision. La performance de la stratégie d'évitement de collision proposée est vérifiée dans des scénarios de collision typiques. Il est démontré que l'évitement des collisions peut être atteint en utilisant la stratégie proposée. Cette méthode dirige l'avion vers un niveau de sécurité mais n'est pas sûr car ce concept ne tient pas compte le comportement futur des obstacles ni l'horizon temporel approprié. La stratégie proposée peut être appliqué pour des scénarios lorsque les UAV sont à vitesse variable.

La méthode VO peut être appliquée dans des environnements dynamiques, elle suppose que les obstacles se déplacent avec une vitesse linéaire constante ce qui n'est pas le cas en pratique, car généralement les obstacles mobiles suivent des trajectoires arbitraires.

L'approche d'ICS probabiliste rend compte de l'incertitude affectant la prédiction de l'évolution future des objets en mouvement. Deux nouveaux algorithmes de vérification probabiliste ICS ont été introduits et comparés à leur homologue déterministe. Les résultats obtenus avec PICS-CHECK englobent ceux d'ICS-CHECK démontrant la capacité des algorithmes probabilistes à intégrer l'incertitude du modèle du futur. En examinant les deux algorithmes, nous avons constaté que la version arrière offre une bonne réponse théorique sans limite de temps de recherche, mais coûte cher en calcul. En revanche, la version avancée est plus efficace mais nécessite la définition d'un horizon temporel d'anticipation qui impacte directement les résultats.

Les concepts $ICS^b$ et $ICS^d$ résolvent le problème de la navigation dans un endroit inconnu avec un champ de vision limité dans le cadre de la sureté passive, la sûreté de mouvement passive est intéressante pour deux raisons : (1) elle permet de fournir au moins une forme de garantie de la sûreté de mouvement dans des scénarios rigoureux, et (2) si chaque objet mobile dans l'environnement l'applique, alors aucune collision n'aura jamais lieu. Mais, dans certaines applications, la sûreté de mouvement passive peut être limitée, vu qu'elle s'occupe uniquement de la nocivité du robot envers l'environnement, i.e. quand une situation de collision survient, le robot prend toutes ses responsabilités sans les partager avec les autres obstacles de l'environnement. Il serait plus intéressant d'explorer des niveaux de sûreté plus élaborés comme la sûreté de mouvement amicalement passive: elle garantit que si une collision a lieu, le robot sera à l'arrêt et que l'objet en question aurait eu le temps de s'arrêter ou éviter la collision (s'il le désirait). Un tel niveau de sûreté de mouvement suppose que les objets mobiles ont des capacités cognitives, qu'ils peuvent aussi réagir et éviter les collisions et pour lesquels une certaine connaissance de leurs propriétés dynamiques est nécessaire (ce qui peut s'appliquer dans plusieurs situations). En

général, il peut être intéressant d'explorer d'autres formes de sûreté de mouvement en fonction de la particularité du problème de navigation considéré. De sa part $ICS^b$ prends en compte tous le mouvement possible des objets présents dans l'environnement mêmes ceux qui ne sont pas perçus, cela induit à un chant de vision rétrécit a travers le temps. Contrairement à ce concept, $ICS^d$ donne plus de liberté de mouvement au robot en négligeant les objets non perçus, mais cela peut conduire une collision si un ICO existe juste après la frontière du champ de vision de système robotisé.

Étant donné un système dynamique et un sous-ensemble d'états satisfaisant un ensemble donné de contraintes, le VK est le sous-ensemble d'états à partir duquel le système peut être maintenu pour toujours dans le sous-ensemble contraint. Notez que, lorsque les contraintes doivent éviter la collision, le VK est en fait le complément de l'ensemble ICS. L'algorithme de viabilité est utilisé comme moyen d'approximer le VK, et donc l'ensemble ICS, pour assurer la sécurité de mouvement des systèmes robotiques mobiles. Il a fallu apporter quelques modifications à l'algorithme pour obtenir une approximation conservatrice du VK, et aussi pour faire face aux environnements dynamiques. La méthode a montré de nombreux atouts ainsi que des lacunes. Un avantage est qu'il peut être utilisé pour calculer le VK pour des systèmes avec n'importe quelle dynamique et de n'importe quelle dimension, au moins théoriquement. Deuxièmement, il ne nécessite pas de prédéfinir un ensemble de trajectoires évasives; à la place, il les recherchera tous par force brute. Cela revient à dire que la résolution est complète. A tout état donné, il trouvera une trajectoire évasive, s'il en existe une sous cette discrétisation. Cela serait particulièrement utile lorsqu'aucun choix clair de trajectoires d'évitement n'est disponible. En outre, l'algorithme détermine non seulement si un état donné est sûr ou non, mais il fournit également les contrôles viables qui maintiendront le système à l'intérieur du VK. De plus, il offre sous certaines hypothèses une preuve de convergence. Le VK du problème discrétisé converge vers le problème continu, à mesure que les étapes de discrétisation diminuent. Dans les environnements dynamiques, nous avons montré qu'il n'était pas possible en général d'assurer une sécurité de mouvement absolue. Ceci est dû au fait que nous devons lier la dimension temporelle et raisonner sur un horizon temporel fini. Néanmoins, nous avons pu obtenir des garanties de sécurité de mouvement, pour certaines classes particulières d'environnements dynamiques. Une autre caractéristique intéressante de l'algorithme est la possibilité de définir des états à éviter de toute nature autre que les états de collision. En revanche, le principal inconvénient de l'algorithme est sa complexité de calcul, il est exponentiel dans l'état et les espaces de contrôle. L'algorithme n'est peut-être pas assez rapide pour les paramètres en temps réel, mais il est toujours intéressant dans les applications où le VK est calculé hors ligne et est ensuite utilisé pour une navigation sûre par la suite.

On a opté pour la méthode $ICS^b$ parce qu'elle est une méthode sûre, sur laquelle tous les critères de sureté de mouvement sont vérifiés.

**Tableau 2.1 : Comparaison des méthodes réactives**

(x) : la caractéristique peut être vérifiée sous certaines conditions

| Méthode | Contraintes Cinématiques | Contraintes Dynamiques | Utilisation D'informations sensorielles | Environnement inconnu | Prise en compte des objets non-perçus | Environnement dynamique | Raisonnement Futur | Sureté |
|---|---|---|---|---|---|---|---|---|
| DW | Oui | Oui | Oui | Partiellement inconnu | Non | Oui | Non | Non |
| ICS Probabiliste | Oui | Oui | Oui | Oui | X | Oui | Oui | X |
| Une stratégie 3D utilisé sur les UAVs | Oui | Oui | Oui | Oui | Non | Oui | Oui | Non |
| ICS[b] | Oui | Oui | Oui | Oui | Oui | Oui | Oui | Oui |
| ICS[d] | Oui | Oui | Oui | Oui | Non | Oui | Oui | Oui |
| VK | Non | Non | Hors ligne | Non | Non | Oui | Oui | X |
| Méthode autonome pour les véhicules robotisés (Hector SLAM, HOG+FNN) | Oui | Oui | Oui | Oui | Non | Oui | Oui | Non |
| SURF | Oui | Non | Oui | Non | Non | Oui | Hors ligne | Non |

## 2.8 Conclusion :

Dans ce chapitre, nous avons présenté un état de l'art sur les méthodes existantes de la navigation réactives d'un robot mobile dans un environnement dynamique. Les méthodes réactives sont des méthodes d'évitement d'obstacles qui calculent le mouvement à appliquer au pas de temps suivant, mais elles manquent de convergence vers le but.

Le concept de sûreté est un niveau avancé et développé par rapport à l'évitement d'obstacles simple, car il prend en compte les états de collision inévitables et le comportement futur des obstacles dynamiques perçus et non- perçus. Les approches traitées ont été évaluées par rapport plusieurs critères.

Afin de profiter des avantages et éliminer les inconvénients que présentent ces méthodes, nous avons opté pour le concept ICS de freinage et l'adapter pour répondre aux besoins de notre travail. Le but de ce mémoire est basé sur le concept de sûreté pour la navigation autonome orientée. Cette notion est présentée dans les chapitres suivants.

# CHAPITRE 3

# LE CONCEPT ICS DE FREINAGE ET LA NAVIGATION REACTIVE SURE D'UN ROBOT MOBILE AUTONOME

## 3.1 Introduction :

Les roboticiens ont toujours essayé de concevoir un système de navigation autonome qui permet à un robot mobile de se déplacer en toute sécurité dans des environnements encombrés d'objets fixes et mobiles.

Dans ce chapitre, nous nous intéressons précisément à résoudre ce problème et développer un concept qui garantit la sûreté de mouvement où le robot ne va jamais se trouver dans une situation de collision avec un obstacle inattendu. Ce concept s'inspire principalement du concept général des ICS qui est nouveau dans la sûreté de mouvement, il est aussi montré que notre concept vérifie bien les trois critères généraux de la sûreté en utilisant un modèle du futur.

Pour vérifier si un état est un ICS de freinage ou non, un algorithme de vérification des ICS a été développé dans la première partie, Il permet d'implémenter le concept ICS de freinage sur n'importe quel système de navigation.

Par la suite une méthode de navigation sûre a été présentée, elle agit pendant un pas de temps donné pour générer un contrôle permettant de conduire le robot d'un état passivement sûr vers un autre. Le principe de l'approche ainsi que l'algorithme développé sont présentés dans la suite du chapitre.

## 3-2 Préliminaires:

Avant de présenter la méthode choisie, il faut définir certaines notions nécessaires pour la mieux comprendre.

### 3-2-1 L'espace de configuration :

Pour éviter les obstacles, aucun point du robot ne doit toucher ces derniers, ce qui rend la précision de la position de chaque point du corps du robot primordiale. D'où vient la notion de L'espace de configuration.

La configuration d'un système robotique est un ensemble de variables indépendantes qui déterminent de manière unique la position et l'orientation de chaque point du système [4,23]. Pour un robot mobile, c'est sa position (x, y) et son orientation $\theta$ (avec $\theta \in [-\pi, \pi]$).

L'espace de configuration, ou espace-C, d'un robot est l'espace de toutes les configurations possibles du système, en d'autres termes, c'est l'espace des positions possibles que le robot peut atteindre.

Dans l'espace de configuration, la notion de configurations interdites ou de collision, c'est-à-dire celles qui génère une collision, est bien connue. Les obstacles présents dans l'espace de travail font que certaines configurations sont interdites. Par exemple, si le robot occupant la position p entre en collision avec l'un des obstacles de l'espace de travail. Cette région est appelée obstacle-C (ou CB) [33]. Par opposition, l'espace libre ($C_{libre}$) est l'ensemble de toutes les configurations libres de collision dans C (pour lesquelles le robot n'intersecte aucun obstacle). $C_{libre}$ représente le complément de CB.

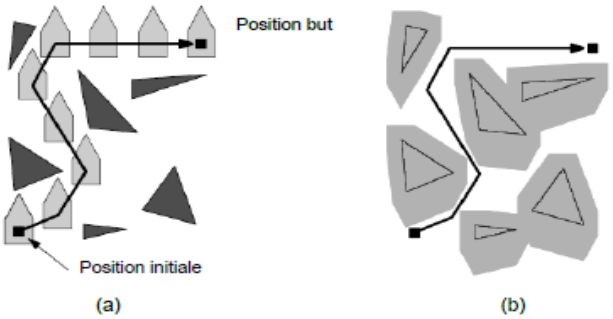



**Figure 3.1 : La translation d'un robot de forme polygonale dans un plan 2D encombré d'obstacles statiques. (a) L'espace de travail, le robot est représenté en gris clair et les obstacles sont représentés en gris foncé,**

**(b) l'espace de configuration**

## 3-2-2 L'espace-temps des configurations :

L'espace de configuration présenté dans le paragraphe précédent est dédié pour la résolution de problèmes de planification de mouvement dans des environnements statiques. Cependant, dans un environnement dynamique caractérisé par des objets mobiles dont la position évolue dans le temps, une simple planification de chemin n'est plus suffisante. Le chemin du robot doit être alors paramétré par le facteur temps, c'est pourquoi il est plus judicieux d'utiliser la notion de trajectoire. Ainsi, la notion de temps est introduite dans l'espace de configuration comme dimension supplémentaire. L'espace résultant est appelé espace-temps des configurations, et il est noté CBT.

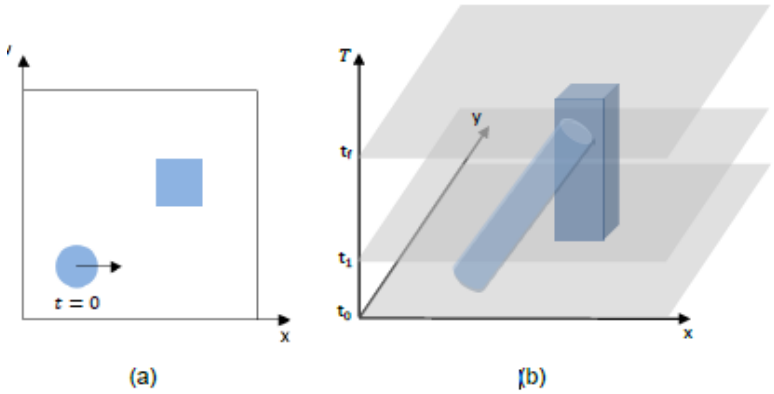



**Figure 3.2 : Un exemple illustratif (a) un espace de configuration avec un objet mobile (le disque avec le vecteur de vitesse) et un objet fixe (le carré), (b) l'espace-temps des configurations correspondant.**

### 3-2-3 L'espace d'états-temps :

L'espace d'état noté S, contrairement à l'espace temps des configurations, est plus approprié lorsqu'il s'agit de résoudre des problèmes de planification de trajectoire où la dynamique du système est prise en compte [23, 34, 35], c'est-à-dire, pour éviter les obstacles, le robot peut se déplacer plus rapidement ou freiner. C'est l'espace des paramètres de configuration et de leurs dérivées.

De même, l'espace des états-temps ST est approprié pour résoudre ces problèmes dans des environnements dynamiques, prenant comptes la dynamique du robot et des obstacles. ST est le résultat de la fusion des deux concepts espace-temps des configurations et espace d'états.

Les contraintes imposées par les obstacles mobiles et la dynamique du robot peuvent être représentées par des régions interdites statiques de l'espace d'états-temps.

### 3-3 Etats de collision inévitable (ICS) :

Toutes les approches réactives partagent le même principe: une ou plusieurs manœuvres d'évitement sont définies, et tout état pour lequel aucune de ces manœuvres d'évitement n'est sans collision est qualifié comme dangereux et est évité. Dans l'évitement d'obstacles en toute sécurité, le problème est de calculer des mouvements pour lesquels il est garanti que, quoi qu'il se passe au moment de l'exécution, le système robotique ne se trouve jamais dans une situation où il n'y a aucun moyen d'éviter la collision avec un obstacle inattendu.

Considérons la navigation, nous entendons essentiellement le problème de la détermination du mouvement élémentaire que le système robotique doit effectuer lors du prochain pas de temps. La principale préoccupation de la navigation est d'assurer la sécurité du système robotique. Dans un environnement comportant des obstacles mobiles, ce souci de sécurité est critique et il est important de prendre en compte à la fois la dynamique du système robotique et le comportement futur des obstacles mobiles.

En un sens, le concept d'état de collision inévitable englobe toutes ces approches. Il est général, il prend en compte à la fois la dynamique du système robotique et les obstacles (fixes ou mobiles, connus ou inconnus), et il considère toutes les manœuvres d'évitement possibles.

Le concept des ICS est à l'origine introduit pour assurer une sûreté de mouvement pour les systèmes robotiques (ugv, uav, navires…) dans des environnements dynamiques. Un ICS est un état pour lequel une collision se produira dans le futur quelque soit le comportement du système.

### 3-3-1 Critères de sureté de mouvement :

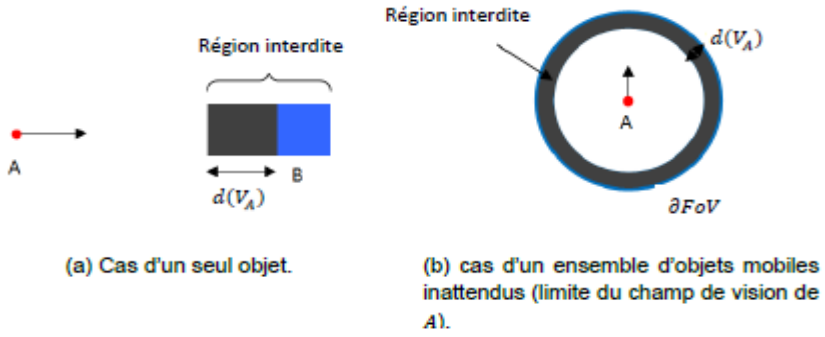
L'objectif de l'évitement d'obstacles est d'assurer la sécurité des systèmes robotiques. Cette dernière est garantie en respectant trois critères [4, 36] s'ils ne sont pas respectés, peuvent mettre le système robotique en danger et provoquer une collision à un moment donné dans le futur. Ces critères de sécurité sont respectivement liés au modèle du système robotique, au modèle de l'environnement et au processus décisionnel.

#### 3-3-1-1 Premier critère : contraintes liées à la dynamique du robot :

Un objet ponctuel B est à la position p le long de la trajectoire du robot A. A ne doit jamais décider d'occuper la position p car il serait en collision avec un obstacle B. En raison de sa dynamique, il faut à A un temps minimum pour ralentir et s'arrêter. La distance parcourue est (Figure 3.3.a):

$$d(va) = v^2 / 2a_{max}. \qquad (3.1)$$

Si A ne tient pas compte de ses propres caractéristiques dynamiques, il pourrait décider d'occuper une position dans la plage [p - d (vA), p [puisque ces positions sont sans collision. Si cela devait arriver, A serait en difficulté car il finirait par s'écraser sur B (quoi qu'il fasse dans le futur). Compte tenu de la dynamique de A, l'intervalle [p – d (vA), p] devient interdit. Cet exemple illustre le fait que, chaque fois qu'un système robotique ne tient pas compte de ses propres caractéristiques dynamiques, il peut décider d'un plan d'action futur pour lequel la sécurité n'est pas garantie et une collision peut éventuellement avoir lieu, d'où le premier critère de sécurité, Pour décider son futur mouvement, un robot doit tenir compte de sa propre dynamique.




(a) Cas d'un seul objet.

(b) cas d'un ensemble d'objets mobiles inattendus (limite du champ de vision de $A$).

**Figure 3.3 : Un exemple de l'influence de la dynamique d'un robot (une masse ponctuelle) sur sa sûreté.**

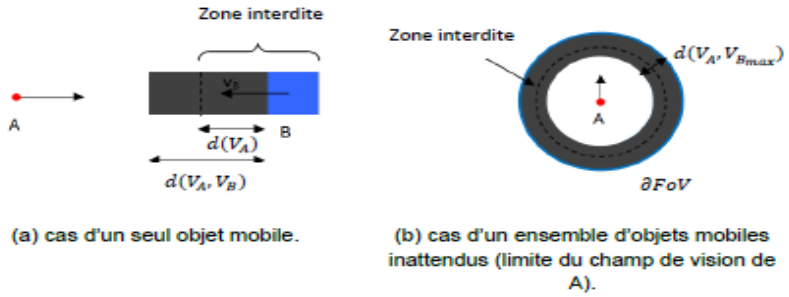
Dans un autre cas plus proche de nos hypothèses, si nous considérons que A à un champ de vision limité de forme circulaire ($\partial FoV$ ) et que chaque point de $\partial FoV$ est un obstacle potentiel (étant donné qu'aucune information n'est disponible sur la possibilité de présence ou non d'un obstacle sur $FoV$ ), alors la région interdite devient la bande autour de $\partial FoV$ (voir figure 3.3.b).

### 3-3-1-2 Deuxièmes critères : contraints liées à la dynamique des obstacles (raisonnement futur) :

Revenons au premier exemple mais en supposant maintenant que B se déplace vers la gauche avec une vitesse constante $VB \leq VB_{max}$. Dans cette situation, il suffit de rester en dehors de la plage de positions [p - d (vA), p] pour garantir la sécurité de A. En effet, à moins que A ne change en sens inverse jusqu'à ce qu'il atteigne une vitesse au moins égal à VB, une collision avec B se produira. La sûreté est alors garantie en vérifiant la condition suivante (figure 3.4.a):

$$d_{A/B} > d(V_A, V_B) \qquad (3.2)$$

$$\text{avec : } d(V_A, V_B) = V_A^2/2a_{max} + v_B\frac{VA}{2a\ max} \qquad (3.3)$$




(a) cas d'un seul objet mobile.

(b) cas d'un ensemble d'objets mobiles inattendus (limite du champ de vision de A).

**Figure 3.4 : L'influence de la dynamique d'un robot et celle des obstacles mobiles sur la sûreté de mouvement d'un robot**

Quand le robot à un champ de vision limité et que l'environnement est inconnu (comme dans notre cas), tous les objets inattendus doivent être pris en considération afin de garantir la sûreté de mouvement. Nous supposons donc que chaque point de la limite de visibilité $\partial FoV$ est considéré comme un obstacle potentiel. De plus, aucune information sur le comportement futur de ces obstacles n'est disponible, la seule information qui peut être considérée est leur vitesse maximale $v_{Bmax}$. La région interdite devient la bande le long de $\partial FoV$ qui correspond à :

$$d(V_A, V_{Bmax}) = V_A^2/2a_{max} + v_{Bmax}\frac{VA}{2a\ max} \qquad (3.4)$$

**3-3-1-3 Critère 3: un horizon temporel approprié :**

Les deux exemples précédents ont révélé le fait que la sureté ne consiste pas simplement à maintenir le système robotique à l'écart des états de collision. Il est fondamental d'éloigner le système robotique des états qui finiront par provoquer une collision à un moment donné dans le futur. À cette fin, un raisonnement sur l'avenir est nécessaire. D'où le troisième critère de sûreté de mouvement.

La question qui se pose est : jusqu'à quel point dans le futur le raisonnement (à savoir la modélisation) doit avoir lieu? La réponse se traduit par l'horizon temporel approprié. Si une connaissance a priori du comportement futur de l'environnement est disponible, le temps horizon est infini, mais quand la connaissance de l'environnement est limitée, le robot a un champ de vision limité, la sûreté doit être garantie sur un horizon temporel fini. Cela revient à assurer qu'aucune collision ne va avoir lieu pendant cette limite de temps, sans se préoccuper de ce qui peut se produire après cette limite.

Ces trois règles représentent en réalité des lois qui doivent être vérifiées par toute approche d'évitement d'obstacle ou de navigation dite sûre. Notons, que ces trois règles sont toutes liées au paramètre temps. Dans un environnement dynamique, ce paramètre est un élément primordial.

**3-3-1-4 Exemple d'illustration :**

Pour illustrer l'intérêt d'étendre ces notions de manière à prendre en compte la dynamique du système en introduisant le concept des états de collision inévitables. Considérons la figure 3.5, soit P une masse ponctuelle qui ne peut se déplacer vers la droite qu'avec une vitesse variable. Un état de P est caractérisé par sa position (x; y) et sa vitesse v. Si l'espace de travail W comporte un mur, les états dont la position correspond au mur sont évidemment des états de collision. Par contre, en supposant qu'il faut à P une certaine distance d(v) pour ralentir et s'arrêter, les états correspondant à la paroi et les états situés à une distance inférieure à d(v) à gauche de la paroi sont tels que, quand P est dans un tel état, quoi qu'il fasse dans le futur, une collision se produira.

Pour la propre sécurité de P, lorsqu'il se déplace à la vitesse v, il ne devrait jamais être dans l'un de ces états de collision inévitables. La taille de la région des états de collision inévitables, c'est-à-dire la région grise située à gauche du mur, dépend de la distance d (v) qui dépend à son tour de la vitesse actuelle de P.

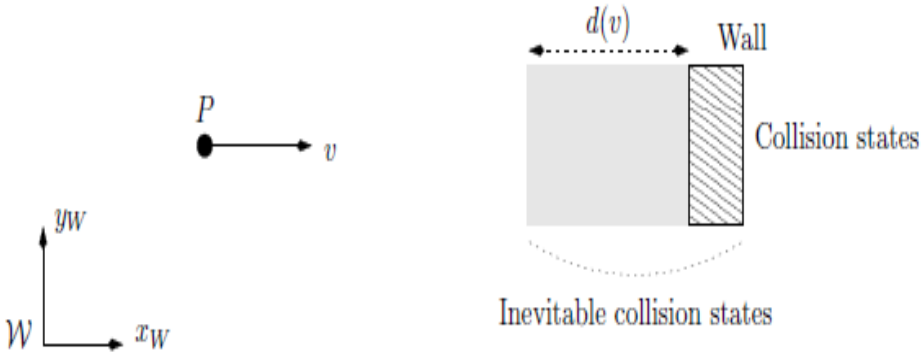



**Figure 3.5 : États de collision inévitables**

## 3-3-2 Modélisation du futur :

Le paragraphe précédent a montré la nécessité de modéliser l'évolution future de l'environnement jusqu'à une anticipation appropriée. Construire un tel modèle spatio-temporel est en soi un défi dans la plupart des situations du monde réel car des informations complètes sur l'environnement et son évolution future ne sont pas disponibles [4, 37]. Le but de cette section est de présenter les principales approches proposées pour relever ce défi et de répondre à la question : quel modèle du futur doit être utilisé dans le cas étudié dans ce mémoire ?

### 3-3-2-1 Modèles déterministes :

Les premières approches adhéreraient à un principe que l'on pourrait appeler le monde gelé: chaque obstacle dans l'environnement du robot est traité comme un obstacle fixe et supposé rester en place dans le futur, par exemple [38]. Plus tard, avec les progrès de la perception dans le domaine de la détection et du suivi des obstacles en mouvement, des modèles du futur basés sur l'extrapolation du comportement futur des obstacles en mouvement à partir de leur état actuel sont apparus. Dans la plupart des cas, l'extrapolation reposait sur une hypothèse de comportement constant, par exemple une vitesse constante [39] ou des techniques de régression [40].

Dans quelques cas, des techniques sophistiquées de prédiction de mouvement à long terme ont été proposées: elles exploiteraient la structure de l'environnement à portée de main ou apprendraient comment les obstacles se déplacent dans un environnement donné.

Dans tous les cas, le trait commun de ces approches est que chaque obstacle se voit attribuer un mouvement futur nominal (connu a priori, estimé ou appris). Les décisions de déplacement sont basées sur ces mouvements futurs nominaux.

### 3-3-2-2 Modèles conservateurs :

Du point de vue de la sécurité des mouvements, les modèles déterministes sont utiles tant que leur prédiction de l'évolution future de l'environnement est fiable. Malheureusement, cette fiabilité peut diminuer considérablement à long terme. Pour résoudre ce problème, des modèles conservateurs de l'évolution future de l'environnement ont été proposés. L'idée est d'envisager tous les mouvements futurs possibles des obstacles de l'environnement. En conséquence, chaque obstacle se voit attribuer son ensemble atteignable, c'est-à-dire l'ensemble de tous les états qu'il peut atteindre dans le futur, pour représenter son mouvement futur [4, 41,42].

Dans ce type de représentation, le comportement du futur est estimé en utilisant les limites des contraintes dynamiques des obstacles mobiles (ex. vitesse maximale) ou des limites géométriques telles qu'une restriction de l'orientation à un intervalle de valeurs. Cet intervalle peut être lié à la nature de l'environnement ou la tâche à accomplir (par exemple quand le robot se déplace dans une route l'orientation des obstacles va correspondre aux directions de la route). Ces contraintes sont modélisées de manière à ce que le modèle du futur englobe tous les mouvements futurs possibles des obstacles.

### 3-3-2-3 Modèles probabilistes :

Les modèles conservateurs sont satisfaisants du point de vue de la sécurité des mouvements puisqu'ils peuvent garantir l'évitement des collisions. Cependant, en raison de la croissance rapide des ensembles accessibles d'obstacles, tout l'espace d'état du robot est finalement interdit et le robot est bloqué. Pour résoudre ce problème, des modèles probabilistes du futur ont été proposés. Dans de tels modèles, le mouvement futur de chaque obstacle est caractérisé par une fonction de densité de probabilité. Les outils utilisés pour prédire le comportement des obstacles mobiles sont très divers, par exemple, les chaînes de Markov, les modèles de Markov cachés et la simulation de Monte Carlo. Pour aborder la sécurité des mouvements avec des modèles probabilistes du futur, Fraichard a proposé une extension probabiliste du concept ICS [25], les modèles probabilistes sont meilleurs pour capturer l'incertitude qui prévaut dans le monde réel, en particulier l'incertitude concernant le comportement futur de la planification du mouvement itératif et du problème de sécurité des obstacles en mouvement. Cependant, ils ne permettent pas une garantie stricte de sécurité de mouvement, ils permettent plutôt de minimiser le risque.

### 3-3-2-4 L'approche utilisée :

Soit A le robot mobile considéré dans ce mémoire, qui s'exécute dans la zone de travail 2D (WS). Sachant que A est équipé d'un capteur de distance (un télémètre laser), il ne

peut percevoir qu'un sous-ensemble de WS. Ce sous-ensemble est le champ de vision de A, il est noté FoV, sa forme est arbitraire et dépend de l'environnement actuel de A (voir Figure 3.5.a). WS est alors partitionné en trois sous-ensembles: (1) FoV, (2) $FoV^C$ et (3)$\partial$FoV, la limite entre les deux. Il parait raisonnable de supposer que A "explore autour de lui"; autrement dit que A se trouve toujours à l'intérieur de FoV. Ce dernier peut contenir des "trous" représentant des objets entièrement perçus par le système sensoriel de (voir figure 3.5.b).FoV représente la région de WS qui est libre d'objet à l'instant de détection.

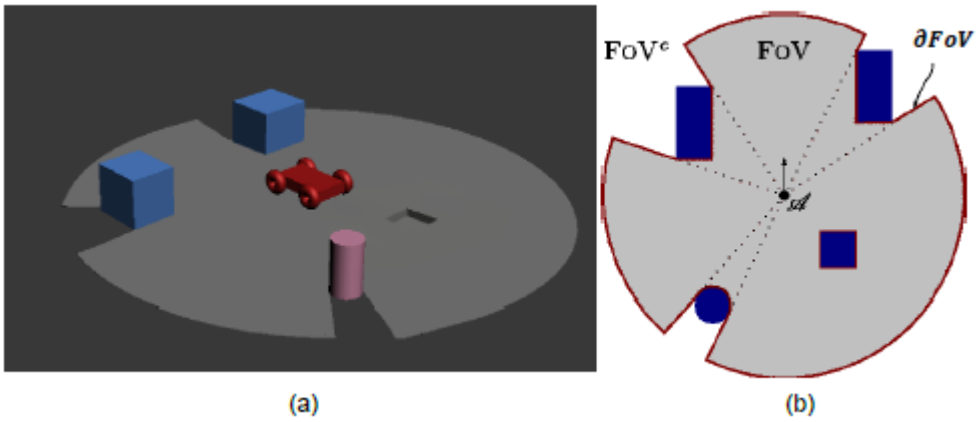



**Figure 3.6 : Un robot avec un champ de vision limité dans un environnement inconnu (a) un scénario avec deux obstacles fixes, un obstacle mobile et un trou (la région en gris foncé est non perçue), (b) le champ de vision correspondant, où $FoV$ est la région perçue (en gris), $\partial FoV$ sa limite et $FoV^c$ est la région non perçue.**

L'environnement de A peut alors contenir aussi bien des objets perçus que des objets non perçus dont le comportement est imprévisible. C'est pourquoi parmi les trois modèles proposés dans les sous-paragraphes précédents, le plus adapté à gérer ce type de comportement afin de garantir la sûreté de mouvement est le second type de représentation ; à savoir le modèle conservatif : tous les mouvements futurs possibles pour un objet donné doivent être pris en considération.

## 3-4 Concept des états de collision inévitables de freinage($ICS^b$) :

Le concept $ICS^b$ est introduit pour traiter le problème de la sureté passive de mouvement ou la navigation des robots est autonome avec des capteurs ayant un champ de vision limité dans des environnements inconnus comportant des objets en mouvement dont le comportement futur est inconnu.

La dynamique du robot A est généralement décrite par des équations différentielles de la forme:

$$\dot{s} = f(s, u) \text{ sujet à } g(s, \dot{s}) \leq 0 \qquad (3.5)$$

où s ∈ S est l'état de A, ṡ sa dérivée temporelle et u ∈ U un contrôle. S et U désignent respectivement l'espace d'états et l'espace de contrôle de A. Soit A (s) le sous-ensemble fermé de l'espace de travail W occupé par A lorsqu'il est dans s. Soit ŭ:[0, $t_f$] → U une trajectoire de contrôle, c'est-à-dire une séquence temporelle de contrôles, $t_f$ est la durée de ŭ. L'ensemble de toutes les trajectoires de contrôle possibles est noté Ŭ. A partir d'un état initial $s_0$ au temps 0, une trajectoire d'état š, c'est-à-dire une séquence temporelle d'états, est dérivée d'une trajectoire de commande ŭ en intégrant (2.5); š ($s_0$, ŭ, t) désigne l'état atteint au temps t. Une trajectoire de commande $ŭ_b$∈ Ŭ telle que $š_b$ ($s_0$, $ũ_b$, $t_b$) est un état où A s'arrête (repos) est une trajectoire de freinage pour $s_0$ et $t_b$ est son temps de freinage. Soit Bi le modèle espace × temps de l'évolution future de l'obstacle, Bi(t) désigne le sous-ensemble de W occupé par Bi à un instant particulier t dans le modèle conservatif de futur.

### 3-4-1 Définition des $ICS^b$ :

Un ICS de freinage ($ICS^b$) est défini de manière informelle comme un état pour lequel quelle que soit la future trajectoire de freinage suivie par A, une collision se produit avant que A ne soit au repos. D'où la définition formelle suivante:

$ICS^b$: s est un $ICS^b$ ssi ∀$ŭ_b$∈$U_{bs}$, ∃t∈[0,$t_b$[, š(s,$ŭ_b$,t)est un état de collision au temps t[4].

A partir de cette définition, on constate que c'est un concept qui vérifie les critères de la sureté de mouvement. La région non p-sûre est définie à partir de A (š(s,$ŭ_b$,t)) qui représente le sous-ensemble fermé de l'espace de travail WS occupé par le système A quand il est à l'état atteint à l'instant t en démarrant de s et en suivant la trajectoire $ũ_b$, ce qui vérifie le 1er critère. $B_i$ (t) définit le sous-ensemble de WS occupé par $B_i$ le modèle conservatif de futur à un instant t de ce modèle, cela vérifie le 2m critère. Le dernier critère revient à utiliser un horizon temporel approprié. Le principe est déjà ancré dans le concept même de $ICS^b$, qui considère des trajectoires de durée finie.

Pour que la sureté passive se réalise, A doit toujours être dans un état passivement sur, si une collision aura lieu le robot sera à l'arrêt.

L'ensemble des $ICS^b$ donnant une collision avec un objet particulier $B_i$ :

$$ICS^b(B_i) = \{s \in S | \forall \tilde{u}_b \in \tilde{U}_b^s, \exists t \in [0, t_b[, A(\tilde{s}(s, \tilde{u}_b, t)) \cap B_i(t) \neq \emptyset\} \quad (3.6)$$

L'ensemble des $ICS^b$ donnant une collision avec l'ensemble des obstacles B:

$$ICS^b(B) = \{s \in S | \forall \tilde{u}_b \in \tilde{U}_b^s, \exists t \in [0, t_b[, A(\tilde{s}(s, \tilde{u}_b, t)) \cap B(t) \neq \emptyset\} \quad (3.7)$$

L'ensemble $ICS^b$ donnant une collision avec $B_i$ une trajectoire donnée $ŭ_b$ est le suivant :

$$ICS^b(B_i, \tilde{u}_b) = \{s \in S | \exists t \in [0, t_b[, A(\tilde{s}(s, \tilde{u}_b, t)) \cap B_i(t) \neq \emptyset\} \quad (3.8)$$

L'ensemble $ICS^b$ donnant une collision avec l'ensemble des obstacles B pour une trajectoire donnée $\tilde{u}_b$ est :

$$ICS^b(B, \tilde{u}_b) = \{s \in S | \exists t \in [0, t_b[, A(\tilde{s}(s, \tilde{u}_b, t)) \cap B(t) \neq \emptyset\} \quad (3.9)$$

Enfin, l'ensemble $ICS^b$ donnant une collision avec $B_i$ pour une trajectoire donnée $\tilde{u}_b$ à un instant t est :

$$ICS^b(B_i, \tilde{u}_b, t) = \{s \in S | A(\tilde{s}(s, \tilde{u}_b, t)) \cap B_i(t) \neq \emptyset\} \quad (3.10)$$

**3-4-2 Les propriétés des $ICS^b$:**

Propriété 1 (Intersection des entrées de contrôle)

$$ICS^b(B) = \bigcap_{\tilde{u}_b \in \tilde{U}_b} ICS^b(B, \tilde{u}_b) \quad (3.11)$$

Propriété 2 (Union des obstacles) :

$$ICS^b\left(\bigcup_{i=1}^{n} B_i, \tilde{u}_b\right) = \bigcup_{i=1}^{n} ICS^b(B_i, \tilde{u}_b) \quad (3.12)$$

Propriété 3 (caractérisation de $ICS^b$) :

$$ICS^b(B) = \bigcap_{\tilde{u}_b \in \tilde{U}_b} \bigcup_{i=1}^{n} ICS^b(B_i, \tilde{u}_b) \quad (3.13)$$

Propriété 4 (approximation de $ICS^b$) :

$$ICS^b(B) \subseteq ICS^b(B, \varepsilon) \quad (3.14)$$

Propriété 5 (horizon temporel de $ICS^b$):

$$ICS^b(B, \varepsilon) = ICS^b(B([0, T_h[), \varepsilon) \quad (3.15)$$

Sachant que :

$$T_h = \max_{\tilde{u}_b \in \varepsilon}\{t_b\} \quad (3.16)$$

Propriété 6 (limites du champ de vision) :

$$ICS^b(B) = ICS^b(\partial FoV \cup FoV^c) = ICS^b(\partial FoV) \tag{3.17}$$

### 3-4-3 Exemple d'application :

Pour illustrer les définitions et les attributs introduits précédemment, un exemple simple est utilisé. Il montre comment calculer $ICS^b$ pour différents types d'objets (objets statiques ou mobiles) avec des formes ponctuelles (le choix de la forme sera expliqué dans le chapitre suivant).

Notons, cet exemple traite principalement de l'aspect représentation schématique de l'$ICS^b$ sans se soucier de la précision des calculs ou de la précision de la représentation, car le but est de comprendre les principes de base.

Le système A considéré est une masse ponctuelle qui se déplace uniquement selon deux trajectoires de freinage possibles ($\check{u}_1$ et $\tilde{u}_2$). A noter qu'en raison de la caractéristique 4, seules deux trajectoires peuvent être considérées dans le calcul au lieu de la trajectoire de freinage infinie possible [4].

#### 3-4-3-1 Point statique :

En supposant que B est un point statique, $ICS^b$ est calculé par ICSb (B, $u_b$) (représenté par la ligne gris clair sur la figure 3.7) et $ICS^b$ (B, u) (représenté par l'arc gris foncé sur la figure 3.7) en fonction de la propriété 1. L'ensemble $ICS^b$ correspond à l'état de l'objet B. Par conséquent, à moins que A ne soit déjà entré en collision avec B, il peut toujours éviter la collision [4].

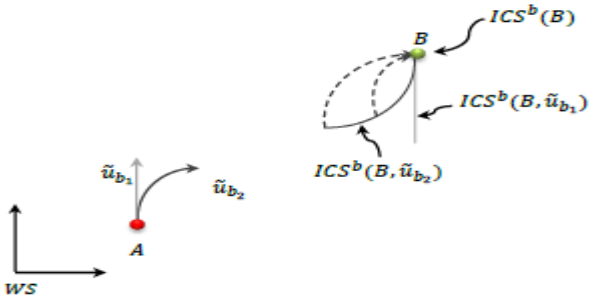



**Figure 3.7. Calcul de $ICS^b$ pour un point statique**

#### 3-4-3-2 point dynamique :

Dans cet exemple, on suppose que l'obstacle est un point en mouvement avec une vitesse linéaire constante. Ensuite, $ICS^b$ dépend de la vitesse relative entre A et B [1], comme le montre la figure 3.8.

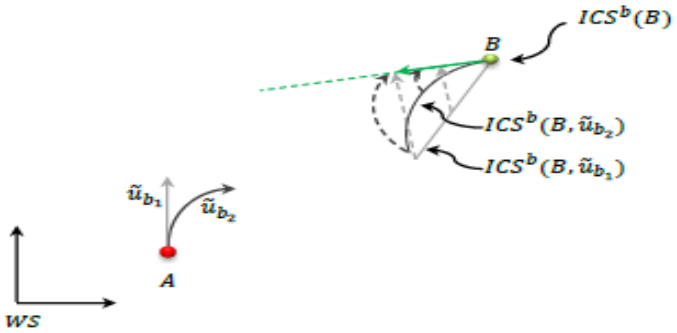



**Figure 3.8. Calcul de $ICS^b$ pour un point mobile.**

## 3-5 L'algorithme de vérification :

Pour vérifier si un état est un $ICS^b$ ou non, autrement dit s'il est sur ou pas, l'algorithme est fait en utilisant le mode de l'évolution future de l'environnement et en se basant sur les propriétés de $ICS^b$.

L'algorithme développé ici a deux contributions, une pour les obstacles fixe (environnement statique) et l'autre pour les obstacles mobiles (environnement dynamique).

Algorithme 1 : vérification des états de collision inévitable

1. **Entrées** : S0 : l'état actuel ; S1 : l'état prochain ; dl : les longueurs des u ; B : les positions des obstacles.
2. **Sorties** : valeur booléenne.
3. **pour tous** U;
4. **faire**
5. **pour tous** B ;
6. **faire**
7. Calcul de $ICS^b$ pour chaque obstacle ; //ajouter le rayon df
8. Calcul de l'intersection entre u et un obstacle B ;
9. **Si** u ∩ B existe **alors**
10. calcul des positions des points d'intersection
11. calcul de d ; // distances entre la position actuelle et les points d'intersection ;

**12.** **si** d>dl **alors** ; //comparaison entre les distances calculées et les longueurs des u ;

**13.** ICS faux ;

**14.** **sinon**

**15.** ICS vrai ;

**16.** **fin si**

**17.** **fin si**

**18**. **fin faire**

**19**. **fin faire**

L'algorithme 1 est un algorithme de vérification des états prochains s'ils sont des ICS ou non. Nous avons développé une méthode qui permet de juger l'état à vérifier s'il appartient à la région interdite ou pas. Cet algorithme a comme entrées l'état actuel, l'état prochain, la longueur de chaque trajectoire générée et la position des obstacles.

Les obstacles sont considérés comme des disques et une distance de sécurité est ajoutée qui représente la zone des états de collision inévitables pour chaque objet (ligne #7).

Après avoir calculé la longueur de chaque trajectoire, il faut vérifier si cette trajectoire s'intersecte avec l'un des obstacles (ligne #8). S'il ya une intersection avec l'un des obstacles, on calcule la distance entre l'état actuel et les points d'intersection (ligne #11). Si cette distance est inferieure à la longueur de la trajectoire (ligne #12) alors cet état est un ICS, donc il n'est pas sûr. Pour que $s_1$ soit un état sûr, il faut que la trajectoire qui mène à cet état n'intersecte pas avec tous les obstacles de l'environnement. Où bien la distance calculée entre l'état actuel et les points d'intersection soit supérieure par rapport la longueur de trajectoire.

A la fin, une valeur booléenne vrai ou faux (lignes #13, #15) est retournée pour dire si l'état $s_1$ est un ICS ou non.

## 3-6 Navigation réactive passivement sûre :

Afin de démontrer la sécurité passive des mouvements et de valider le concept d'$ICS^b$, une méthode de navigation réactive orientée a été développée pour un robot mobile ayant un champ de vision limité dans un environnement dynamique inconnu appelée PASSAVOID. C'est un système d'évitement d'obstacles, permet de montrer l'efficacité du concept ICS de freinage quant à la garantie d'une sûreté de mouvement passive. C'est un système réactif qui agit pendant un pas de temps donné pour générer un contrôle permettant de conduire le robot d'un état passivement sûr vers un autre. Il permet de sélectionner une trajectoire passivement sûre jusqu'au but où le concept ICS de freinage est utilisé. Sa tâche primaire est de garder le robot dans des états sûrs ou de manière équivalente, de le conduire loin des états ICS. Cette méthode garantit la sûreté passive du mouvement peu importe ce qui se passe dans l'environnement. Autrement dit, si une

collision a lieu, il est garanti que le robot sera au repos quand ça se produira. Le principe de l'approche ainsi que l'algorithme développé sont présentés dans ce chapitre.

### 3-6-1 Principe de cette approche :

Cette approche est une méthode de navigation réactive. A chaque pas de temps, l'objectif est de calculer un contrôle constant qui sera appliqué au système robotique au prochain pas de temps; il doit être Autorisé, c'est-à-dire que la trajectoire d'état correspondante doit être sûre.

Ce concept s'appuie sur l'algorithme de vérification des ICS pour fonctionner. C'est un schéma de navigation réactive qui fonctionne avec un pas de temps donné. Son but est de calculer la commande constante u qui sera appliquée au robot lors du prochain pas de temps; u doit être admissible, c'est-à-dire que la trajectoire d'état correspondante doit être sans ICS) [27].

La méthode développée fonctionne comme la plupart des travaux d'évitement de collision réactifs standard. Dans tous les cas, leur principe de fonctionnement est de caractériser d'abord les régions interdites dans un espace de contrôle donné puis de sélectionner un champ admissible, c'est-à-dire qui n'est pas interdit. Par conséquent, l'évitement de collision dépend également de la capacité du système d'évitement de collision à portée de main à trouver une telle commande admissible. La navigation réactive sûre a également recours à l'échantillonnage pour trouver un contrôle admissible. Cependant, contrairement aux schémas classiques d'évitement de collision, elle est conçue de telle manière qu'il est garanti que, si un contrôle admissible existe, il fera partie de l'ensemble d'échantillonnage [27].

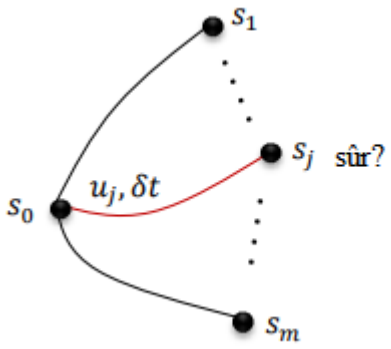



**Figure 3.9 : Principe de fonctionnement**

Le principe de fonctionnement est illustré sur la figure ci-dessus. Soit $s_0$ l'état actuel de robot et U un ensemble échantillonné de commandes: U = {$u_1$... $u_m$}. Un contrôle donné $u_j \in$ U est appliqué pendant une durée t. Il fait passer le robot de l'état $s_0$ à l'état $s_j$. Si la trajectoire d'état entre $s_0$ et $s_j$ est sûre c'est-à-dire elle est sans collision, alors $u_j$ est admissible [1].

Un contrôle admissible c'est un contrôle qui permet au robot de passer d'un état sûr à un autre. Un état sûr est l'état qui s'agit pas d'un ICS, et c'est l'état pour le quel le robot n'aura jamais face à une collision avec un obstacle inattendu peu importe qui va se passer dans l'environnement.

**3-7 Garantie de la sûreté passive du mouvement :**

Une telle méthode fonctionne bien tant qu'une commande admissible peut être trouvée dans U à chaque pas de temps. Mais si, en fin de compte, U est vide, cela signifie que chaque commande dans U amène A à un ICS. En d'autres termes, la sureté passive ne sera pas obtenue et une collision aura lieu alors que A est toujours en mouvement. Pour résoudre ce problème, il est nécessaire de garantir que U contient au moins un contrôle admissible.

Si l'état $s_0$ est sûr alors il existe au moins un contrôle admissible u qui peut être utilisé pour conduire A à un état qui est également sûr.

La conception de cette approche dont la sécurité de mouvement passive est garantie doit simplement conduire le robot d'un état sûr au suivant. Maintenant, en supposant que $s_0$ est sûr. Il est garanti qu'il existe au moins un contrôle admissible u qui, s'il est appliqué au robot pendant un pas de temps, le mènera à un autre état sûr.

**3-8 L'Algorithme générale de PASSAVOID :**

L'algorithme développé pour cette approche comporte les étapes suivantes

à chaque pas de temps, il calcule les contrôles admissibles en vérifiant si l'état s est sûr et permet au robot de se déplacer d'une manière sûre dans les états non interdits en choisissant l'état prochain sûr le plus proche de point but. Si les états prochains sont tous des ICS le robot exécute la manœuvre de freinage où il va appliquer un contrôle nul.

Algorithme 2 **:** PASSAVOID

1. **Entrées :** $s_0$ :l'état actuel ; $\delta t$: pas de temps ; w(t) : modèle de l'environnement.
2. **Sortie :** u.
3. modélisation de l'environnement
4. Générer U= (V, Ω) ; //génération des trajectoires de contrôles .
5. **Pour tous** U **;** // pour chaque trajectoire générée.
6. **faire**
7. Calculer $s_1$ (x1, y1, $\varphi$1) ; // la position de l'état prochain.

**8.** Calculer dl // la longueur de chaque trajectoire

**9.** **Pour tous B** ; // pour chaque obstacle.

**10.** **Faire**

**11.** calculer ICS // calculer les états de collision inévitables.

**12.** **fin faire**

**13.** **Si** $s_1 \notin$ ICS **alors** // si l'état $s_1$ est sûr

**14.** calculer dmin **;** // calculer la distance minimale vers le point but.

**15.** u=umin ; // le contrôle qui correspond à la distance minimale.

**16.** **finsi .**

**17.** **Si tous** $s_1 \in$ ICS **alors** // si tous les états prochains ne sont pas sûr.

**18.** u=(0,0) ; // contrôle nul.

**19.** **Fin si**

**20.** **fin faire**

**21.** Sélectionner u // sélectionner le contrôle pour aller à l'état prochain

**22.** $s_0 \leftarrow s_1$ ; // l'état prochain devient l'état actuel pour le pas de temps suivant pour refaire le calcul

L'Algorithme 2 est un algorithme général de la navigation réactive qui garantit la sûreté passive où les étapes requises pour vérifier si l'état prochain est passivement sûr ou non sont données.

En plus de l'état actuel, cet algorithme a comme entrées $\delta t$ ; le pas de temps et w(t) ; le modèle de l'environnement qui donne l'emplacement des obstacles.

D'abord, il faut modéliser l'environnement (ligne #3 algorithme 2) en considérant les obstacles perçus et non perçus en tenant compte de la limite de champ de vision comme obstacles mobiles inattendus. Ces derniers sont modélisés en disque qui a le rayon du robot plus la distance de sécurité, la collision se produira quoi que le robot fasse s'il entre dans cette zone parce que le temps de collision sera inferieur au temps de freinage, le centre de chaque disque représente le point de limite des ondes laser émises chaque les 0.25° dans le demi plan en face le robot.

Ensuite, l'espace de contrôle doit être échantillonné en générant des trajectoires de contrôle, chaque trajectoire doit avoir une vitesse linéaire et angulaire (ligne #4 algorithme 2). Pour chaque trajectoire donnée, un calcul de l'état prochain est effectué en se basant sur les équations odométriques (ligne #7 algorithme 2) ainsi que le calcul de la longueur de la trajectoire qui mène à cet état (ligne #8 algorithme 2). Une fois l'état prochain est déterminé, il faut vérifier s'il est passivement sûr ou non en calculant les états de collision inévitable (ligne #11 algorithme 2), c'est le but de l'algorithme 1. Ensuite, déterminer si cette trajectoire est un candidat d'être admissible ou non.

Si l'état à vérifié est passivement sûr (n'appartient pas à la région ICS), un calcul de distance par rapport au point but est effectué et l'état le plus proche de point but est déterminé (ligne #14 algorithme 2). Le contrôle choisi est celui qui correspond à cette distance minimale (ligne #15 algorithme 2).

Dans le cas où aucun des états prochains est passivement sûr (autrement dit, si les états prochains sont tous des ICS), le robot doit exécuter une manœuvre de freinage et un contrôle nul est appliqué (ligne #18 algorithme 2).

Une fois ce calcul est terminé, le système prend la bonne décision et applique le contrôle qui convient pour se déplacer à l'état prochain ou se freiner (ligne #21 algorithme2). L'état prochain va devenir l'état actuel (ligne #22 algorithme 2) et ce processus va se répéter à chaque pas de temps.

Ce processus est testé en vérifiant la sûreté en deux cas, le premier est simple en présence des obstacles fixes seulement, le deuxième est testé pour un environnement dynamique ou il y'a des obstacles fixes et mobiles.

Dans les deux cas on a utilisé des informations réelles des capteurs (télémètre laser et odomètre) seulement, ces informations étaient enregistrées dans un fichier bag qui nous a permet de vérifier l'algorithme sans avoir l'implémenté sur la chaise. Le champ de vision est réduit de 270° à 180°, disant que le danger est présent qu'en face le robot. Les résultats sont détaillés dans le prochain chapitre.

Ce concept est une sécurité passive prouvée dans le sens où il est garanti que le robot restera toujours à l'écart de l'ICS de freinage, peu importe ce qui se passe dans l'environnement. Avec cette méthode, il est assuré que le robot va prendre la bonne décision sans doute.

## 3-9 Conclusion :

Dans ce chapitre, nous avons montré comment un système de navigation déplace d'une manière autonome et sûre en évitant les obstacles fixes, mobiles et inaperçus dans un environnement inconnu. Cela a été réalisé grâce au concept ICS de freinage où un

algorithme qui permet de définir si un état est $ICS^b$ ou non. Pour valider ce concept, nous avons présenté une méthode de navigation sûre orientée vers un but qui permet à un robot mobile ayant un champ de vision limité de se déplacer d'un état sûr à un autre en garantissant la sûreté passive.

# CHAPITRE 4

# IMPLEMENTATION ET RESULTATS

## 4-1 Introduction :

L'objectif de ce travail est de résoudre un problème critique dans la navigation des robots, à savoir la garantie de la sûreté de mouvement. Ce problème a été largement étudié dans les chapitres précédents où deux algorithmes étaient développés qui permettent à un robot mobile de se déplacer dans un milieu inconnu et dynamique en tenant compte des obstacles fixes et mobiles ainsi que les objets inattendus pour rejoindre son point de but.

Afin de montrer l'efficacité de cette approche et sa performance, ces algorithmes sont testés en C++ sous ROS en utilisant des informations réelles en deux parties. La première consiste à tester le fonctionnement du programme dans un environnement qui contient des obstacles fixes. La deuxième phase est consacrée pour les obstacles dynamiques.

Le modèle du robot utilisé correspond à la plateforme mobile de la chaise roulante robotisée FAURSA disponible au sein de CDTA (voir chapitre 1).

Dans ce présent chapitre nous allons exposer l'implémentation et les résultats obtenus dans l'implémentation après avoir présenté les différentes étapes de l'implémentation.

## 4-2 Système d'exploitation des robots « ROS » :

Comme son nom l'indique, ROS (Robot Operating System) est un système d'exploitation pour robots. De même que les systèmes d'exploitation pour PC, serveurs ou appareils autonomes, ROS est un système d'exploitation complet pour la robotique de service.

Avant les OS robotiques, chaque concepteur de robot, chaque chercheur en robotique consacrait un temps non négligeable à concevoir matériellement son robot ainsi que le logiciel embarqué associé. L'idée principale d'un OS robotique est d'éviter de réinventer la roue à chaque fois et de proposer des fonctionnalités standardisées faisant abstraction du matériel, tout comme un OS classique pour PC, d'où l'analogie de nom.

ROS est développé et maintenu par une société californienne, Willow Garage, fondée en 2006 par Scott Hassan, un des premiers employés de Google qui a participé au

développement de la technologie du moteur de recherche et qui est aussi à l'origine des Yahoo Groups. Le PDG de Willow Garage est Steeve Cousin.

La philosophie de ROS se résume dans les cinq grands principes suivants :

- Peer to Peer : permet à chacune des composantes d'un robot de dialoguer en direct avec une autre composante, de manière synchrone ou asynchrone en fonction des besoins.

- Basée sur des outils : adoption d'un design qui utilise un grand nombre de petits outils pour faire la compilation et l'exécution des différents composants ROS.

- Multi langages : ROS est neutre d'un point de vue langage et peut être programmé en différents langages.

- Léger : facile d'usage.

- Gratuit et open source.

**4-2-1 Les notions de base de ROS :**

Le principe de base d'un OS robotique est de faire fonctionner en parallèle un grand nombre d'exécutables qui doivent pouvoir échanger de l'information de manière synchrone ou asynchrone. Par exemple, un OS robotique doit interroger à une fréquence définie les capteurs du robot (capteur de distance à ultrasons ou infrarouge, capteur de pression, capteur de température, gyroscope, accéléromètre, caméras, microphones…), récupérer ces informations, les traiter (faire ce que l'on appelle la fusion de données), les passer à des algorithmes de traitement (traitement de la parole, vision artificielle, localisation et cartographie simultanée…) et enfin contrôler les moteurs en retour. Tout ce processus s'effectue en continu et en parallèle. D'autre part, l'OS robotique doit assurer la gestion de la concurrence afin d'assurer l'accès efficace aux ressources du robot.

ROS répond à tout cette problématique grâce à des notions de base simples qui sont :

-Le package : c'est l'unité principale d'organisation logicielle de ROS. Un package est un répertoire qui contient les nœuds, les bibliothèques externes, des données, des fichiers de configuration et un fichier de configuration xml.

-Les nœuds : c'est une instance d'un exécutable. Un nœud peut correspondre à un capteur, un moteur, un algorithme de traitement, de surveillance. Chaque nœud qui se lance se déclare au Master.

-Le Master : est un service de déclaration et d'enregistrement des nœuds qui permet ainsi à des nœuds de se connaître et d'échanger de l'information.

-Les topics: L'échange de l'information s'effectue soit de manière asynchrone via un topic ou de manière synchrone via un service. Un topic est un système de transport de

l'information basé sur le système de l'abonnement / publication (subscribe / publish). Un ou plusieurs nœuds pourront publier de l'information sur un topic et un ou plusieurs nœuds pourront lire l'information sur ce topic. Le topic est en quelque sorte un bus d'information asynchrone. Le topic est typé, c'est-à-dire que le type d'information qui est publiée (le message) est toujours structuré de la même manière. Les nœuds envoient ou reçoivent des messages sur des topics.

-Publisher et subscriber : Le principal mécanisme permettant aux nœuds ROS d'échanger des données est l'envoi et la réception de messages. Les messages sont transmis sur un thème, et chaque thème a un nom unique dans le réseau ROS. Si un nœud souhaite partager des informations, il utilise un Publisher pour envoyer des données à un topic. Un nœud qui souhaite recevoir ces informations utilise un subscriber à ce même topic. Les publishers et les subscribers sont découplés par des topics et peuvent être créés et détruits dans n'importe quel ordre. Un message peut être publié dans un topic même s'il n'y a pas de subscribers actifs.

-Les messages : Un message est une structure de donnée composite. Un message est composé d'une combinaison de types primitifs (chaînes de caractères, booléens, entiers, flottants…). Par exemple un nœud représentant un servomoteur du robot, publiera certainement son état sur un topic (selon ce que vous aurez programmé) avec un message contenant par exemple un entier représentant la position du moteur, un flottant représentant sa température, un autre flottant représentant sa vitesse. La description des messages est stockée dans nom_package/msg/monMessageType.msg. Ce fichier décrit la structure des messages.

-Les services : Le topic est un mode de communication asynchrone. Le service en revanche répond à une autre nécessité, celle d'une communication synchrone entre deux nœuds. Cette notion se rapproche de la notion d'appel de procédure distante (remote procedure call)

La description des services est stockée dans nom_package/srv/monServiceType.srv. Ce fichier décrit les structures de données des requêtes et des réponses.

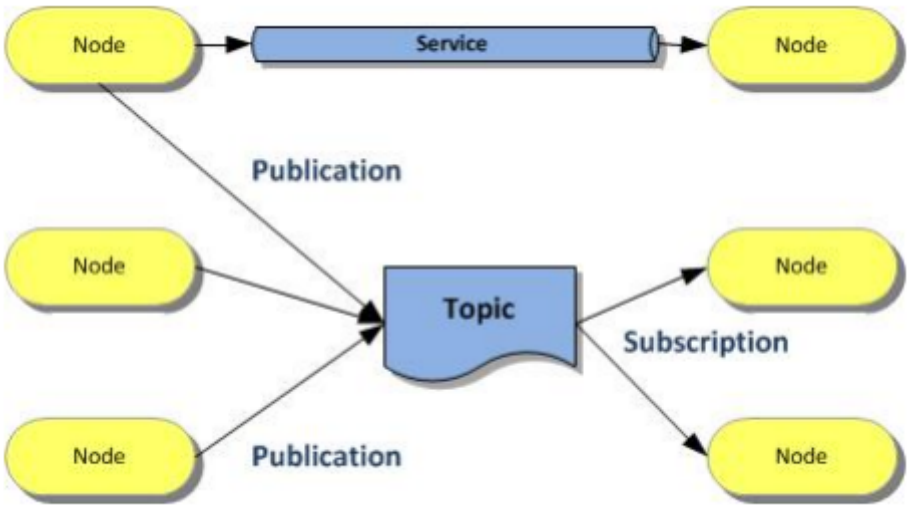



**Figure 4.1 : Mode de communication synchrone et asynchrone**

-Les bags : Les « bags » sont des formats pour stocker et rejouer les messages échangés. Ce système permet de collecter par exemple les données mesurées par les capteurs et les rejouer ensuite autant de fois qu'on le souhaite afin de faire de la simulation sur des données réelles. Ce système est également très utile pour déboguer un système a posteriori

## 4-2-2 De nombreux outils très utiles dans ROS :

ROS est une collection d'outils et d'algorithmes. Certains sont très utilisés lors de la programmation, de la simulation ou de l'exécution des comportements des robots. Citons quelques outils ou algorithmes que le programmeur de ROS retrouvera souvent :

• Stage : un simulateur 2D.

• Gazebo : un simulateur 3D.

• Rviz : un système de visualisation 3D (contrairement à Gazebo, il n'inclut pas de moteur physique).

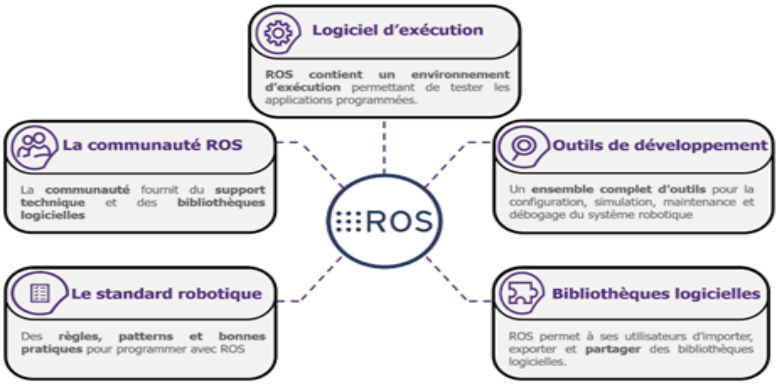



**Figure 4.2: ROS et ses différents composants**

## 4-3 l'implémentation de PASSAVOID (programmation c++ sous ROS) :

Pour implémenter l'algorithme PassAvoid (algorithme 2) sur la chaise robotisée, nous avons utilisé le système d'exploitation ROS.

Pour programmer sous ROS, il faut avant tout créer un workspace. Ce dernier contient trois documents : « build », « devel » et un document source « src » sur lequel les packages vont être créés (Figure. 4.3).

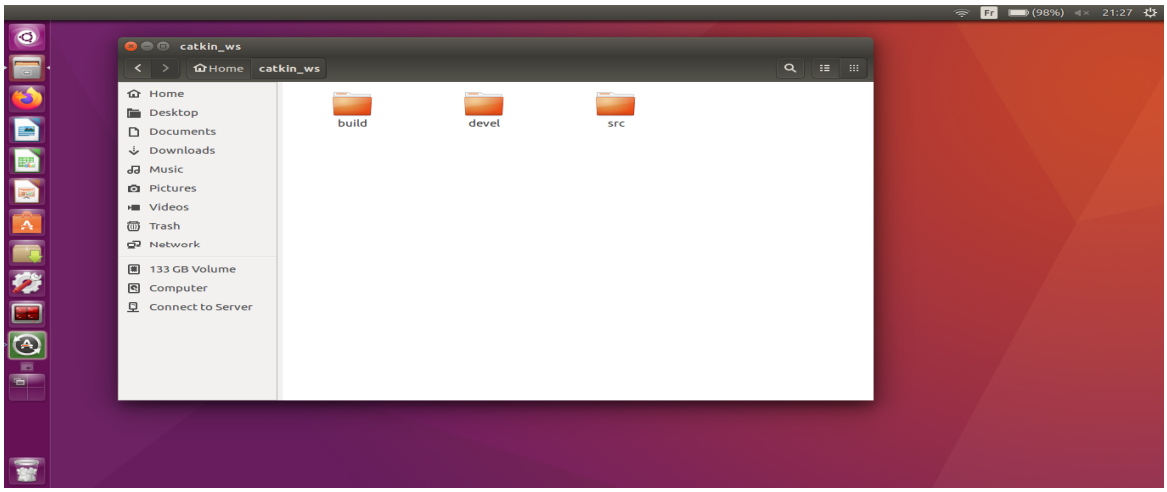



**Figure 4.3 : Capture d'écran du contenu de ROS workspace**

Pour créer un package, nous devons choisir son nom et les dépendances qu'il va contenir. Ces dépendances incluent les types de messages et les langages de programmation utilisés. Le package créé contient : deux documents « include », « src » et deux fichiers « CMakeList » et « package.xml» (Figure. 4.4).

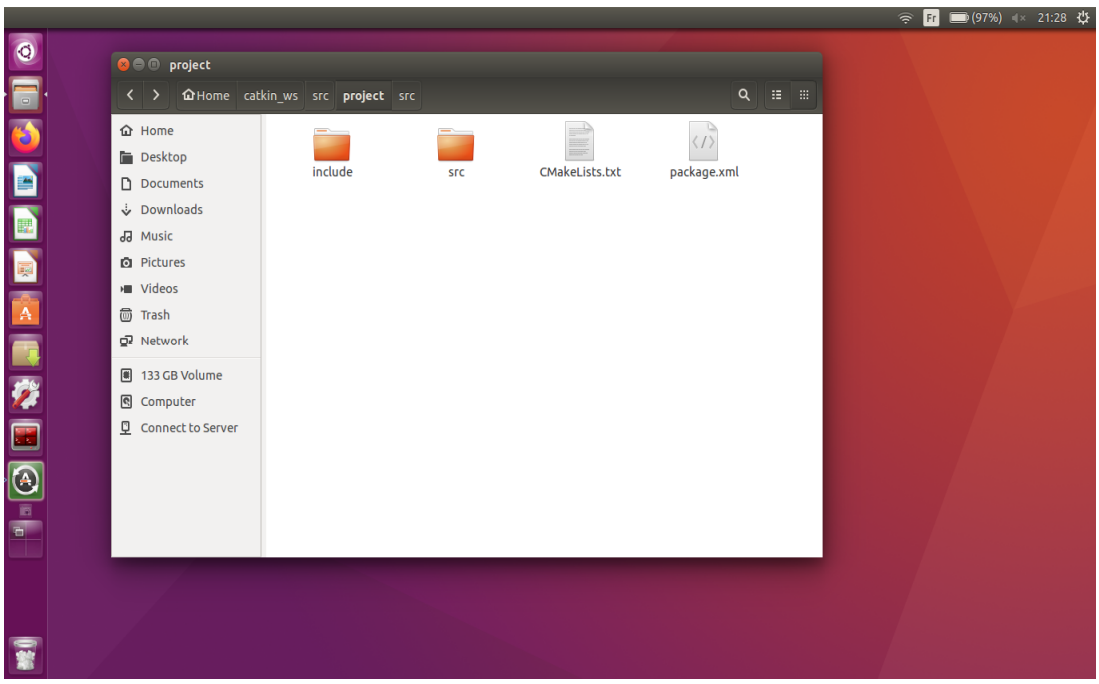



**Figure 4.4: Capture d'écran du contenu de package.**

-Le document « include » contient les bibliothèques qu'on peut utiliser selon les dépendances choisies.

-Le document « src » est l'emplacement où on va créer les nœuds.

-Le fichier « package.xml » contient les dépendances de build et de run de l'exécutable.

-Le fichier « CMakeList » contient les dépendances des langages et des types de messages utilisés. Afin d'exécuter le programme, on doit ajouter deux lignes dans le CMakeList, ces deux lignes sont représentées dans la figure suivante (elles sont encadrées) (figure 4.5).

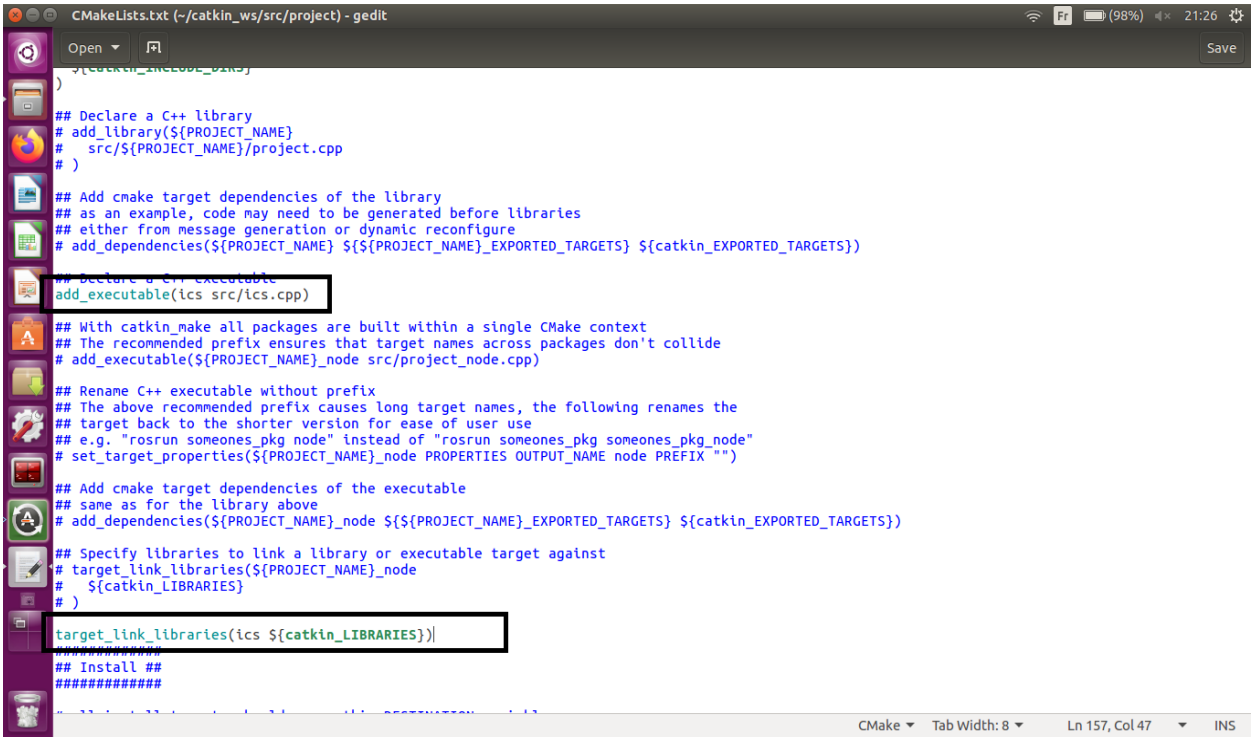



**Figure 4.5 : Capture d'écran qui montre les lignes à changer dans un fichier CMakeList.**

Dans le document src de package, le fichier bag est mis et un nœud est créé (figure 4.6).

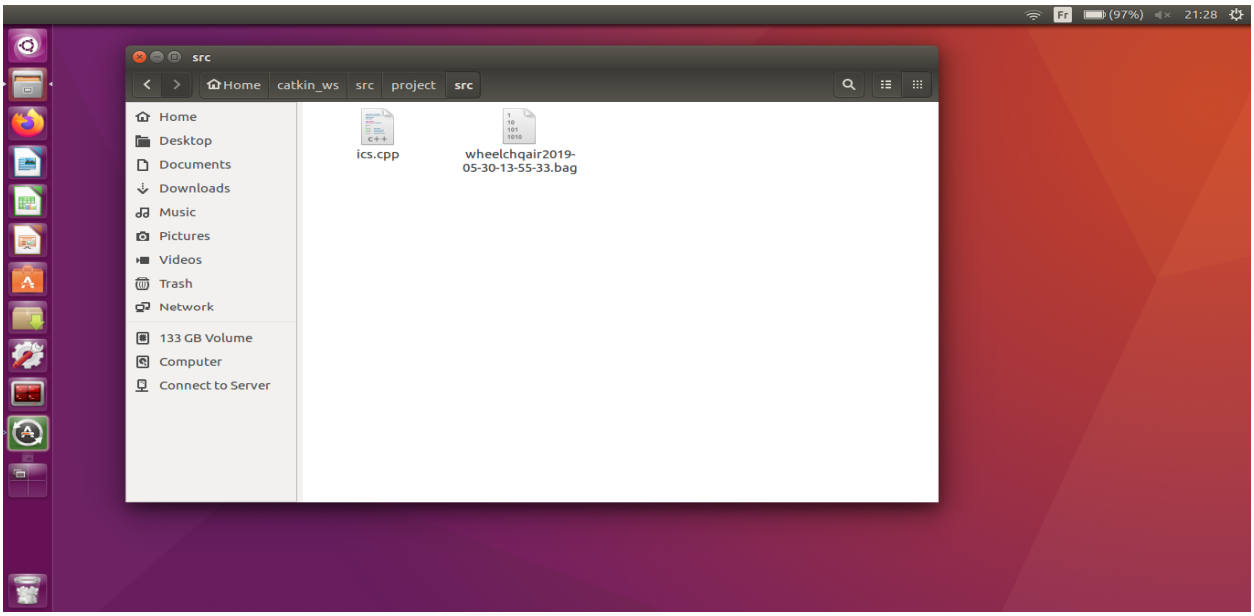



**Figure 4.6 : Capture d'écran de contenu de fichier source de package.**

Ce dernier souscrit à un fichier bag pour lire les données des capteurs laser et des odomètres, pour calculer les zones ICS et générer le control qui permet au robot de se déplacer d'un point sûr à un autre en publiant la vitesse linéaire et angulaire.

Après avoir créé le nœud et écrit le code source, il est temps de le compiler, mais avant tout il faut lancer le Master en utilisant la commande « roscore » (figure 4.7).

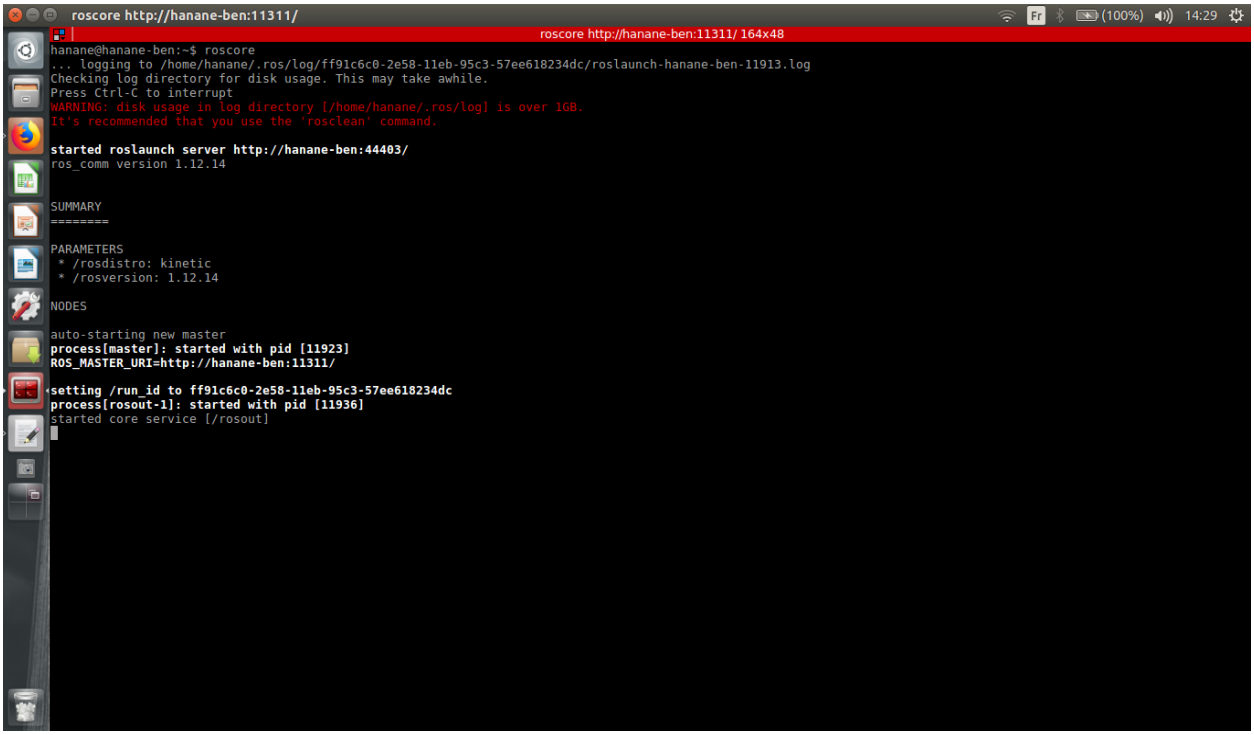



**Figure 4.7 : Lancement de master.**

Ensuite, nous devons accéder au workspace et sélectionner la source en utilisant les commandes suivantes dans le terminal linux (Figure 4.8).

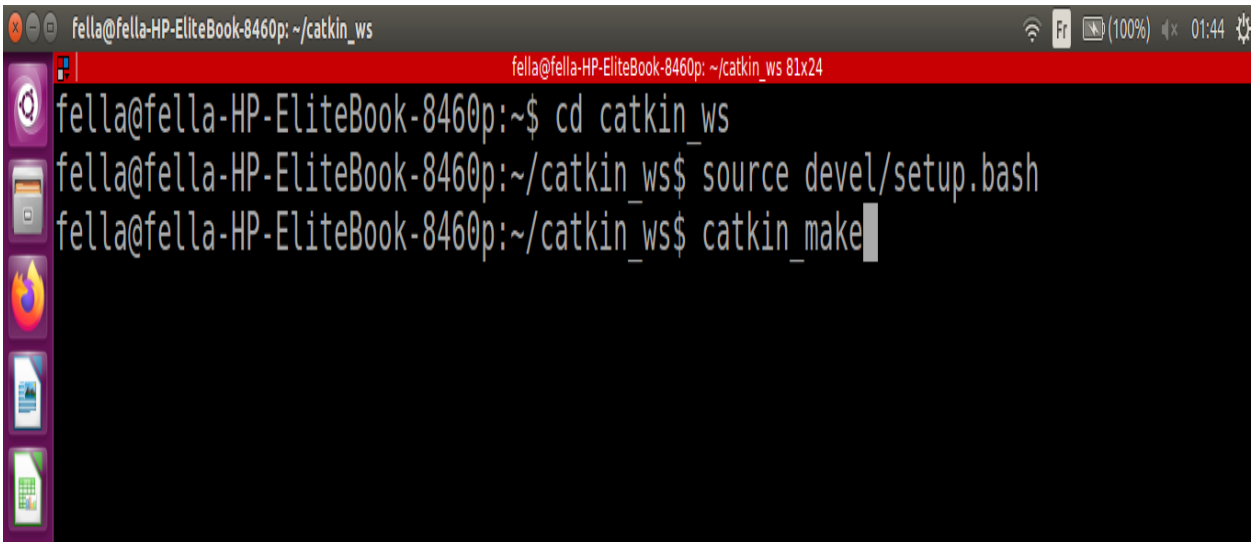



**Figure 4.8 : Les commandes de build et compilation.**

Maintenant, nous pouvons compiler le workspace en utilisant la commande « catkin_make » (Figure.4 .9).

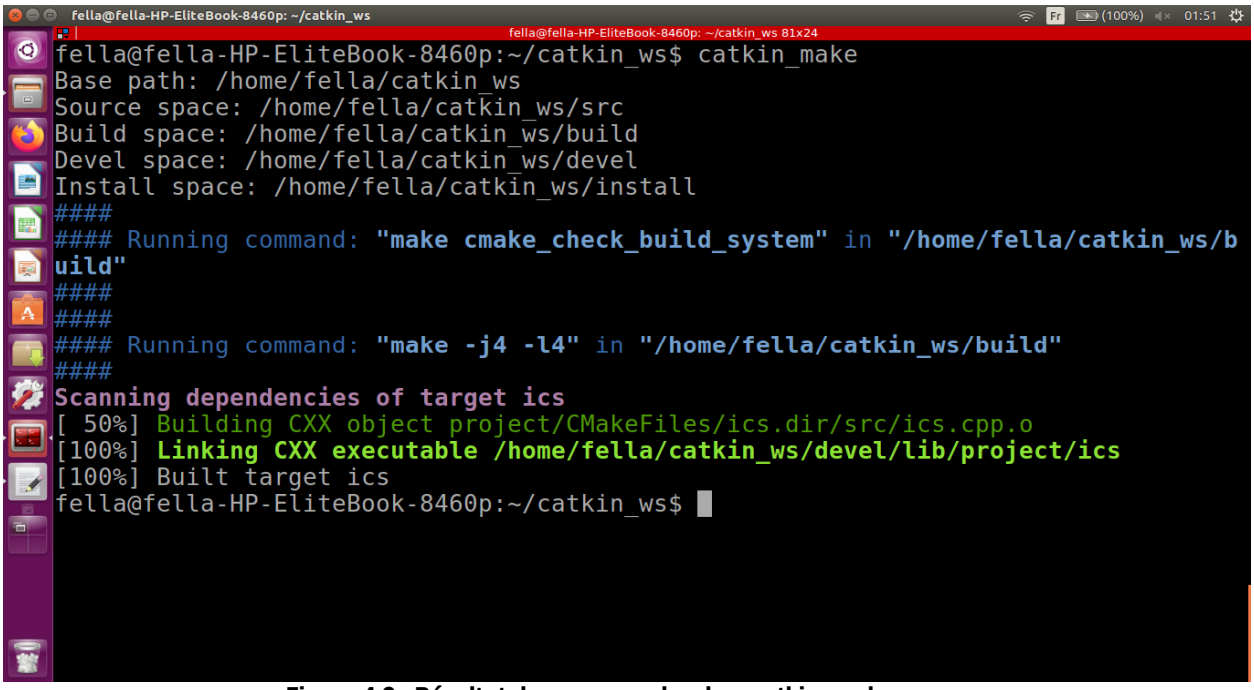



**Figure 4.9 : Résultat des commandes de « catkin_make ».**

## 4-4 Test et résultats :

Les données utilisées sont des données réelles capturées à bord de la chaise et enregistrées dans un fichier bag. En lançant le fichier bag par la commande « rosbag play » le programme se comporte comme s'il est exécuté directement sur le robot. Les topics /scan et /ARDUINO1/Odom sont lancés et le bag publie ses données pour que le nœud peut s'inscrire à ces topics et avoir les informations

La simulation Rviz est une représentation en 3D de l'emplacement des obstacles dans l'entourage du robot, Les données affichées par Rviz dans la figure 4.10 sont le scan laser

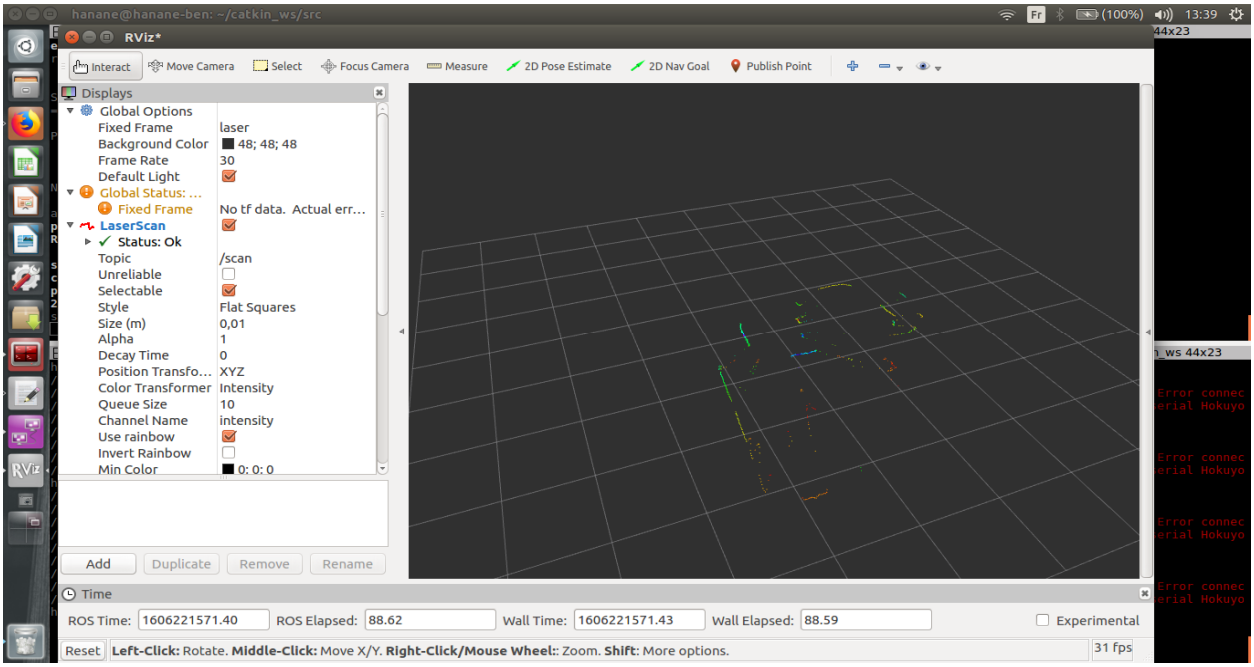



**Figure 4.10 : simulation Rviz**

L'outil rqt_graph montre les liens topic/nœud, il représente les topics sous forme de rectangles et les nœuds en forme ovale, le lien entre notre nav nœud Nav_node et les topics /scan, /ARDUINO1/Odom et /ARDUINO1/Velicity et est présenté dans la figure 4.11

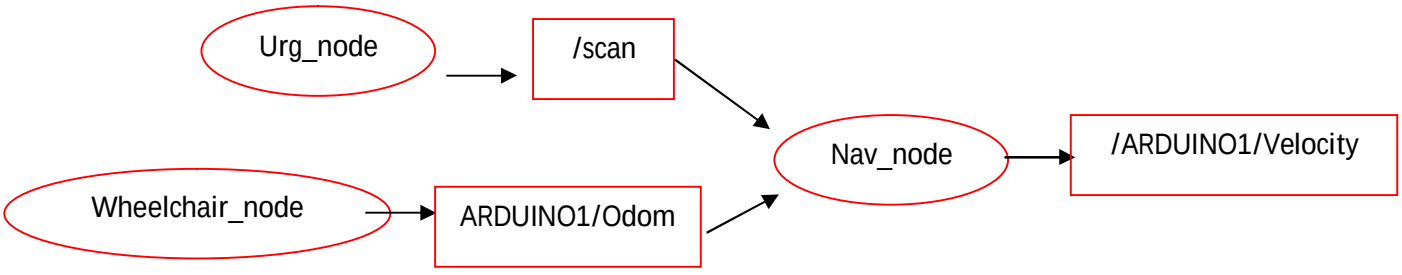



**Figure 4.11 : rqt_graph lien entre bag et notre nœud**

Ainsi que la représentation graphique, l'outil « rostopic list » nous met au courant de tous les topics actifs (figure 4.12).

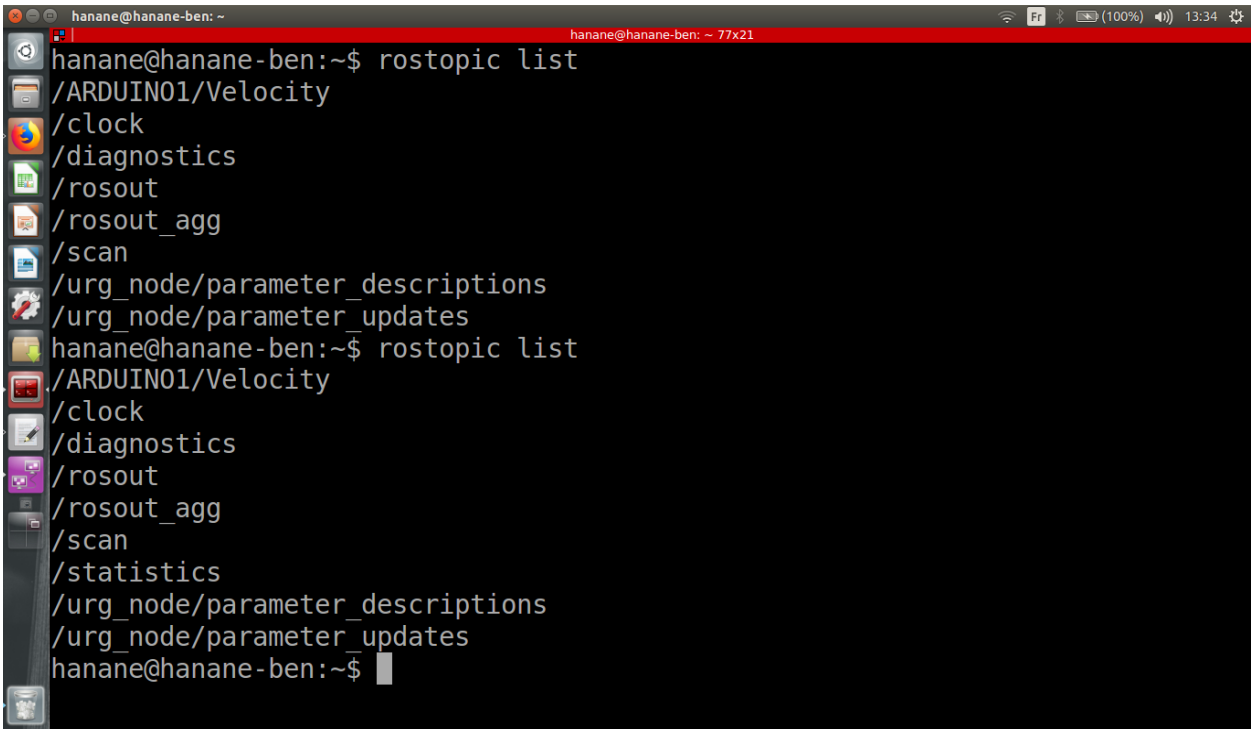



**Figure 4.12: Résultat de « rostopic list »**

Le but de l'algorithme d'intelligence artificielle que nous avons développé est de prendre la bonne décision concernant la direction du robot en se basant sur le calcul des régions interdites et de les éviter. Cette approche de navigation réactive s'intéresse non seulement à l'évitement des obstacles en toute sécurité, mais aussi à orienter le robot vers un point but final. La vitesse linéaire de la chaise est égale à 10 m/s avec un vecteur des vitesses angulaires entre -12 rad/s et 12 rad/s avec un pas de 4 rad/s entre chaque trajectoire considérée, entouré d'obstacles de vitesse max égale à 15 km/h environ 4.17 m/s

## 4-4-1 résultats obtenus pour le cas statique :

Après avoir fait tout le calcul, l'écart entre l'état prochain et le point but doit être déterminé pour choisir celui qui est le plus proche et sélectionner le contrôle qui mène à cet état.

La figure suivante est un scan écran du résultat exécuté sur laquelle les trajectoires sont classées par rapport à la sureté de chacune et les distances minimales sont calculées (figure 4.13).

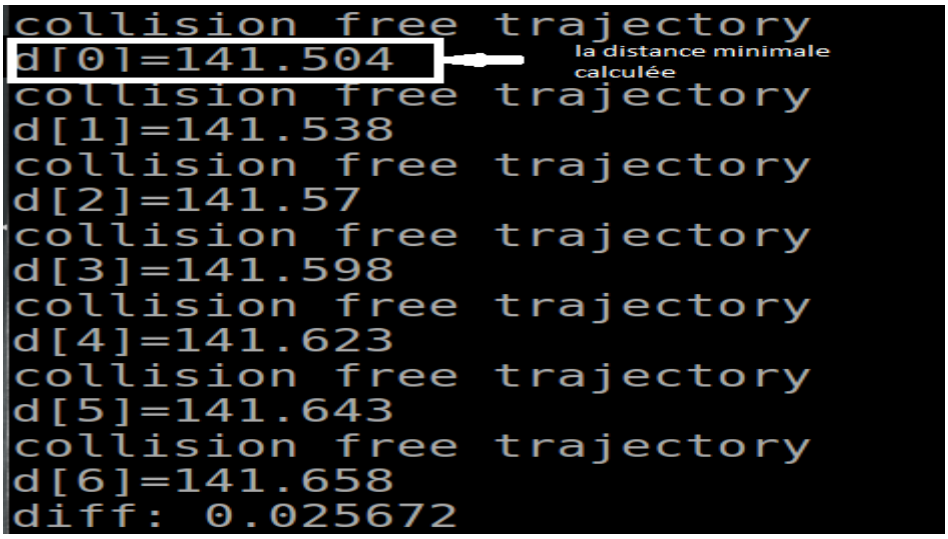



**Figure 4.13: Capture d'écran de la console des résultats pour l'environnement statique.**

L'affichage de l'outil « rostopic echo » est capturé pour montrer que le programme fonctionne bien, qu'il a sélectionné les bonnes informations et que la première trajectoire est choisie (figure 4.14).

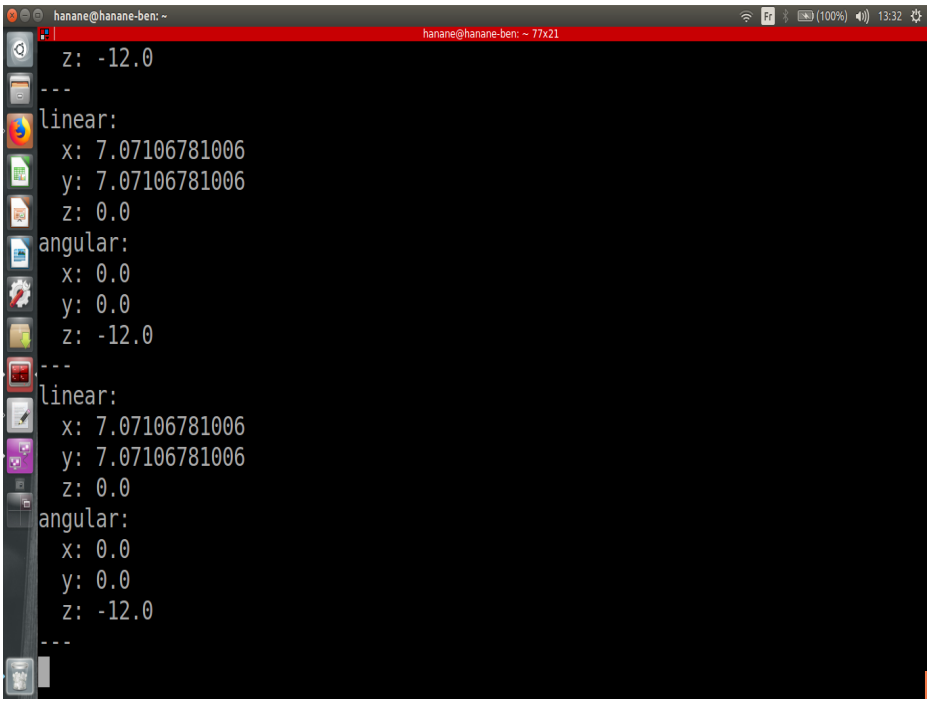



**Figure 4.14 : Capture d'écran de topicecho pour l'environnement statique**

## 4-4-2 résultats obtenus pour le cas dynamique :

Tout comme pour le cas statique, mais en intégrant la vitesse max des obstacles dans les fonctions de calcul, on a obtenu la distance la plus petite vers le but (figure 4.15) et le control qui lui correspond a été choisi (figure 4.16).

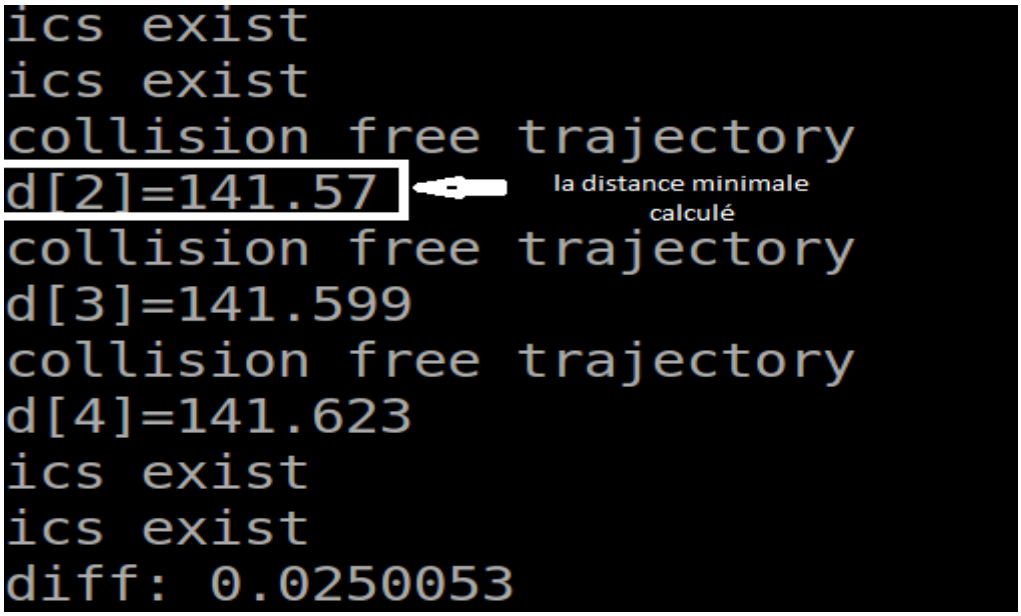



**Figure 4.15: Capture d'écran de la console des résultats pour l'environnement dynamique**

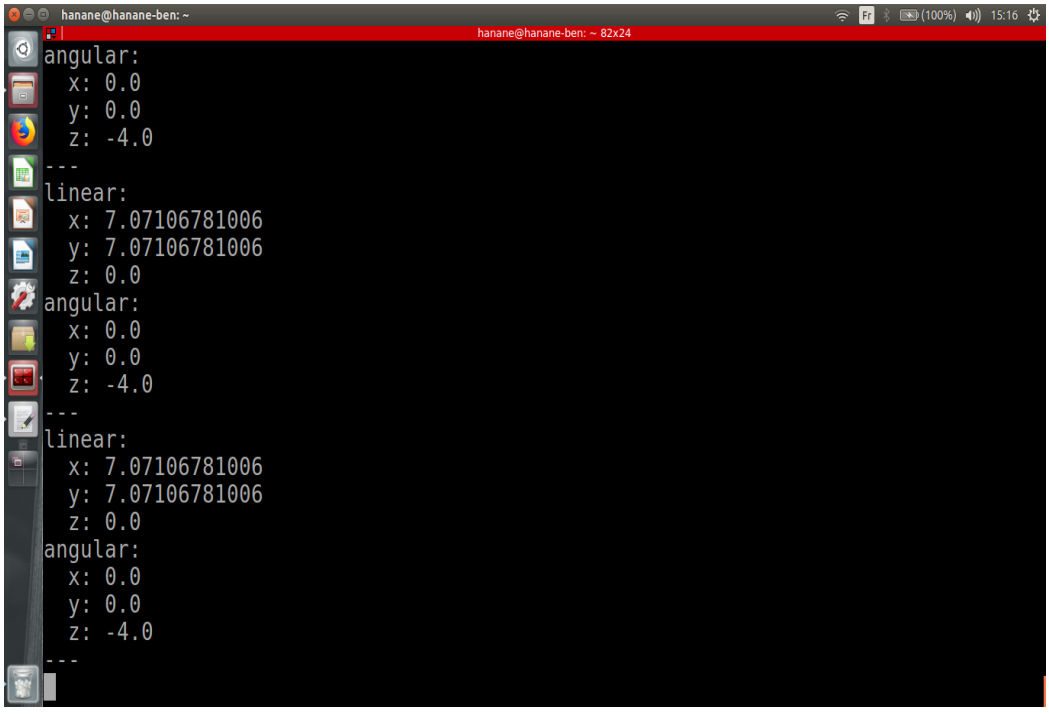



**Figure 4.16: Capture d'écran de topic echo pour l'environnement dynamique**

Comme c'est visible sur les deux figures si dessus la troisième trajectoire qui correspond à la distance minimale a été choisi ; ce qui confirme le bon fonctionnement du programme.

## 4-5 Conclusion

Ce chapitre a été consacré à la mise en œuvre des algorithmes développés dans les chapitres précédents et aux tests des différentes phases pour un environnement qui contient des objets fixes ou mobiles pour un robot mobile de type différentiel ayant un champ de vision limité.

Le calcul des états de collision inévitable de freinage a été montré dans un milieu qui contient des obstacles perçus et non-perçus. Puis, la sureté de l'état prochain du robot est vérifiée.

Les résultats des programmes effectués sous ROS ont montré la performance de ce concept de définir les ICS et les éviter en déplaçant d'un état sûr à un autre jusqu'au point de but.

# CONCLUSION GENERALE ET PERSPECTIVES

La réalisation de ce projet nous a permis de découvrir un nouveau domaine très important et passionnant qui est la navigation autonome sûre appliquée dans le domaine de la robotique. Il nous a permet aussi d'apprendre de nouvelles notions dans ce domaine.

L'objectif de ce mémoire est de développer une méthode intelligente de navigation sûre orientée dans un milieu dynamique inconnu en garantissant la sûreté de mouvement en se basant uniquement sur les informations sensorielles.

La sûreté est un niveau élevé par rapport à un évitement simple où on assure que peu importe ce qui va se passer dans l'environnement le robot ne va jamais se trouver dans une situation où il n'y a aucun moyen d'éviter la collision avec un obstacle en évitant les états de collision inévitables. Un algorithme de vérification des ICS est présenté dans ce mémoire pour vérifier si l'état prochain est interdit ou pas. Cet algorithme est intégré dans l'algorithme général de ce concept qui permet au robot mobile de passer d'un état sûr à un autre loin des ICS ; il va rejoindre son point de destination à la fin en toute sécurité. Si tous les états à vérifier sont des ICS le robot exécute la manœuvre de freinage, cela veut dire, si une collision se produit il sera à l'arrêt ; ce qui vérifie la sûreté passive, car la sûreté absolue est impossible à garantir dans le cas réel.

Cette intelligence est apte à être appliquer dans n'importe quel système robotique, sachant que nous avons utilisé des données réelles capturées à bord de la chaise roulante enregistrées dans un fichier bag.

Afin d'évaluer la performance et l'efficacité de ce concept, nous avons a programmé les algorithmes en cpp sous ROS et vérifié la sûreté passive de mouvement en désignant les régions ICS. On a testé cette approche en considérant des obstacles fixes, mobiles ainsi que les objets inattendus.

Les résultats de la simulation obtenus ont donné des résultats satisfaisants et ont prouvé que le robot prend la bonne décision pour se déplacer d'un état à un autre en toute sécurité.

A la fin de ce projet, nous sommes convaincues que la réalisation de ce mémoire a exigé beaucoup de temps mais nous a permis de développer notre sens de l'organisation, notamment dans la répartition des tâches et la gestion du temps. Ce travail nous a aussi appris la bonne façon de faire une recherche scientifique. Et malgré les obstacles que nous avons rencontrés, nous avons appris comment gérer une situation critique.

Ce travail reste, comme toute œuvre humaine, incomplet et perfectible, les améliorations que nous pouvons lui apporter sont énormes et peuvent varier selon le type d'application que nous souhaitons.

Ce travail pourrait être étendu dans les directions suivantes :

-Une méthode de planification peut être ajoutée ; ce concept est basé sur l'évitement d'obstacles mais il manque l'optimisation de la trajectoire.

- Dans certaines applications, la sûreté de mouvement passive peut être limitée parce que le robot prend toute la responsabilité dans des situations de collision. Il sera plus intéressant de partager cette responsabilité avec les obstacles et si une collision va avoir lieu, le robot sera à l'arrêt et l'objet en question aurait eu le temps de s'arrêter ou éviter la collision aussi. C'est le principe de la sûreté de mouvement amicalement passive.

- Il est plus important d'appliquer le concept de sûreté en aéronautique car l'avion demande plus de sécurité pour assurer son vol, éliminer les dégâts et minimiser la pression exercée sur les contrôleurs aériens. Nous pouvons proposer la méthode d'ICS[d], appliquée sur les navires, parce qu'elle est une méthode sûre, sur laquelle tous les critères de sureté de mouvement sont vérifiés, le principe de cette approche peut être appliqué sur la chaise roulante et sur un système aérien. La maniabilité de l'avion ressemble à celle de navire (question de s'arrêter n'importe où ou freiner est impossible que dans des endroits précis). De plus le système robotisé qui utilise ce concept n'a pas besoin de changer sa trajectoire mais l'évitement se fait dans la trajectoire elle-même, ce qui est convenable pour le cas d'un avion. Le calcule d'une distance limite entre le système robotisé et tous autres obstacles est la meilleure méthode de sureté en vol.

# Référence :


1. Contribution à la localisation dynamique du robot mobile d'intérieur B21r en utilisant la plateforme multi sensorielle par Bouraine Sara .mémoire de magister Blida. Novembre 2007.
2. Robotique mobile by Bernard Bayle. Cours Télécom Physique Strasbourg Université de Strasbourg
3. https://www.memoireonline.com/01/16/9368/m_Conception-et-realisation-d-un-robot-mobile--base-d-arduino4.html
4. Contribution à la planification de mouvement en environnements dynamiques pour des robots mobiles de type voiture: cas du ROBUCAR par Sara Bouraine .thèse de doctorat 2016

5. Dynamic window based approach to mobile robot motion control in the presence of moving obstacles. IEEE International Conference on Robotics and Automation May 2007

6. Car-Like Mobile Robot Navigation in Unknown urban Areas by Somia Brahimi, Rachid Tiar, Ouahiba Azouaoui, Mustapha Lakrouf and Malik Loudini. IEEE 19th International Conference on Intelligent Transportation Systems (ITSC) Windsor Oceanico Hotel, Rio de Janeiro, Brazil, November 1-4, 2016
7. Autonomous navigation and obstacle avoidance of a micro-bus by Fernandez, R. Dominguez, D. Fernandez-Llorca, J. Alonso, and M. A. Sotelo. International Journal of Advanced Robotic Systems, vol. 10, no. 212, 2013.
8. Darpa urban challenge, California, USA, Nov 2007. [Online]. Available: http://archive.darpa.mil/grandchallenge
9. An open experiment of autonomous navigation of mobile robot in the city: Tsukuba challenge 2014 and the results by S. Yuta. Journal of robotics and mechatronics, vol. 27, no. 4, pp. 318–326, 2015.
10. Autonomous moving technology for future urban transport by Y. Hosoda, M. Koga, and K. Yamamoto, Hitachi Review, vol. 60, no. 2, 2011.
11. An open experiment of mobile robot autonomous navigation at the pedestrian streets in the city - tsukuba challenge by S. Yuta, M. Mizukawa, H. Hashimoto, H. Tashiro, and T. Okubo, in IEEE International Conference on Mechatronics and Automation, Aug 2011, pp. 904–909.
12. Obstacle Avoidance for Flight Safety on Unmanned Aerial Vehicles by Wilbert G. Aguilar, Verónica P. Casaliglla, José L. Pólit, Vanessa Abad, and Hugo Ruiz. Springer International Publishing AG 2017
13. A 3D Collision Avoidance Strategy for UAVs in a Non-Cooperative Environment by Xilin Yang, Luis Mejias Alvarez, Troy Bruggemann. J Intell Robot Syst 2013
14. Motion Planning in Dynamic Environments Using Velocity Obstacles by P. Fiorini and Z. Shiller, The International Journal of Robotics Research, vol. 17, no. 7, pp. 760–772, 1998
15. Reciprocal Velocity Obstacles for real-time multi-agent navigation by J. van den Berg, M. Lin, and D. Manocha, IEEE International Conference on Robotics and Automation, May 2008, pp. 1928–1935
16. Convention on the International Regulations for Preventing Collisions at Sea, by IMO (COLREGs) in 1972. [Online]. Available:
http://www.imo.org/en/About/conventions/listofconventions/pages/colreg.aspx
17. The Hybrid Reciprocal Velocity Obstacle by J. Snape, J. v. d. Berg, S. J. Guy, and D. Manocha, IEEE Transactions on Robotics, vol. 27, no. 4, pp. 696–706, Aug 2011

18. Motion planning for mobile robots using the Safety Velocity Obstacles method by Zoltán Gyenes and Emese Gincsainé Szádeczky-Kardoss. , IEEE International Conference on Robotics and Automation, 2018
19. Motion Planning in Dynamic Environments using Velocity Obstacles by Paolo Fiorini and Zvi Shiller. The International Journal of Robotics Research July 1998 vol. 17.
20. IVO: Inverse Velocity Obstacles for Real Time Navigation by P. S. Naga Jyotish, Yash Goel,A. V. S. Sai Bhargav Kumar and K. Madhava Krishna. July 2019, Chennai, India
21. Reciprocal velocity obstacles for real-time multi-agent navigation by Jur Van den Berg, Ming Lin, and Dinesh Manocha. IEEE International Conference on Robotics and Automation, pages 1928–1935. 2008.
22. An Airborne Approach for Conflict Detection and Resolution Applied to Civil Aviation Aircraft based on ORCA by Haotian Niu ,Cunbao Ma ,Pei Han and Jiwu Lv. IEEE 8th Joint International Information Technology and Artificial Intelligence Conference (ITAIC 2019)
23. Towards Safe Vehicle Navigation in Dynamic Urban Scenarios by Kristijan Macek, Dizan Alejandro Vasquez Govea, Thierry Fraichard, Roland Siegwart. Automatika October 2008

24. Inevitable Collision States A Step Towards Safer Robots? by Thierry Fraichard & Hajime Asama InU. Conference on Intelligent Robots and Systems Las Vegas, Nevada. October 2003

25. Inevitable Collision States: a Probabilistic Perspective by Antoine Bautin, Luis Martinez-Gomez, Thierry Fraichard. IEEE International Conference on Robotics and Automation June 2010.

26. An efficient and generic 2D inevitable collision state checker by L. Martinez-Gomez and T. Fraichard, in Proc. of the IEEE-RSJ Int. Conf. on Intelligent Robots and Systems, Nice (FR), Sept. 2008.

27. Provably Safe Navigation for Mobile Robots with Limited Field-of-Views in Unknown Dynamic Environments by Sara Bouraine, Thierry Fraichard, et Hassen Salhi. IEEE International Conference on Robotics and Automation. June 2014

28. Motion Safety for Vessels: An Approach Based on Inevitable Collision States by Michael Blaich, Simon Weber, Johannes Reuter and Axel Hahn. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) .Congress Center Hamburg Sept 28 - Oct 2, 2015. Hamburg, Germany
29. Development of an e-Navigation Strategy by IMO Report of the Correspondence Group on e-Navigation NAV, vol. 13, no. 53, 2007.
30. Safe Motion using Viability Kernels by Mohamed Amine Bouguerra, Thierry Fraichard and Mohamed Fezari. IEEE International Conference on Robotics and Automation · May 2015
31. Viability Theory: New Directions by J.-P. Aubin, A. Bayen, and P. Saint-Pierre. Springer, 2011
32. Approximation of the viability kernel by P. Saint-Pierre. Applied Mathematics and Optimization, vol. 29, no. 2, pp. 187–209, 1994.
33. Robot Motion Planning by Latombe J.-C. Livre, Kluwer, Boston, MA, (1991).
34. On the complexity of kynodynamic planning by J. Canny, B. Donald, J. Reif, and P. Xavier. In Proc of the Symp. on the Foundations of Computer Science, pages 306 {316, White Plains, NY (US),November 1988.

35. Provably-good approximation algorithms for optimal kinodynamic robot motion plans by P.G. Xavier. PhD thesis, Cornell Univ., Ithaca, NY (US), April 1992.
36. A Short Paper about Motion Safety by Thierry Fraichard, Inria Rhône-Alpes and LIG-CNRS Lab, Grenoble (FR) IEEE International Conference on Robotics and Automation Roma, Italy, 10-14 April 2007

37. Iterative Motion Planning and Safety Issue by Thierry Fraichar and Thomas M. Howard. Book January 2012.
38. The vector field histogram – fast obstacle avoidance for mobile robots by Borenstein, J.Y. Korem. IEEE Transctions on Robotics and Automation 7(3), 278–288 (1991).
39. Motion Planning in Dynamic Environments Using Velocity Obstacles by Paolo Fiorini, Zvi Shiller**.** First Published July 1, 1998.sage journals
40. Motion prediction of moving objects based on autoregressive model by A. Elnagar and K. Gupta. IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans (Volume: 28, Issue: 6, Nov 1998).
41. Analysis of gross-chromosomal rearrangements in Saccharomyces cerevisiae. *Methods Enzymol* by Schmidt KH, et al. IEEE International Conference on Robotics and Automation (2006)

42. Computational Geometry by de Berg M, Cheong O, van Kreveld M and Overmars M. book on springer 2006
46. Présentation de ROS par Jerome Laplace.devlpppez.com club des developpeurs 23 Novembre 2011.
47. Détection et reconnaissance de doigts par camera « RGB-D » pour tele-operation gestuelle d'une main robotique par khaled belgacem. Université vincennes-saint-denis 2014