République Algérienne Démocratique et Populaire
Ministère de l'enseignement Supérieur et de la Recherche Scientifique
Université Ammar Telidji – Laghouat
Faculté Des Sciences et Sciences de l'ingénierie

PROJET DE FIN D'ETUDES

Pour L'obtention du Diplôme

D'INGENIEURE D'ETAT EN INFORMATIQUE

Option : Intelligence Artificielle

Thème :

Commande Référence Du Robot KHEPERA

Réalisé par :

- Ben abdallah Mounir

- Daoula Abd elhakim

Encadré par :

- Melle : TAABA Kheira

Année Universitaire : 2009 / 2010

Table de figures :

Sommaire

Chapitre III : Présentation du simulateur

INTRODUCTION GENERAL

L'homme conçoit et construit des robots pour le remplacer ou l'assister dans de nombreuses tâches. La télé robotique associant ces deux aspects d'autonomie et de télé opération, est un des principaux domaines de la robotique. La télé robotique est une forme avancée de télé opération où par exemple un véhicule télécommandé, peut fonctionner sans la supervision de son opérateur (i.e. de façon autonome) durant de courtes périodes, ce qui fait de lui un robot. L'objectif d'un robot mobile est de se déplacer d'un point de départ à un point cible en évitant les obstacles, tout en essayant d'optimiser sa trajectoire. Le déplacement se fait d'une manière autonome et /ou télé opérée. Il est nécessaire de contrôler la navigation afin d'avoir un système de commande robuste.

Dans le vaste domaine de la robotique mobile, l'étude du déplacement a une grande importance. De nombreuses approches du déplacement existent en utilisant différents types de capteurs. Les robots mobiles, connaissant leur position, se déplacent d'un point à un autre . Dans certains cas, seule trajet parcouru est important tout comme sa dépendance du temps. Dans d'autres cas, la trajectoire à suivre par le robot doit être définie.

A cet effet nous présentons dans ce rapport le but de travail qui est :le suivi de trajectoire par le robot mobile "KHEPERA" en utilisons comme méthode d'optimisation : les algorithmes génétiques qui s'appuyant sur des techniques derives de la génétique et des mécanismes d'évolution de la nature : croisement, mutation, selection.

nous organisons ce memoire en trois chapitre:

*le premier chapitre pour decrit les concepts fondamentaux de robotique mobile :aperçu historique ,leur utilite et domaine d'application.

*le deuxieme chapitre donne une idee sur le principe générale de l'algorithme génétique nous citons ici les différents phases manipulées telque:sélection des individus, croisement et mutation.et aussi les méthodes de representation des individus dans l'AG (codage) .

* et dans le troisième chapitre en présente notre logiciel de simulation qui dons les resultants de suivi de trajectoire des différents environnements .

I.1.Introduction :

De manière générale, on regroupe sous l'appellation robots mobiles l'ensemble des robots à base mobile, par opposition notamment aux robots manipulateurs. L'usage veut néanmoins que l'on désigne le plus souvent par ce terme les robots mobiles à roues. Les autres robots mobiles sont en effet le plus souvent désignés par leur type de locomotion, qu'ils soient marcheurs, sous-marins ou aériens.

On peut estimer que les robots mobiles à roues constituent le gros des robots mobiles. Historiquement, leur étude est venue assez tôt, suivant celle des robots manipulateurs, au milieu des années 70. Leur faible complexité en a fait de bons premiers sujets d'étude pour les roboticiens intéressés par les systèmes autonomes. Cependant, malgré leur simplicité apparente (mécanismes plans, à actionneurs linéaires), ces systèmes ont soulevé un grand nombre de problèmes difficiles. Nombre de ceux-ci ne sont d'ailleurs toujours pas résolus. Ainsi, alors que les robots manipulateurs se sont aujourd'hui généralisés dans l'industrie, rares sont les applications industrielles qui utilisent des robots mobiles. Si l'on a vu depuis peu apparaitre quelques produits manufacturiers (chariots guidés) ou grand public (aspirateur), industrialisation de ces systèmes bute sur divers problèmes délicats.

Ceux-ci viennent essentiellement du fait que, contrairement aux robots manipulateurs prévus pour travailler exclusivement dans des espaces connus et de manière répétitive, les robots mobiles sont destinés à évoluer de manière autonome dans des environnements peu ou pas structurés. [01]

Néanmoins, l'intérêt indéniable de la robotique mobile est d'avoir permis d'augmenter considérablement nos connaissances sur la localisation et la navigation de systèmes autonomes. La gamme des problèmes potentiellement soulevés par le plus simple des robots mobiles à roues en fait un sujet d'étude à part entière et forme une excellente base pour l'étude de systèmes mobiles plus complexes.

I.2. Aperçu historique :

Les premiers travaux sur la robotique remontent a la fin de la deuxième guère mondiale. Ver la fin des années quarante, des programmes des recherches commencèrent

à Oak Ridge et laboratoire national d'argent pour développer du manipulateur mécanique téléguidés dans le but de transporter les matériaux radioactifs.

Au milieu des années cinquante, les travaux sur les manipulateur téléguidés ont aboute a des systèmes capables de faire des opérations répétitives et autonomes, c'est le cas des manipulateurs programmables commande par un grand ordinateur et qui a été crée par George C.DEVOL.

1948-1949 **William Grey Walter** s'appuie Elmer et Elsie, deux robots autonomes qui ressemblaient à des tortues. Officiellement, ils étaient appelés **Machina speculatrix** parce que ces robots aimait à explorer leur environnement. Elmer et Elsie étaient équipés d'un capteur de lumière, si ils ont trouvé une source de lumière, ils se déplacerait vers elle, d'éviter ou le déplacement d'obstacles sur leur chemin. Ces robots ont démontré que les comportements complexes pourraient découler d'une conception simple, Elmer et Elsie seulement eu l'équivalent de deux cellules nerveuses. [02]

FIG.I.1 : La tortue de Grey Walter

C'est vers la fin des années 1960 que, de deux sources complètement différentes, est apparu le concept du robot mobile autonome :

d'un coté, spécifiquement au standford reseaech institute, ont été menées des recherches sur les possibilités d'équiper des machines de capacité de déduction et de réaction « logique » à des événements extérieurs. Pour essayer en vraie grandeur les principes développés par les chercheurs, on construit Shakey, c'est une machine à roue, connectée à un gros ordinateur, et qui évolue dans un univers su cubes et de pyramides de taille et de couleur différentes. ces missions : prendre un objet et le porter ailleurs, quelle que soit la position absolue dans la salle d'évolution relative par rapport à d'autres objets. ses moyens de perception : essentiellement un camera qui lui permet d'acquérir des images

de son environnement. Ses performances : une cinquantaine de minute pour effectuer une mission.

De l'autre coté, l'industrie nucléaire a besoin d'une machine permettent d'agir à distance dans son environnements encombrés et inaccessibles à l'homme. General Electric développe quadrupède pour essayer de résoudre ce problème. A peu prés en même temps, au début de l'industrie spatial s'échafaudent (en particulier au Jet Propulsion Laboratory de Pasadna)les projet de Luna et Mars Rover, destinés à permettre l'exploration des planètes avec des opérateurs restant sur terre.

Laboratoire et industriels, informaticiens vont continuer leurs travaux en parallèle pendant plusieurs années. En découleront, coté recherche en informatique, les développements de l'intelligence artificielle et coté industriel, la télé opération est une partie de la robotique classique.

Les robots mobiles autonomes eux, résultent de la synthèse de secs travaux. Celle-ci s'est effectuée vers la fin des années 70, avec une trois pôles géographiques principaux (le Japon, les Etat –Unis et la France) et une très large gamme d'application (du robot domestique au robot militaire ou l'exploration planétaire). Le nombre de chercheurs crû notablement et leur orientation vers la solution de problèmes industriels s'est précisée en particulier vers le domaine des tâches non productives.

En citons quelque robots et les années de leurs développement :

- **JASON :** de l'université de BARKELEY, 1970 proposait de solutions légèrement différentes quand à son système décisionnel. Celui-ci fut élaborer autours de deux générateur de plans l'un procédural, l'autre utilisant des concepts flous.

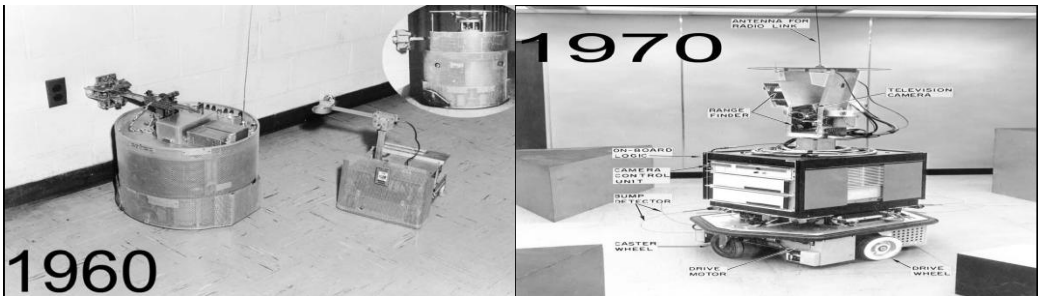


FIG.I.2 : A gauche : Robot "Beast" de l'université John Hopkins dans les années 1960.
A droite : Le robot Shakey de Stanford en 1969 a été une plate-forme de démonstration des recherches en intelligence artificielle.

- **En 1975 JPL** du Jet Propulsion Laboratory, la première étude du robot destinée **à** l'exploration planétaire. Son système de locomotion présentait une nette progression puisqu'il s'agissait d'une plate forme à quatre roues motrices et directrices.

- **CART** de l'université de standford débuta en 1979 et fut principalement à l'étude de perception à partir d'une seule camera.

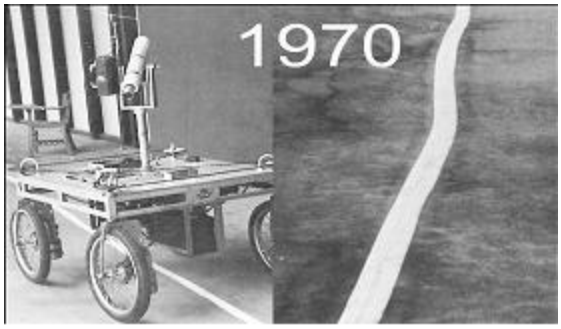


FIG.I.3 : Le Stanford Cart date des années 1980

- **CMU ROVER** de Carnegie-Million University ,1981 marque une progression importante quant à l'architecture informatique des robots mobiles. En effet, la puissance de calcule embraquée s'accroît, chaque fonction possédant son propre microprocesseur. Ce robot peut atteindre une vitesse de 10 km/h.

- **VESA,** crée à l'INS de rennes en 1981 est doté d'un bras tactile télescopique ainsi que d'un arceau de sécurité afin de réaliser la détection des obstacles dans un environnement inconnu.

- **En 1984, le robot YMABIKO**, sa navigation est basée sur une modélisation de l'environnement sous la forme d'un squelette composée d'un ensemble de points représentant chacun un obstacle.

- **Les robots HERMIS I et HERMIS II-B en 1985.**

HERMIS II-B est le dernier né d'une série de robots mobile autonome développés au CESAR-YAK Ride National Laboratory'c Center. Ces robots sont capables d'évoluer dans des environnements hostiles plus ou moins connus.

- **ROBOTRYER en 1986,** le robuter est un robot mobile destiné à roues multifonctions, de base rectangulaire ou circulaire destiné aux applications industrielles. Ce robot est maintenant fabriqué commercialisé par la firme française Robosoft.

- **Le robot Blanche3 en 1991**, est robot mobile à trois roues : une directrice à l'avent, deux fixes à l'arrière. Ce robot développé au NEC Research institute, a été conçu

pour évoluer dans les couloires connus s'un bâtiment administratif ou hospitaliser plutôt que dans un environnement naturel non structuré Jusqu'à maintenant, il n'existe pas encore de production massive de robot mobile. Pourquoi ?on peut s'étonner que le progrès technologiques et la baisse des coûts en matière de capteurs, d'actionneurs et de processeurs ne débouchent pas plus vite. Seulement, la problématique posé par la robotique ne se réduit pas à la technologie. Pathfinder.p3 et Aibo sont d'inclinables succès. Ils restent des succès technologiques qui ne doivent que peut aux progrès réalises en trente ans de recherche. [07]

I.3. Définition :

Un robot est une machine équipée de capacités de perception, de décision et d'action qui lui permettent d'agir de manière autonome dans son environnement en fonction de la perception qu'il en a. [03]

La robotique mobile est un champ de recherche très varié, du fait principalement des nombreuses applications potentielles. Afin de réaliser des tâches dans le cadre d'une de ces applications, un robot doit posséder la capacité de naviguer dans son environnement. [04]

I.4. Les applications des robots mobiles :

Aujourd'hui, le marché commercial de la robotique mobile est toujours relativement restreint, mais il existe de nombreuses perspectives de développement qui en feront probablement un domaine important dans le futur. Les applications des robots peuvent se trouver dans de nombreuses activités "ennuyeuses, salissantes ou dangereuses" , mais également pour des applications ludiques ou de service, comme l'assistance aux personnes âgées ou handicapées.

Parmi les domaines concernés, citons :

– La robotique de service (hôpital, bureaux)

– La robotique de loisir (aibo, robot 'compagnon')

– La robotique industrielle ou agricole (entrepôts, récolte de productions agricoles, mines)

– La robotique en environnement dangereux (spatial, industriel, militaire)

A cela, s'ajoute a l'heure actuelle des nombreuses plates-formes conçues essentiellement pour les laboratoires de recherche. La Figure .I.4 montre quelques exemples de robot réel :

FIG.I.4 : Exemples de robots utilisés dans différentes applications

I.5. Les avantages des robots mobiles :

Un robot entièrement autonome a la capacité de

Obtenir de l'information sur l'environnement.

- Travail pour une période prolongée sans intervention humaine.
- Déplacer tout ou partie d'elle-même tout au long de son environnement de fonctionnement sans assistance humaine.
- Évitez les situations qui sont nocifs pour les personnes, les biens, ou lui-même sauf si ceux-ci sont une partie de sa conception.

I.6. Présentation de robot KHEPERA :

Le robot KHEPERA est un minirobot conçu comme outil de recherches et d'enseignement dans le cadre d'un programme suisse de recherches. Il a été développé la

première fois en 1992, par une équipe de recherche du laboratoire de microprocesseur et d'interface (LAMI) à l'institut de la technologie fédéral suisse Lausanne (EPFL). Il permet la confrontation au vrai monde des algorithmes développés dans la simulation pour l'exécution de trajectoire, l'action d'éviter d'obstacle, et le prétraitement d'information sensorielle. Le robot KHEPERA est maintenant largement répandu autour du monde comme plateforme pour différentes expériences et applications de la robotique.

Le KHEPERA (diamètre 55 mm, hauteur 30 mm, poids 80 g dans sa configuration de base) est montré sur la Figure .I.5, il dispose d'un processeur embarqué performant (M68331, 32 bits, cadencé à 16 MHz), associé à une EEPROM de 128 K octets et une RAM statique de 256 K octets. Le BIOS (Basic I/Os System) est le système bas niveau embarqué dans le robot. Il offre des capacités multitâches et permet la gestion de plusieurs modules logiciels : acquisition et conversion des données sensorielles, asservissements des moteurs, contrôle de la communication entre différents modules du robot et l'extérieur.[04]

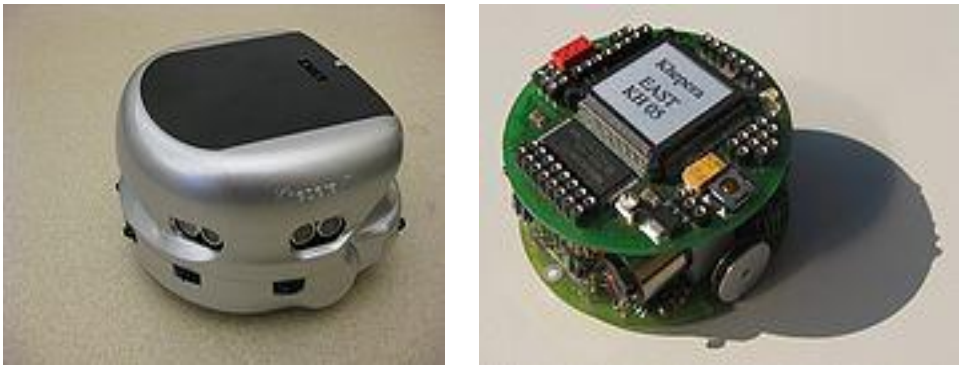


FIG.I.5 : Le robot KHEPERA

I.7. modélisation :

I.7.1.Description :

Le robot KHEPERA est un robot de type unicycle, est actionné par deux roues indépendantes et possédant éventuellement une roue folle assurant sa stabilité. Le schéma des robots de type unicycle est donné à la Figure .I.6.

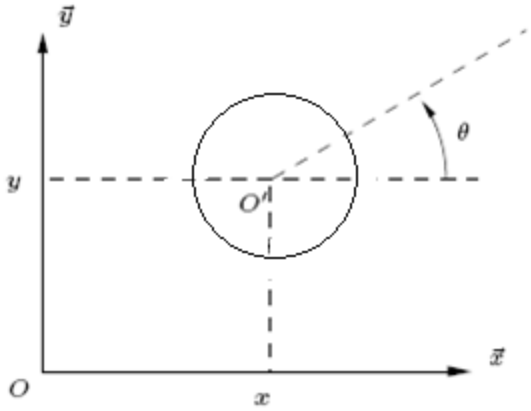


FIG.I.6 : Robot mobile de type unicycle

Ce type de robot est très répandu en raison de sa simplicité de construction et de propriétés cinématiques intéressantes.

I.7.2. Définition :

On note R= (O , $\vec{x}$, $\vec{y}$, $\vec{z}$). un repère fixe quelconque, dont l'axe $\vec{z}$ est vertical, Et $\dot{R}$ = ($\dot{O}$, $\vec{x}$, $\vec{y}$, $\vec{z}$) Un repère mobile lié au robot. On choisit généralement pour $\dot{O}$ un point remarquable de la plate-forme, typiquement le centre de l'axe des roues motrices, comme illustré à la Figure .I.7 .

Par analogie avec la manipulation, on appelle situation ou souvent posture du robot le vecteur :

$$\xi = \begin{pmatrix} x \\ y \\ \theta \end{pmatrix} \qquad (1.1)$$

Où x et y sont respectivement l'abscisse et l'ordonnée du point $\dot{O}$ dans R et θ l'angle ($\vec{x}$, $\vec{x}$). La situation du robot est donc définie sur un espace M de dimension m = 3, comparable à l'espace opérationnel d'un manipulateur plan. [01]

La configuration d'un système mécanique est connue quand la position de tous ses points dans un repère donné est connue. Alors que pour un bras manipulateur cette notion est définie sans ambiguïté par les positions angulaires des différentes articulations, on peut, dans le cas d'un robot mobile, donner une vision plus ou moins fine de la configuration, comme on le verra par la suite. Dans tous les cas, on définira la configuration du robot mobile par un vecteur :

$$q = \begin{pmatrix} q1 \\ q2 \\ \cdot \\ \cdot \\ \cdot \\ q3 \end{pmatrix} \qquad (1.2)$$

De n coordonnées appelées coordonnées généralisées. La configuration est ainsi définie sur un espace N de dimension n, appelée l'espace des configurations.

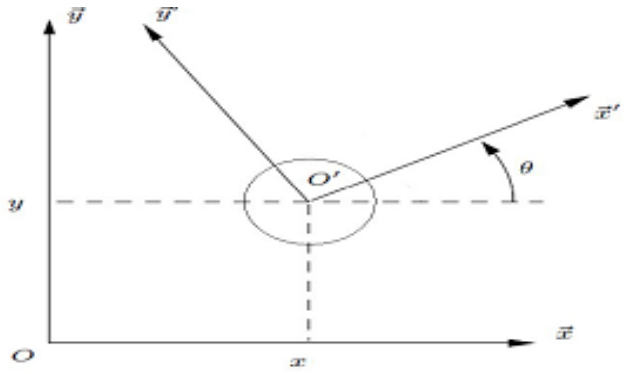


FIG.I.7 : Repérage d'un robot mobile

I.7.3. Roulement sans glissement :

La locomotion à l'aide de roues exploite la friction au contact entre roue et sol. Pour cela, la nature du contact (régularité, matériaux en contact) a une forte influence sur les propriétés du mouvement relatif de la roue par rapport au sol. Dans de bonnes conditions, il y a roulement sans glissement (r.s.g.) de la roue sur le sol, c'est-à-dire que la vitesse relative de la roue par rapport au sol au point de contact est nulle. Théoriquement, pour vérifier cette condition, il faut réunir les hypothèses suivantes :

– le contact entre la roue et le sol est ponctuel ;

– les roues sont indéformables, de rayon *r*.

En pratique le contact se fait sur une surface, ce qui engendre bien évidemment de légers glissements. De même, alors qu'il est raisonnable de dire que des roues pleines sont indéformables, cette hypothèse est largement fausse avec des roues équipées de pneus.

Malgré cela, on supposera toujours qu'il y a r.s.g. et, par ailleurs, que le sol est parfaitement plan.

Mathématiquement, on peut traduire la condition de r.s.g. sur une roue. Soit *P* le

centre de la roue, *Q* le point de contact de la roue avec le sol, ω l'angle de rotation propre de la roue et Θ l'angle entre le plan de la roue et le plan (O, $\vec{x}$, $\vec{z}$). Comme indiqué à la Figure .I.8. La nullité de la vitesse relative $\overrightarrow{vq}$ roue/sol au point de contact permet d'obtenir

une relation vectorielle entre la vitesse $\overrightarrow{vp}$ du centre *P* de la roue et le vecteur vitesse de rotation $\vec{w}$ de la roue :

$$\overrightarrow{vq} = \overrightarrow{vp} + \vec{w} \wedge \overrightarrow{PQ} = \vec{O} \qquad (1.3)$$

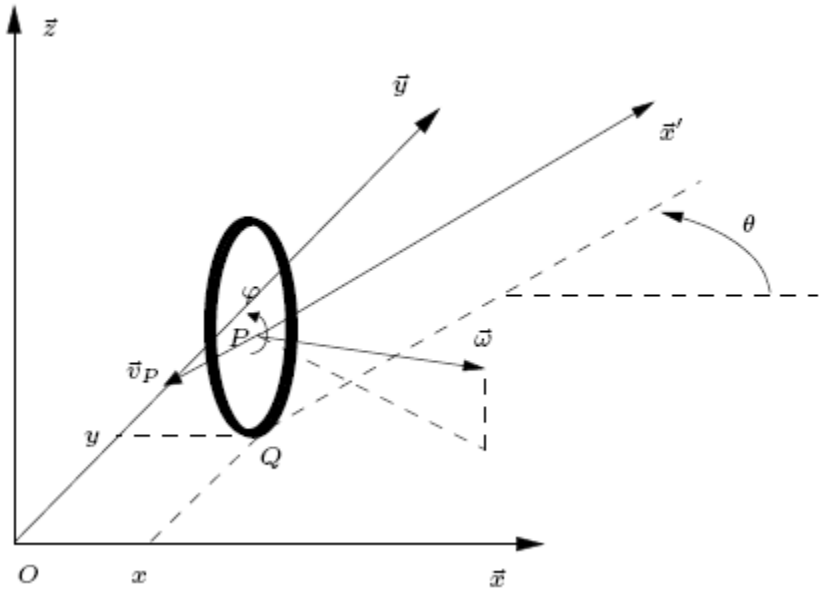


FIG.I.8 : Caractérisation du roulement sans glissement

Les points P et Q ont pour coordonnées respectives $(x\ y\ r)^T$ et $(x\ y\ 0)^T$. Il vient alors :

$$\dot{x}\,\vec{x} + \dot{y}\vec{y} + (\dot{\theta}\vec{z} + \dot{\varphi}\,(\sin\theta\vec{x} - \cos\theta\vec{y})) \wedge (-r\vec{z}) = \vec{O}$$

$$(\dot{x} + r\,\dot{\varphi}\cos\theta)\,\vec{x} + (\dot{y} + r\,\dot{\varphi}\sin\theta)\,\vec{y} = \vec{O}$$

Ceci nous donne le système de contraintes scalaires :

$$\dot{x} + r\dot{\varphi}\cos\theta = 0 \qquad (1.4)$$

$$\dot{y} + r\dot{\varphi}\sin\theta = 0 \qquad (1.5)$$

Que l'on peut transformer pour faire apparaitre les composantes de vitesse dans le plan de

la roue d'une part et perpendiculairement à la roue d'autre part :

$$-\dot{x}\sin\theta + \dot{y}\cos\theta = 0 \qquad (1.6)$$

$$\dot{x}\cos\theta + \dot{y}\sin\theta = -r\dot{\varphi} \qquad (1.7)$$

Ces contraintes traduisent le fait que le vecteur $\vec{v}P$ soit dans le plan de la roue et ait pour module $r\dot{\varphi}$.

I.7.4. Contraintes non holonomes :

Les équations précédentes, caractérisant le r.s.g. d'une roue sur le sol, sont des contraintes non holonomes. Nous nous proposons dans ce paragraphe de préciser ce que recouvre ce terme et de caractériser les systèmes non holonomes.

Soit un système de configuration q soumis à des contraintes indépendantes sur les vitesses, regroupées sous la forme $A^T(q)\dot{q} = 0$. S'il n'est pas possible d'intégrer l'une de ces contraintes, elle est dite non intégrable ou non holonome. De manière concrète l'existence de contraintes non holonomes implique que le système ne peut pas effectuer certains mouvements instantanément. Par exemple, dans le cas de la roue, il ne peut y avoir de translation instantanée parallèlement à l'axe de la roue. Un tel déplacement nécessitera des manoeuvres. De même, comme on le sait bien, une voiture ne peut se garer facilement sans effectuer de créneaux.

Il n'est pas évident de dire a priori si une contrainte est intégrable ou non. Pour cela, on a recours à l'application du théorème de Frobenius, dont une version complète pourra être trouvée dans un ouvrage de référence de géométrie différentielle ou de commande non-linéaire . Seule la connaissance du crochet de Lie est nécessaire à notre étude. Pour deux vecteurs bi(q) et bj(q), cet opérateur est défini par

$$[\, bi(q), bj(q)\,] = \frac{\partial bj}{\partial q}\, bi - \frac{\partial bi}{\partial q}\, bj \qquad (1.8)$$

I.7.5. Disposition des roues :

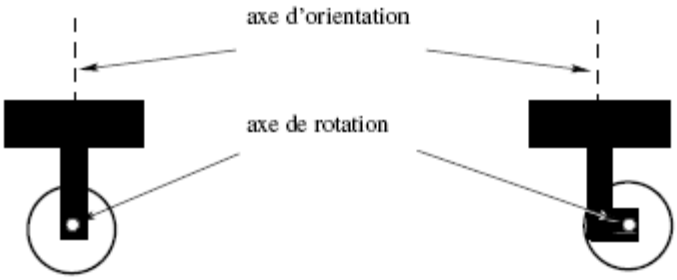
C'est la combinaison du choix des roues et de leur disposition qui confère à un robot son mode de locomotion propre. Sur les robots mobiles, on rencontre principalement trois types de roues (voir Figure .I.9) :

-les roues fixes dont l'axe de rotation, de direction constante, passe par le centre de la

roue

-roues centrées orientables, dont l'axe d'orientation passe par le centre de la roue

-les roues décentrées orientables, souvent appelées roues folles, pour lesquelles l'axe d'orientation ne passe pas par le centre de la roue.



(a) Roues fixes (axe d'orientation fixe) et roues centrées orientables (b) Roues décentrées orientables

FIG.I.9 : Les principaux types de roues des robots mobiles

De manière on rencontrera aussi des systèmes particuliers, tels que les roues suédoises, 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 occasionner d'éventuels blocages. 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 trois principales catégories de robots mobiles à roues, que l'on va présenter maintenant.

I.7.6. Centre instantané de rotation :

Les roues motrices ayant même axe de rotation, le CIR du robot est un point de cet axe. Soit p le rayon de courbure de la trajectoire du robot, c'est-à-dire la distance du CIR au point O (voir Figure .I.10). Soit L l'entre-axe et w la vitesse de rotation du robot

autour du CIR. Alors les vitesses des roues droite et gauche, respectivement notées v_d et v_g et définies à la Figure .I.10 vérifient :

$$v_d = -r\dot{\varphi}d = (p + L)w \qquad (1.9)$$

$$v_g = r\dot{\varphi}g = (p - L)w \qquad (1.10)$$

ce qui permet de déterminer p et w à partir des vitesses des roues :

$$p = L\frac{\dot{\varphi}d-\dot{\varphi}g}{\dot{\varphi}d+\dot{\varphi}g} \qquad (1.11)$$

$$w = -\frac{r(\dot{\varphi}d-\dot{\varphi}g)}{2l} \qquad (1.12)$$

L'équation (1.11) permet de situer le CIR sur l'axe des roues. Par ailleurs ces équations expliquent deux propriétés particulières du mouvement des robots de type unicycle : si $\dot{\varphi}d = -\dot{\varphi}g$, le robot se déplace en ligne droite ; si $\dot{\varphi}d = \dot{\varphi}g$, alors le robot effectue un rotation sur lui-même. L'utilisation de ces deux seuls modes de locomotion, bien que limitée, permet de découpler les mouvements et de fournir une solution simple pour amener le robot d'une posture à une autre. C'est sans doute là une des raisons du succès de ce type de robots. Pour élaborer une stratégie plus fine de déplacement, il est cependant intéressant de savoir comment la posture du robot est reliée à la commande de ses roues.

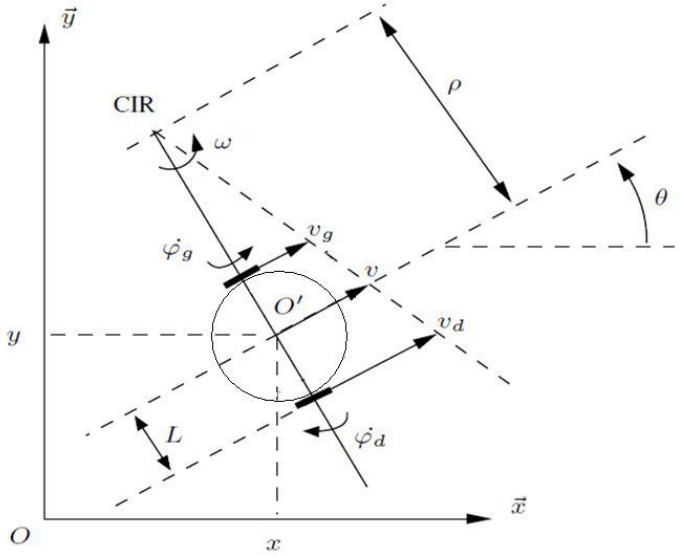


FIG.I.10 : Centre instantané de rotation d'un robot de type unicycle

I.7.7. Choix de la commande :

En ce qui concerne la commande, si l'on se contente de traiter le cas cinématique, on peut considérer que celle-ci est donnée, au plus bas niveau, par les vitesses de rotation des roues. Ceci étant, on préfère généralement exprimer cette commande par la vitesse longitudinale du robot, notée v (en $\hat{O}$) et sa vitesse de rotation $\dot{\theta}$ (autour de $\hat{O}$). Il y a en effet équivalence entre les deux représentations. D'une part, on a :

$$v = \frac{vd + vg}{2} = \frac{r(\dot{\varphi}d - \dot{\varphi}g)}{2} \qquad (1.13)$$

D'autre part, la vitesse de rotation du robot est égale à la vitesse de rotation autour du CIR :

$$w = \dot{\theta} = -\frac{r(\dot{\varphi}d - \dot{\varphi}g)}{2l} \qquad (1.14)$$

Conformément à l'équation (1.12). On montre que ces relations sont parfaitement inversibles et qu'il y a ainsi équivalence entre les couples ($\dot{\varphi}$d; $\dot{\varphi}$g) et (v, w). Désormais, on utilise plut$\hat{o}$t ce dernier couple de grandeurs, plus parlantes, quitte à calculer ensuite les angles ou vitesses de consigne des asservissements des roues.

I.7.8. Modèle cinématique en posture :

Relier la dérivée de la posture à la commande u = $(v\ w)^T$ est facile. Une simple considération géométrique donne :[01]

$$\dot{x} = v \cos\theta \qquad (1.15)$$

$$\dot{y} = v \sin\theta \qquad (1.16)$$

$$\dot{\theta} = w \qquad (1.17)$$

soit : $$\dot{\xi} = \begin{pmatrix} cos\theta & 0 \\ sin\theta & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} v \\ w \end{pmatrix} \qquad (1.18)$$

I.7.9. Mouvement admissible avec contrôle d'orientation :

On exprime l'erreur de posture ξ_e dans le repère R_r du robot entre posture courante et posture de référence. [01]

$$\xi_e = \begin{pmatrix} x_e \\ y_e \\ \theta_e \end{pmatrix} = \begin{pmatrix} cos\theta_r & sin\theta_r & 0 \\ -sin\theta_r & cos\theta_r & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x - x_e \\ y - y_e \\ \theta - \theta_e \end{pmatrix} \qquad (1.19)$$

I.8. Conclusion :

Nous avons présenté dans ce chapitre des généralités importantes sur la robotique mobile , aperçu historique, la définition, les application et les avantages et les présentation des robots mobiles , et on a citée la modalisation de robot KHEPERA (définition, roulement ,contrainte non holonomes, model cinématique en posture,…….etc.) .

il faut présenter la méthode d'optimisation (les algorithmes génétiques), Les détaille , les principes, et les différents techniques de cette méthode sera l'objectif du chapitre suivant.

II.1.Introduction :

Un algorithme génétique recherche le ou les extrema d'une fonction définie sur un espace de données. Pour l'utiliser, on doit disposer les cinq éléments suivants :

1) Un principe de codage de l'élément de population. Cette étape associe à chacun des points de l'espace d'état une structure de données. Elle se place généralement après une phase de modélisation mathématique du problème traité. La qualité du codage des données conditionne le succès des algorithmes génétiques. Le codages binaires ont été très utilisés à l'origine. Les codages réels sont désormais largement utilisés, notamment dans les domaines applicatifs pour l'optimisation de problèmes à variables réelles.

2) Un mécanisme de génération de la population initiale. Ce mécanisme doit être capable de produire une population d'individus non homogène qui servira de base pour les générations futures.

Le choix de la population initiale est important car il peut rendre plus ou moins rapide la convergence vers l'optimum global. Dans le cas où l'on ne connaît rien du problème à résoudre, il est essentiel que la population initiale soit répartie sur tout le domaine de recherche.

3) Une fonction à optimiser. Celle-ci retourne une valeur appelée fitness ou fonction d'évaluation de l'individu.

4) Des opérateurs permettant de diversifier la population au cours des générations et d'explorer l'espace d'état. L'opérateur de croisement recompose les gènes d'individus existant dans la population, l'opérateur de mutation a pour but de garantir l'exploration de l'espace d'états.

5) Des paramètres de dimensionnement : taille de la population, nombre total de générations ou critère d'arrêt, probabilités d'application des opérateurs de croisement et de mutation.

II.2. Pourquoi les algorithme génétique ?:

Les algorithmes génétiques fournissent des solutions aux problèmes n'ayant pas de solutions calculables en temps raisonnable de façon analytique ou algorithmique.

Selon cette méthode, des milliers de solutions (génotypes) plus ou moins bonnes sont crées au hasard puis sont soumises à un procédé d'évaluation de la pertinence de la solution mimant l'évolution des espèces : les plus "adaptés", c'est-à-dire les solutions au problème qui sont les plus optimales survivent davantage que celles qui le sont moins et la population évolue par générations successives en croisant les meilleures solutions entre elles et en les faisant muter, puis en relançant ce procédé un certain nombre de fois afin d'essayer de tendre vers la solution optimale.

Le mécanisme d'évolution et de sélection est indépendant du problème à résoudre : seules changent trois fonctions :

• la fonction qui s'occupe de représenter le problème en codant chaque information caractérisant une solution possible selon un codage bien particulier, chaque information représente alors un gène et toutes les valeurs que peuvent prendre cette caractéristique représentent les allèles possibles pour ce gène, et en concaténant tous ces gènes pour obtenir un chromosome qui lui représente une solution dans son intégralité

• la fonction inverse qui à partir d'un chromosome permet d'obtenir une solution par décodage du génome.

• la fonction qui évalue l'adaptation d'une solution à un problème, sa pertinence.

Cette technique est d'application générale.

En effet, quand on utilise les algorithmes génétiques, aucune connaissance de la manière dont résoudre le problème n'est requise, il est seulement nécessaire de fournir une fonction permettant de coder une solution sous forme de gènes (et donc de faire le travail inverse).

ainsi que de fournir une fonction permettant d'évaluer la pertinence d'une solution au problème donné.

Cela en fait donc un modèle minimal et canonique pour n'importe quel système évolutionnaire et pour n'importe quel problème pouvant être abordé sous cet angle, sous ce paradigme.

Cette représentation nous permet donc d'étudier des propriétés quasiment impossibles à étudier dans leur milieu naturel, ainsi que de résoudre des problèmes n'ayant pas de solutions calculables en temps raisonnables si on les aborde sous d'autres paradigmes, avec des performances quantifiables, facilement mesurables et qu'on peut confrontés aux autres stratégies de résolution.

II.3. Domaine d'utilisation :

Les algorithmes génétiques peuvent être particulièrement utiles dans les domaines suivants :

• Optimisation : optimisation de fonctions, planification, ...etc.

• Apprentissage : classification, prédiction, robotique, ...etc.

• Programmation automatique : programmes LISP, automates cellulaires, ...etc.

• Etude du vivant, du monde réel : marchés économiques, comportements sociaux, systèmes immunitaires, ...etc.

II.4. Forme et fonctionnent des algorithmes génétiques :

Les principales différences des algorithmes génétiques par rapport aux autres paradigmes sont les suivantes :

• On utilise un codage des informations : on représente toutes les caractéristiques d'une solution par un ensemble de gènes, c'est-à-dire un chromosome, sous un certain codage (binaire, réel, code de Gray, etc. ...), valeurs qu'on concatène pour obtenir une chaîne de caractères qui est spécifique à une solution bien particulière (il y a une bijection entre la solution et sa représentation codée

• On traite une population "d'individus", de solutions : cela introduit donc du parallélisme.

• L'évaluation de l'optimalité du système n'est pas dépendante vis-à-vis du domaine.

• On utilise des règles probabilistes : il n'y a pas d'énumération de l'espace de recherche, on en explore une certaine partie en étant guidé par un semi-hasard : en effet des opérateurs comme la fonction d'évaluation permet de choisir de s'intéresser à

une solution qui semble représenter un optimum local, on fait donc un choix délibéré, puis de la croiser avec une autre solution optimale localement, en général la solution obtenue par croisement est meilleure ou du même niveau que ses parents, mais ce n'est pas assuré, cela dépend des aléas du hasard, et cela et d'autant plus vrai pour l'opérateur de mutation qui ne s'applique qu'avec une certaine probabilité et dans le cas où il s'applique choisit aléatoirement sur quel(s) locus(loci) introduire des modifications.

Un algorithme génétique générique à la forme suivante :

1) Initialiser la population initiale P.

2) Evaluer P.

3) Tant Que (Pas Convergence) faire :

a) P ' = Sélection des Parents dans P

b) P ' = Appliquer Opérateur de Croisement sur P '

c) P ' = Appliquer Opérateur de Mutation sur P '

d) P = Remplacer les Anciens de P par leurs Descendants de P '

e) Evaluer P

Fin TantQue.

Le critère de convergence peut être de nature diverse, par exemple :

• Un taux minimum qu'on désire atteindre d'adaptation de la population au problème.

• Un certain temps de calcul à ne pas dépasser.

• Une combinaison de ces deux points.

II.5. Variantes :

Beaucoup d'algorithme basé sur des idées forment l'algorithme génétique principal

II.5.1 Le codage :

Chaque paramètre d'une solution est assimilé à un gène, toutes les valeurs qu'il peut prendre sont les allèles de ce gène, on doit trouver une manière de coder chaque

allèle différent de façon unique (établir une bijection entre l'allèle "réel" et sa représentation codée).

Un chromosome est une suite de gène, on peut par exemple choisir de regrouper les paramètres similaires dans un même chromosome (chromosome à un seul brin) et chaque gène sera repérable par sa position : son locus sur le chromosome en question.

Chaque individu est représenté par un ensemble de chromosomes, et une population est un ensemble d'individus.

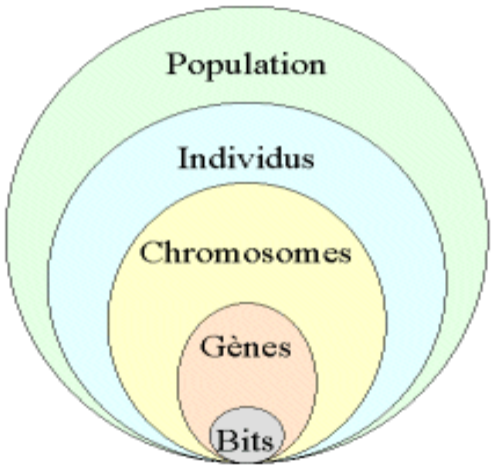


FIG.II.1 : les cinq niveaux d'organisation d'un algorithme génétique

Il y a trois principaux types de codage utilisables, et on peut passer de l'un à l'autre relativement facilement :

II.5.1.a. Le codage binaire :

C'est le plus utilisé. Chaque gène dispose du même alphabet binaire {0, 1} Un gène est alors représenté par un entier long (32 bits), les chromosomes qui sont des suites de gènes sont représentés par des tableaux de gènes et les individus de notre espace de recherche sont représentés par des tableaux de chromosomes.

Ce cas peut être généralisé à tout alphabet allélique n-aire permettant un codage plus intuitif, par exemple pour le problème du voyageur de commerce on peut préférer utiliser l'alphabet allélique {c1, c2, c3, ..., cn} où ci représente la ville de numéro i.•

II.5.1.b. Le codage réel :

Cela peut-être utile notamment dans le cas où l'on recherche le maximum d'une fonction réelle.

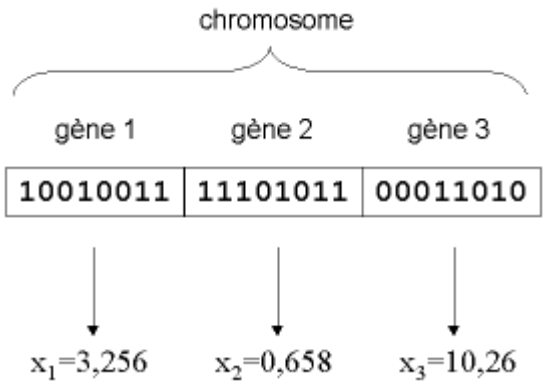


FIG.II.2 : illustration schématique du codage des variables réelles

II.5.1.c. Le codage de gray :

Dans le cas d'un codage binaire on utilise souvent la "distance de Hamming" comme mesure de la dis similarité entre deux éléments de population, cette mesure compte les différences de bits de même rang de ces deux séquences.

Et c'est la que le codage binaire commence à montrer ses limites. En effet, deux éléments voisins en termes de distance de Hamming ne codent pas nécessairement deux éléments proches dans l'espace de recherche. Cet inconvénient peut être évité en utilisant un "codage de Gray" : le codage de Gray est un codage qui a comme propriété que entre un élément n et un élément n + 1, donc voisin dans l'espace de recherche, un seul bit diffère.

II.5.2. Les opérateurs :

II.5.2.1. L'opérateur de sélection :

Cet opérateur est chargé de définir quels seront les individus de P qui vont être dupliqués dans la nouvelle population P' et vont servir de parents (application de l'opérateur de croisement).

Soit *n* le nombre d'individus de P, on doit en sélectionner *n/2* (l'opérateur de croisement nous permet de repasser à *n* individus).

Cet opérateur est peut-être le plus important puisqu'il permet aux individus d'une population de survivre, de se reproduire ou de mourir. En règle générale, la probabilité

de survie d'un individu sera directement reliée à son efficacité relative au sein de la population.

On trouve essentiellement quatre types de méthodes de sélection différentes :

• La méthode de la "loterie biaisée" (roulette wheel) de GoldBerg,

• La méthode "élitiste",

• La sélection par tournois,

• La sélection universelle stochastique.

II.5.2.1.a. La loterie biaisée ou roulette wheel :

Cette méthode est la plus connue et la plus utilisée.

Avec cette méthode chaque individu a une chance d'être sélectionné proportionnelle à sa performance, donc plus les individus sont adaptés au problème, plus ils ont de chances d'être sélectionnés.

Pour utiliser l'image de la "roue du forain", chaque individu se voit attribué un secteur dont l'angle est proportionnel à son adaptation, sa "fitness".

On fait tourner la roue et quand elle cesse de tourner on sélectionne l'individu correspondant au secteur désigné par une sorte de "curseur", curseur qui pointe sur un secteur particulier de celle-ci après qu'elle se soit arrêté de tourner.

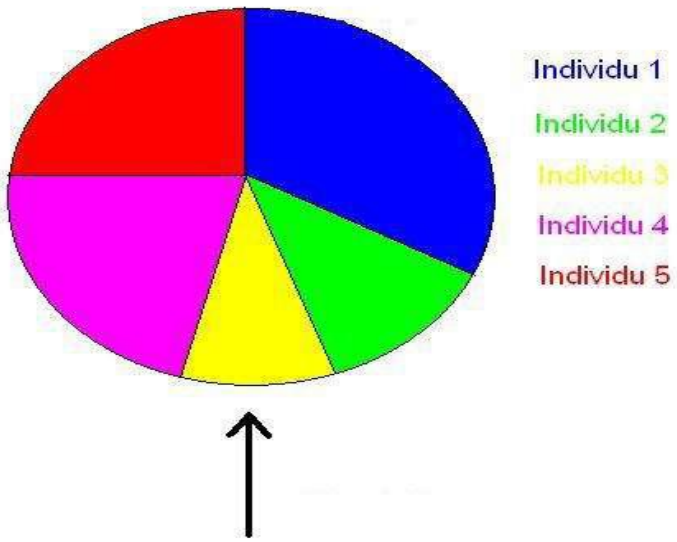


FIG.II.3 : la méthode de sélection de la loterie biaisée

Cette méthode, bien que largement répandue, a pas mal d'inconvénients :

• En effet, elle a une forte variance. Il n'est pas impossible que sur n sélections successives destinées à désigner les parents de la nouvelle génération P', la quasi-totalité, voire pire la totalité des n individus sélectionnés soient des individus ayant une fitness vraiment mauvaise et donc que pratiquement aucun individu voire aucun individu a forte fitness ne fasse partie des parents de la nouvelle génération. Ce phénomène est bien sûr très dommageable car cela va complètement à l'encontre du principe des algorithmes génétiques qui veut que les meilleurs individus soient sélectionnés de manière à converger vers une solution la plus optimale possible.

• A l'inverse, on peut arriver à une domination écrasante d'un individu "localement supérieur".

Ceci entrainant une grave perte de diversité. Imaginons par exemple qu'on ait un individu ayant une fitness très élevée par rapport au reste de la population, disons dix fois supérieure, il n'est pas impossible qu'après quelques générations successives on se retrouve avec une population ne contenant que des copies de cet individu. Le problème est que cet individu avait une fitness très élevée, mais que cette fitness était toute relative, elle était très élevée mais seulement en comparaison des autres individus. On se retrouve donc face à problème.

connu sous le nom de "convergence prématurée; l'évolution se met donc à stagner et on atteindra alors jamais l'optimum, on restera bloqué sur un optimum local.

Il existe certaines techniques pour essayer de limiter ce phénomène, comme par exemple le "scaling", qui consiste à effectuer un changement d'échelle de manière à augmenter ou diminuer ladé manière forcée a fitness d'un individu par rapport à un autre selon leur écart de fitness.

Malgré tout, il est conseillé d'opter plutôt pour une autre méthode de sélection.

II.5.2.1.b. La méthode élitiste :

Cette méthode consiste à sélectionner les n individus dont on a besoin pour la nouvelle génération P' en prenant les n meilleurs individus de la population P après l'avoir triée de manière décroissante selon la fitness de ses individus.

Il est inutile de préciser que cette méthode est encore pire que celle de la loterie biaisée dans le sens où elle amènera à une convergence prématurée encore plus rapidement et surtout de manière encore plus sûre que la méthode de sélection de la loterie biaisée ; en effet, la pression de la sélection est trop forte, la variance nulle et la diversité inexistante, du moins le peu de diversité qu'il pourrait y avoir ne résultera pas de la sélection mais plutôt du croisement et des mutations.

Là aussi il faut opter pour une autre méthode de sélection.

II.5.2.1.c. La sélection par tournois :

Cette méthode est celle avec laquelle on obtient les résultats les plus satisfaisants.

Le principe de cette méthode est le suivant : on effectue un tirage avec remise de deux individus de P, et on les fait "combattre". Celui qui a la fitness la plus élevée l'emporte avec une probabilité p comprise entre 0.5 et 1. On répète ce processus n fois de manière a obtenir les n individus de P' qui serviront de parents.

La variance de cette méthode est élevée et le fait d'augmenter ou de diminuer la valeur de *p* permet respectivement de diminuer ou d'augmenter la pression de la sélection.

II.5.2.1.d. La sélection universelle stochastique :

Cette méthode semble être très peu utilisée et qui plus est possède une variance faible, donc introduit peu de diversité, nous n'entrerons donc pas dans les détails, on se contentera d'exposer sa mise en oeuvre :

On prend l'image d'un segment decoupé en autant de sous-segments qu'il y a d'individus. Les individus sélectionnés sont désignés par un ensemble de points équidistants.

II.5.2.2. L'opérateur de croisement ou crossover :

Le crossover utilisé par les algorithmes génétiques est la transposition informatique du mécanisme qui permet, dans la nature, la production de chromosomes qui héritent partiellement des caractéristiques des parents.

Son rôle fondamental est de permettre la recombinaison des informations présentes dans le patrimoine génétique de la population.

Cet opérateur est appliqué après avoir appliqué l'opérateur de sélection sur la population P; on se retrouve donc avec une population P' de *n/2* individus et on doit doubler ce nombre pour que notre nouvelle génération soit complète.

On va donc créer de manière aléatoire *n/4* couples et on les fait se "reproduire".

Les chromosomes (ensembles de paramètres) des parents sont alors copiés et recombinés de façon à former deux descendants possédant des caractéristiques issues des deux parents.

Détaillons ce qui se passe pour chaque couple au niveau de chacun de leurs chromosomes :

Un, deux, voire jusqu'à *lg - 1* (où lg est la longueur du chromosome) points de croisements (loci) sont tirés au hasard, chaque chromosome se retrouve donc séparé en "segments". Puis chaque segment du parent 1 est échangé avec son "homologue" du parent 2 selon une probabilité de croisement *pc* . De ce processus résulte 2 fils pour chaque couple et notre population P' contient donc bien maintenant *n* individus.

On peut noter que le nombre de points de croisements ainsi que la probabilité de croisement *pc* permettent d'introduire plus ou moins de diversité.

En effet, plus le nombre de points de croisements sera grand et plus la probabilité de croisement sera élevée plus il y aura d'échange de segments, donc d'échange de paramètres, d'information, et plus le nombre de points de croisements sera petit et plus la probabilité de croisement sera faible, moins le croisement apportera de diversité.

Ci-dessous, un schéma illustrant un croisement en un point, un autre pour un croisement en deux points, et enfin un schéma représentant un croisement avec *lg - 1* points de croisements (on notera d'ailleurs sur ce schéma que l'échange d'un segment avec son homologue ne se fait pas toujours) :

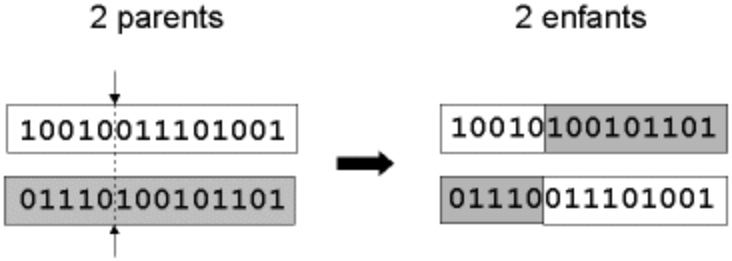


FIG.II.4 : croisement avec un point de crossover

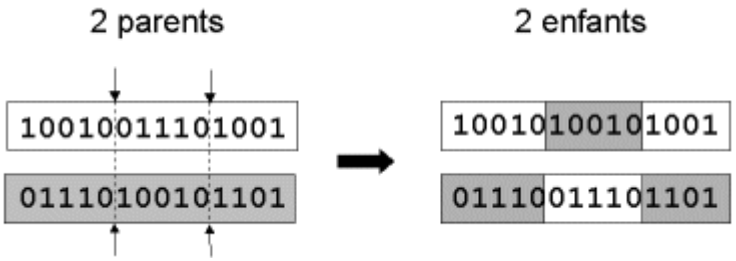


FIG.II.5 : croisement avec 2 points de crossover

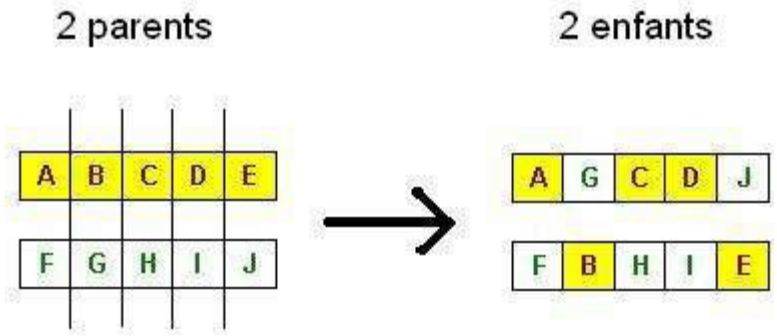


FIG.II.6 : croisement uniforme

On peut citer aussi une autre méthode très utilisée dans le cas des problèmes modélisés par un codage binaire, il s'agit du **croisement uniforme**. La mise en oeuvre de ce procédé est fort simple, elle consiste à définir de manière aléatoire un "masque", c'est-à-dire une chaîne de bits de même longueur que les chromosomes des parents sur lesquels il sera appliqué. Ce masque est destiné à savoir, pour chaque locus, de quel parent le premier fils devra hériter du gène s'y trouvant; si face à un locus le masque présente un 0, le fils héritera le gène s'y trouvant du parent n° 1, si il présente un 1 il en héritera du parent n° 2. La création du fils n° 2 se fait de manière symétrique : si pour un gène donné le masque indique que le fils n° 1 devra recevoir celui-ci du parent n° 1 alors le fils n° 2 le recevra du parent n°2, et si le fils n° 1 le reçoit du parent n° 2 alors le fils 2 le recevra du parent n° 1.

L'opérateur de croisement favorise l'exploration de l'espace de recherche. En effet, considérons deux gènes A et B pouvant être améliorés par mutation. Il est peu

probable que les deux gènes améliorés A’ et B’ apparaissent par mutation dans un même individu. Mais si un parent porte le gène mutant A' et l'autre le gène mutant B', l’opérateur de croisement permettra de combiner rapidement A’ et B’ et donc de créer un nouvel individu possédant cette combinaison, combinaison grâce à laquelle il est possible qu'il soit encore plus adapté que ses parents.

L’opérateur de croisement assure donc le brassage du matériel génétique et l’accumulation des mutations favorables. En termes plus concrets, cet opérateur permet de créer de nouvelles combinaisons des paramètres des composants.

Malgré tout, il est possible que l'action conjointe de la sélection et du croisement ne permette pas de converger vers la solution optimale du problème.

II.5.2.3. L'opérateur de mutation :

Cet opérateur consiste à changer la valeur allélique d'un gène avec une probabilité *pm* très faible, généralement comprise entre 0.01 et 0.001.

On peut aussi prendre *pm = 1 / lg* où l*g* est la longueur de la chaîne de bits codant notre chromosome.

Une mutation consiste simplement en l'inversion d'un bit (ou de plusieurs bits, mais vu la probabilité de mutation c'est extrêmement rare) se trouvant en un locus bien particulier et lui aussi déterminé de manière aléatoire; on peut donc résumer la mutation de la façon suivante :

On utilise une fonction censée nous retourner *true* avec une probabilité *pm* .

Pour chaque locus **faire**
    Faire appel à la fonction
    Si cette fonction nous renvoie *true* **alors**
        on inverse le bit se trouvant à ce locus
    FinSi
FinPour

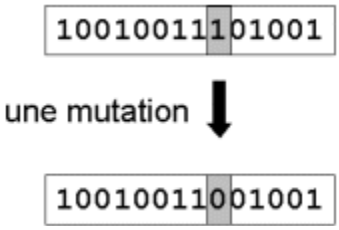


FIG.II.7 : une mutation

L'opérateur de mutation modifie donc de manière complètement aléatoire les caractéristiques d'une solution, ce qui permet d'introduire et de maintenir la diversité au sein de notre population de solutions. Cet opérateur joue le rôle d'un "élément perturbateur", il introduit du "bruit" au sein de la population.

Cet opérateur dispose de 4 grands avantages :

• Il garantit la diversité de la population, ce qui est primordial pour les algorithmes génétiques.

• Il permet d'éviter un phénomène connu sous le nom de **dérive génétique**. On parle de dérive génétique quand certains gènes favorisés par le hasard se répandent au détriment des autres et sont ainsi présents au même endroit sur tous les chromosomes. Le fait que l'opérateur de mutation puisse entraîner de manière aléatoire des changements au niveau de n'importe quel locus permet d'éviter l'installation de cette situation défavorable.

• Il permet de limiter les risques d'une convergence prématurée causée par exemple par une méthode de sélection élitiste imposant à la population une pression sélective trop forte. En effet, dans le cas d'une convergence prématurée on se retrouve avec une population dont tous les individus sont identiques mais ne sont que des optimums locaux. Tous les individus étant identiques, le croisement ne changera rien à la situation. En effet, l'échange d'informations par crossover entre des individus strictement identiques est bien sûr totalement sans conséquences; on aura beau choisir la méthode de croisement qu'on veut on se retrouvera toujours à échanger des portions

de chromosomes identiques et la population n'évoluera pas. L'évolution se retrouvant bloquée on n'attendra jamais l'optimum global.

La mutation entrainant des inversions de bits de manière aléatoire permet de réintroduire des différences entre les individus et donc de nous extirper de cette situation.

Il est quand même utile de garder à l'esprit que ceci n'est pas une solution "miracle" et qu'il est bien entendu plus intelligent de ne pas utiliser de méthodes de sélection connues pour entrainer ce type de problème.

• La mutation permet d'atteindre la propriété d'ergodicité.

L'ergodicité est une propriété garantissant que chaque point de l'espace de recherche puisse être atteint.

En effet, une mutation pouvant intervenir de manière aléatoire au niveau de n'importe quel locus, on a la certitude mathématique que n'importe quel permutation de notre chaîne de bits.

peut apparaître au sein de la population et donc que tout point de l'espace de recherche peut être atteint.

Grâce à cette propriété on est donc sûr de pouvoir atteindre l'optimum global.

On notera que la mutation règle donc le problème exposé à la fin du Section sur le croisement. [16]

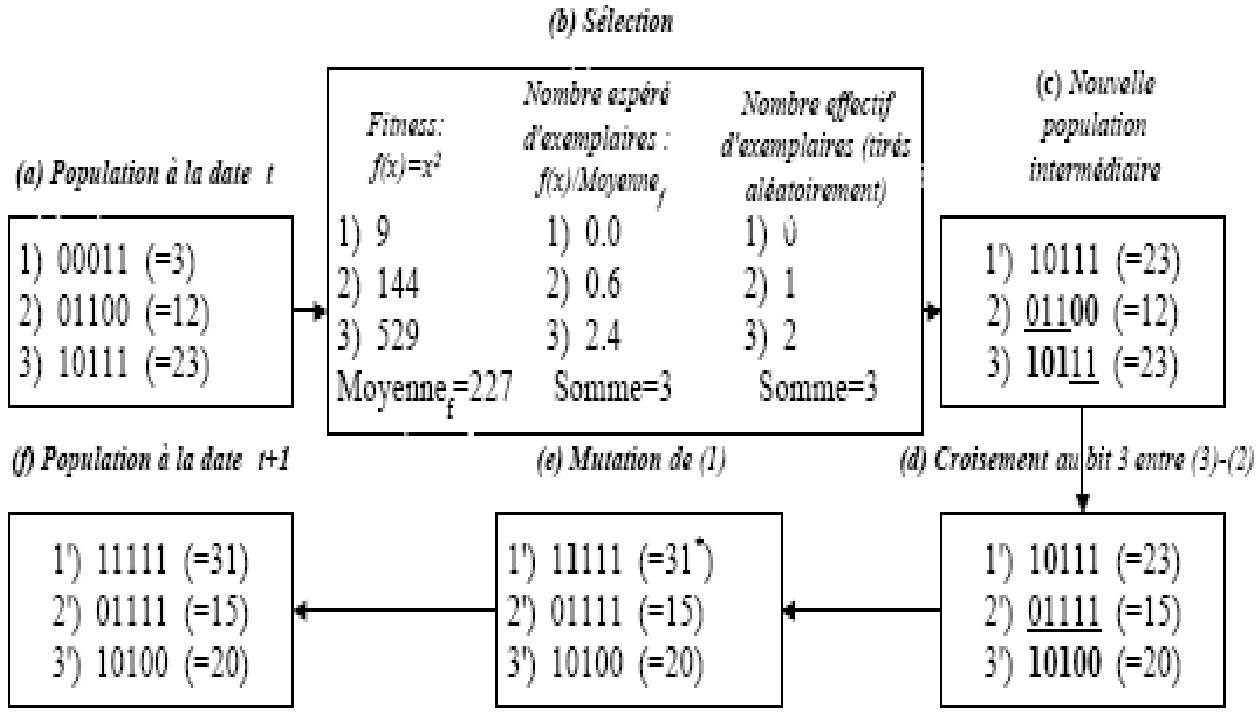


FIG.II.8 : Un exemple simple

Une application des AG à l'optimisation de la fonction f(x)=x2 sur le domaine formé par des entiers entre 0 et 31.Les entiers sont codés avec une chaîne binaire de cinq bits : 00001=1, 11111=31. L'exemple utilise une population initiale aléatoire de trois individus (chromosomes) et l'AG construit une nouvelle population à travers une reproduction sélective(**étape (b)**), des croisements (combinaison de solutions existantes - **étape (d)**) et des mutations (expériences aléatoires, introduction de la nouveauté - **étape (e)**). L'AG atteint en une étape la solution optimale (x=31) dans cet exemple schématique. Pour chaque chaîne, le croisement, sa position et le partenaire impliqué sont déterminés de manière aléatoire. La mutation simplement change la valeur du bit concerné : de 0 vers 1 ou de 1 vers 0. Ce processus est contrôlé par : la taille de la population, la taille des chaînes binaires et les probabilités de croisement et de mutation. [11]

Conclusion :

Nous avons présenté dans ce chapitre les applications des' algorithmes génétiques sont multiples : forme et fonctionnent, variant (codage et opération) etc.

Nos exemples d'application nous ont permis de nous rendre compte que le codage des données pour modéliser un problème est complexe. D'autre part, nous sommes aussi aperçus des difficultés pour choisir pertinemment de bons paramètres pour les divers opérateurs (mutation, croisement, sélection), Des choix par rapport aux opérateurs eux mêmes sont aussi à gérer, sachant que certains sont plus appropriés au problème et qu'ils permettent d'optimiser.

Les algorithmes génétiques seuls ne sont pas très efficaces dans la résolution d'un problème. Ils apportent cependant assez rapidement une solution acceptable. Néanmoins, il est possible de l'améliorer assez efficacement en le combinant avec un algorithme déterministe.

III.1. Introduction :

Nous présentons dans ce chapitre la description détaillée avons utilisé de notre simulateur et les résultat obtenu pour chaque commande.

III.2.Présentation des environnements :

Dans notre travail nous avons créé plusieurs environnements tel que les environnement libre et contraignant ,dans l'environnement libre en utilise une segment de droite comme trajectoire de référence sans l'utilisation des algorithmes génétique et dans les environnement contraignant en utilise plusieurs chemins come trajectoire de référence avec et sans l'utilisation des algorithmes génétiques pour la comparaison des trajectoires de robot et de référence ainsi que la minimisation de l'erreur entre les deus trajectoire .

III.3.Optimisation par les algorithmes génétiques :

III.3.1. Le codage :

Le codage est représente sous forme de chaîne de bits (0, 1) contenant toute les' information nécessaire pour la description d'un point dans l'espace d''etat. La longueur de chaque gène est le nombre minimal de bits que nous permet de coder tous les' erreurs qui se trouvent dans l'environnement.

Et chaque individu contient un nombre minimal de bit de gène multiplié par le nombre minimal des coins que le robot peut les passe.

III.3.2. La fonction principal de l'algorithme génétique :

La réalisation de la fonction de l'algorithme génétique consiste à développer tous les fonctions utilisées dans notre fonction principale suivante :

Algorithme génétique (N=taille_population, K=taille_d'individu, nb_génération,pc,pm)

Début

Initialisation de N individus de K bit ;

Pour g=1 à nb_génération

Générer une nouvelle population.

Appliquer l'opérateur de sélection.

Appliquer l'opérateur de croisement pc.

Appliquer l'opérateur de mutation pm.

Appliquer l'opérateur d'insertion de conditions dans une règle.

Appliquer l'opérateur de suppression de conditions d'une règle.

Appeler la fonction Calcule_Fitness.

Fin pour.

Fin.

Pour la première génération : la population initiale, les individus sont généralement créés au hasard. Une fois que les individus générés, il est nécessaire d'évaluer chacune d'elles. Une fonction d'évaluation (FE ou fitness function) doit donc être définie.

Ainsi, plus la valeur de la FE d'un individu est élevée, plus ses allèles (valeur de ses gènes) sont pertinents. Cette évaluation se fait le plus souvent à l'aide de plusieurs critères.

Dans un deuxième temps, les couples d'individus (parents) qui pourront se reproduire sont sélectionnés.

Les principes généraux de croisement et de mutation sont les suivants : pour le croisement deux individus sont sélectionnés suivant la probabilité **pc** .ces deux parents seront donc compilés pour générer deux enfants. Ils peuvent être combinés par des croisements mono ou multipoints. Dans tous les cas, les enfants posséderont une partie des gènes d'un parent et une partie de l'autre. Pour la mutation, un individu est sélectionné, un ou plusieurs de ses gènes est transformé suivant la probabilité **pm**.

III.3.2.1. L'initialisation :

La population initiale sera composée des N individus générées aléatoirement.

Chaque individu contient K bits.

```
Fonction initialisation (N=taille_population, K=taille_d'individu)
    Début
        Pour i=1 a N faire
            Pour j=1 a K faire
                Parent_i [i, j]=Générer aléatoirement 1 ou 0 ;
            Fin pour
        Fin pour
    Fin
```

III.3.2.2. La procédure d'évaluation :

Pour l'évaluation de chaque individu on utilise une fonction qui calcul la somme de nombre d'obstacle qui se trouve entre chaque deux positions et l'erreur entre le chemin réel et entre le chemin référence.

Et pour l'adaptation de la valeur de nombre d'obstacle et l'erreur, on utilise la fonction suivante :

$$F = |y_c - y_r|$$

y_c : La disposition dans la chemin réel (courant).

y_r : La disposition dans la chemin référence.

III.3.2.3. La procédure de sélection :

La procédure de sélection que nous avons utilisée est la roulette biaisée de Goldberg « roulette Wheel » cette dernière fonctionne sur le principe suivant :

Chaque individu i est représenté par sa fitness.

On procède alors à la sélection de N individus survivants en effectuant N lignes

aléatoire dans le domaine [0,1]. Soit R le nombre aléatoire tiré, l'individu j sélectionne.

(Selon cette procédure de sélection, plus un individu est fort, plus il y aura des chances de survivre).

III.3.2.4. Le croisement :

Les individus survivants à la phase de sélection sont appariés aléatoirement et chaque paire formée va subir le croisement avec une probabilité pc, de nombreux types de croisement différents existent dans la littérature, préservant plus ou moins l'identité génétique des parents et permettent de se déplacer dans tout l'espace des grande, plus la convergence de l'algorithme est rapide (mais plus on risque de convergence vers un optimum local),nous avons utilisé le croisement à 1 point , pour faire ça on génère un point entier (cpoint) entre (1 et le longueur d'individus -1) à l'aide des fonctions round et rand (sous matlab) puis nous avons fait le croisement à partir de ce point.

```
Fonction croisement (parent1,parent2,pc)
    Début
        Si (random<pc)
            Générer un point (cpoint) entre (1 et K-1) ;
            Fils_1=parent_1 [1, cpoint] U parent_2 [cpoint, K] ;
            Fils_2= parent_2 [1, cpoint] U parent_1 [cpoint, K] ;
            Evaluer les fils ;
        Sinon
            Fils_1=parent_1 ;
            Fils_2=parent_2 ;
        Fin si
    Fin
```

III.3.2.5. La mutation :

Les individus de la population issus du croisement vont ensuite subir à un processus de mutation avec une probabilité **pm**. Nous présentons ici la plus simple qui

consiste à modifier un bit aléatoirement d'un individu, cette méthode est expliquée sur la fonction suivante :

```
Fonction mutation (parent, pm)
    Début
        Si (random<pm)
            Générer un point (cpoint) entre (1 et K) ;
            Fils [mpoint]=valeur absolu (parent [mpoint]-1) ;
            Evaluer le fils ;
        Sinon
            Fils=parent ;
        Fin si
    Fin
```

La mutation évite la dégénérescence de la population (en d'autre terme, elle permet de quitter les extrêmes locaux) [15]

III.4. Environnement du travail :

Ce logiciel a été développé sur une machine de type Intel pentium IV, dans l'environnement WINDOWS XP, avec l'utilisation de MATLAB 7.1 comme un langage de programmation, facile et simple.

III.5. Présentation du logiciel :

Dans la figure ci-dessous, on trouve la fenêtre principale de notre simulateur.

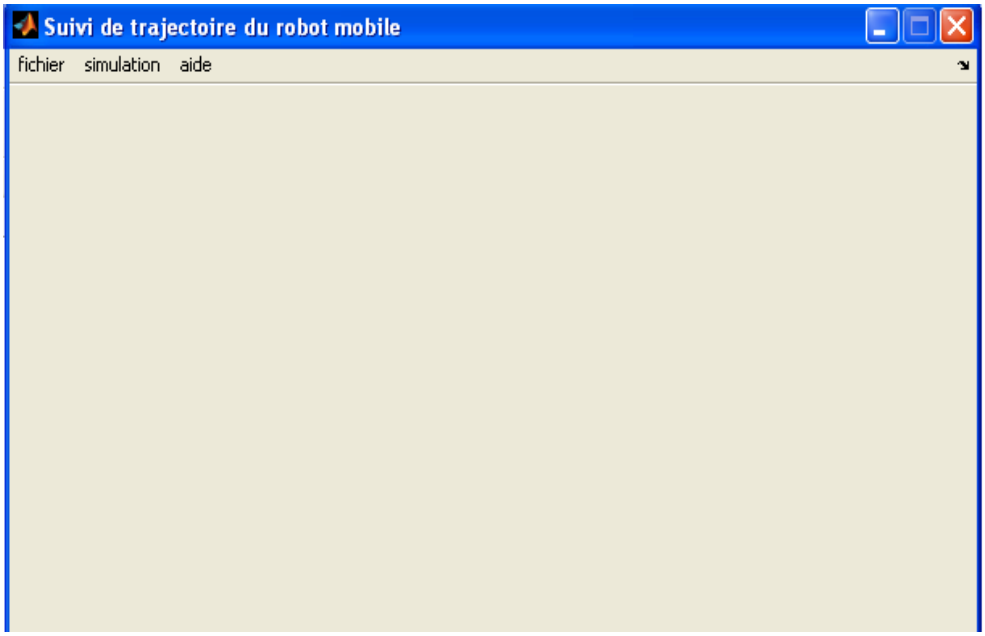
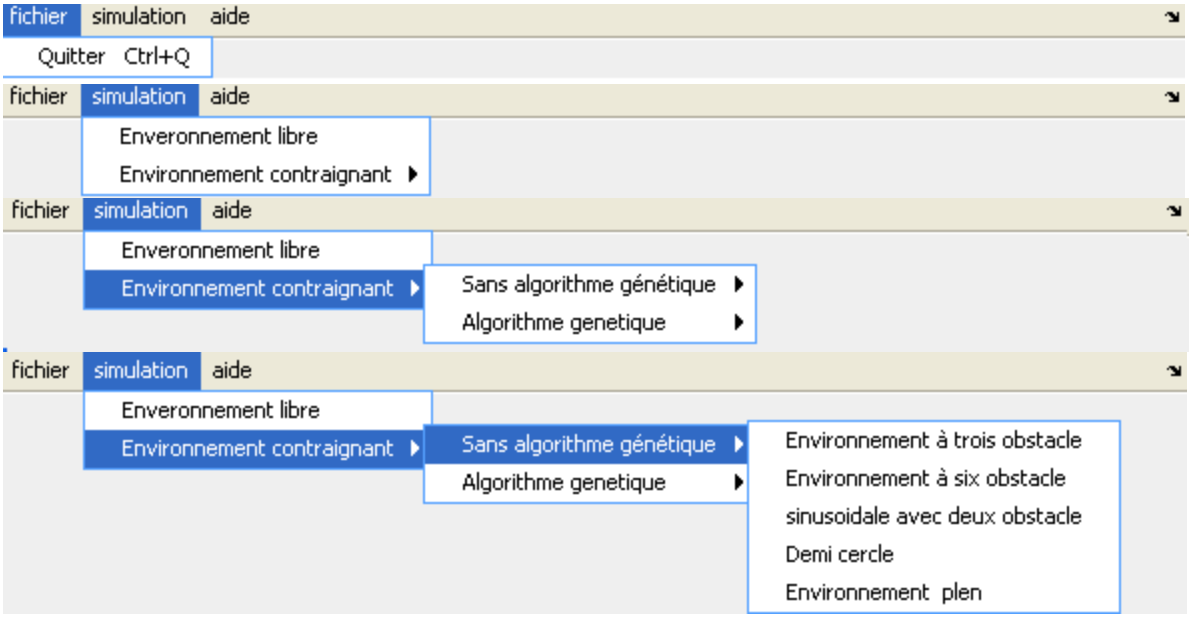


FIG.III.1 : la fenêtre principale

III.5.1. barre de menu :

On trouve dans la barre de menu trois boutons principaux : fichier, simulation, aide. Chaque bouton contient des boutons secondaires illustrés par la figure suivante :



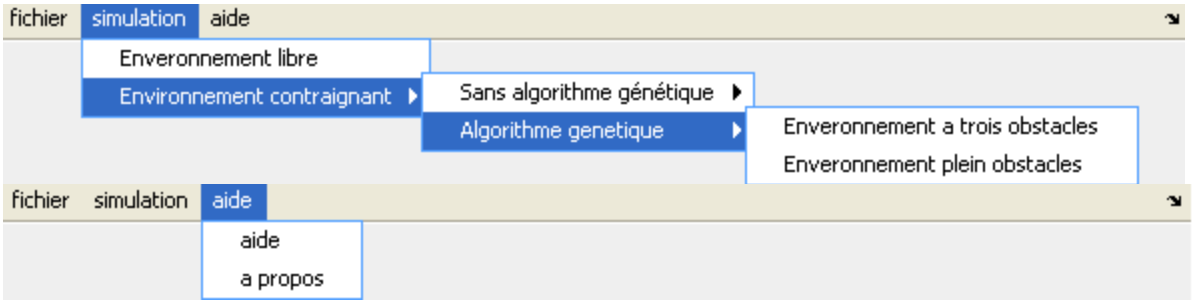


FIG.III.2 : les boutons de la barre de menu.

III.5.2. La description détaillée de chaque bouton :

- Le bouton fichier :

FIG.III.3 : le bouton Quitter.

Ce bouton pour quitte le simulateur.

- Le bouton environnement libre :

FIG.III.4 : le bouton environnement libre.

Ce bouton permet le passage à la figure de l'environnement libre, illustré ci-dessous :

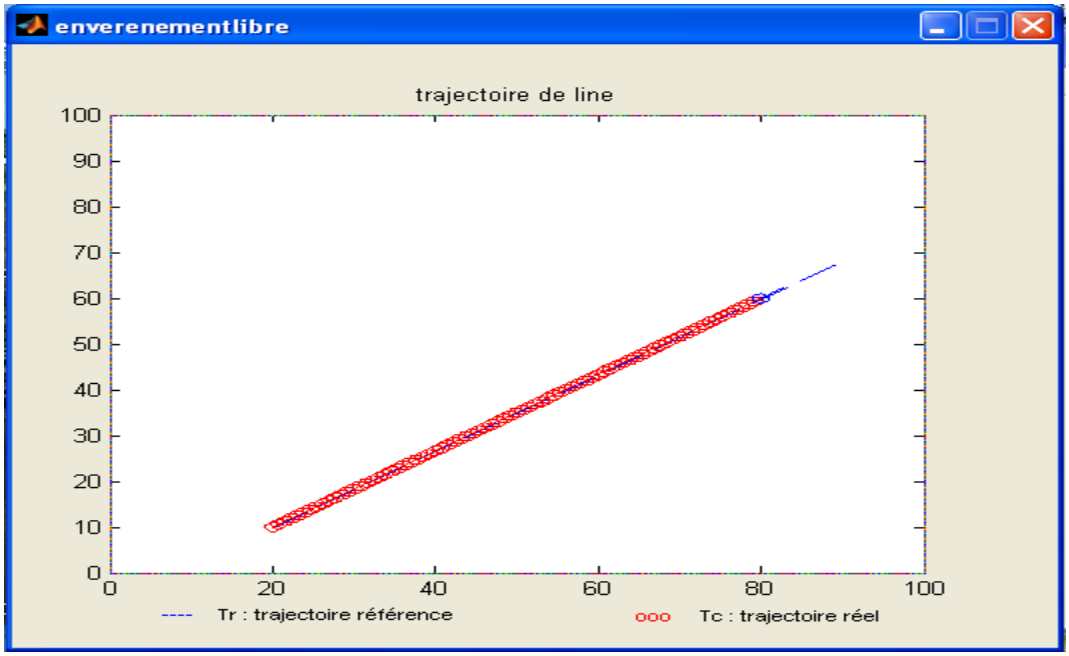


FIG.III.5 : L'environnement libre

On remarque ici que le robot suit une ligne droite, puisque on n'a aucun obstacle.

Dans l'environnement contraignant on applique la méthode d'algorithme génétique et sans algorithme génétique.

- Le bouton Sans Algorithme Génétique :

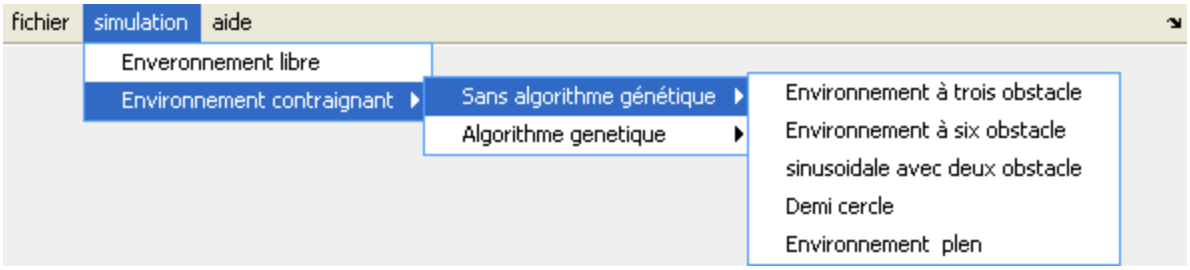


FIG.III.6 : le bouton Sans Algorithme Génétique

Dans notre logiciel nous avons simulé aussi une autre méthode on observe que le robot établir chemin réel pour suivi le chemin référence et éviter les 'obstacles, Cette méthode n'utilise pas l'algorithme génétique.

Lorsqu'on commence la simulation, par un clique de bouton sans algorithme génétique (trois obstacle, six obstacle, sinusoïdale, demi cercle et plein obstacle), comme indique la figure suivante.

- Le bouton environnement a trois obstacles (Sans AG) :

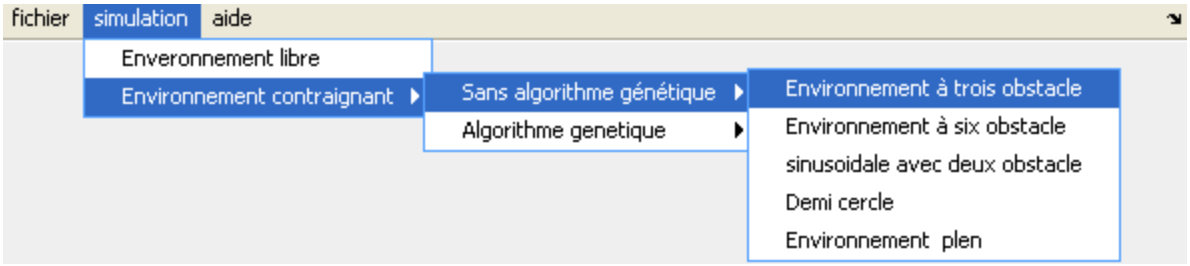


FIG.III.7 : le bouton environnement a trois obstacles.

Ce bouton permet le passage à l'environnement a trois obstacles, la simulation est commence et désigné la trajectoire réel du robot et la trajectoire référence qui sont illustrés par la figure suivante (Fig.III.8) :

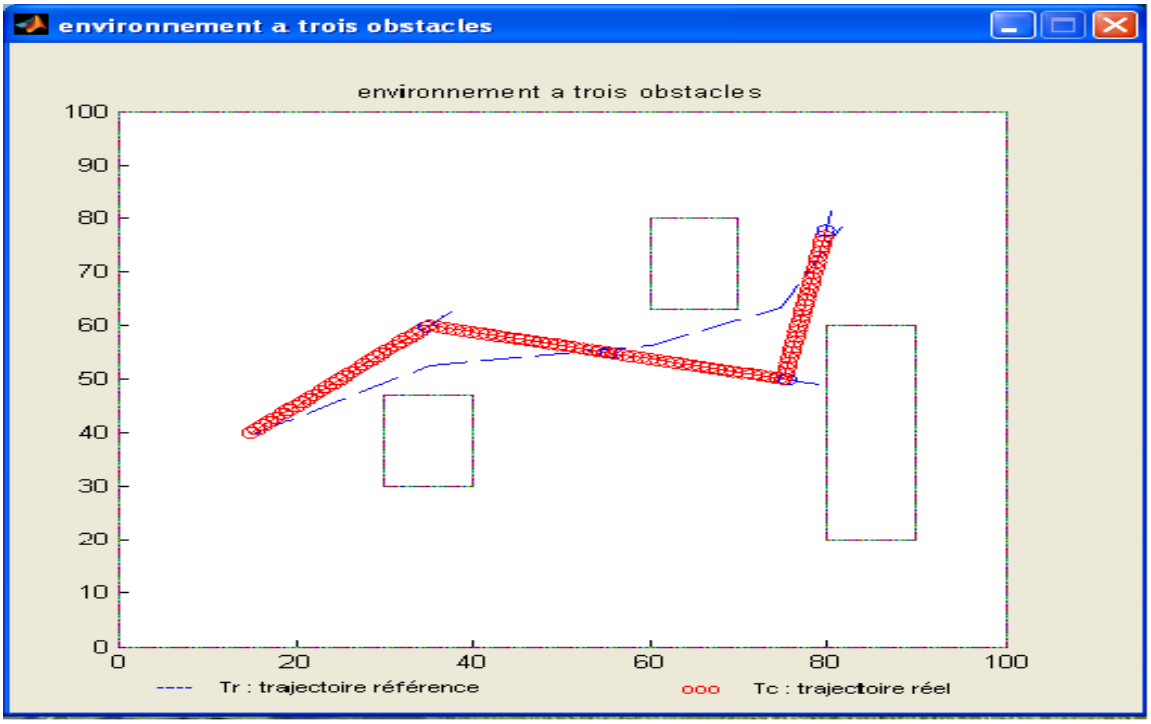


FIG.III.8 : L'environnement a trois obstacles

Dans cette figure on observe que le robot ne suivre pas correctement la trajectoire de référence et fait un grand erreur.

- Le bouton environnement à Six obstacles (Sans AG) :

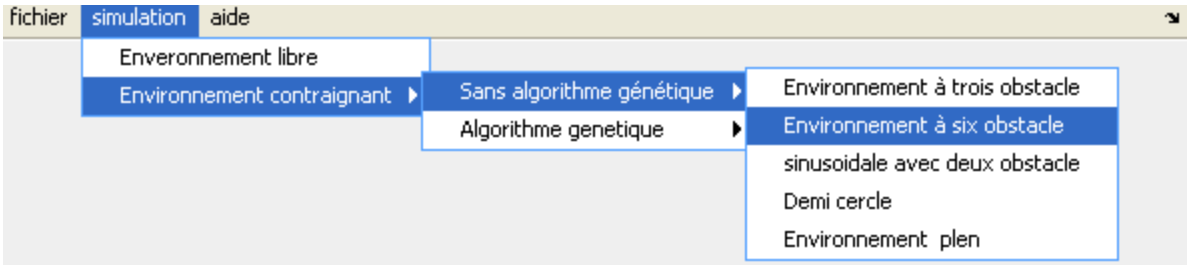


FIG.III.9 : le bouton environnement a six obstacles.

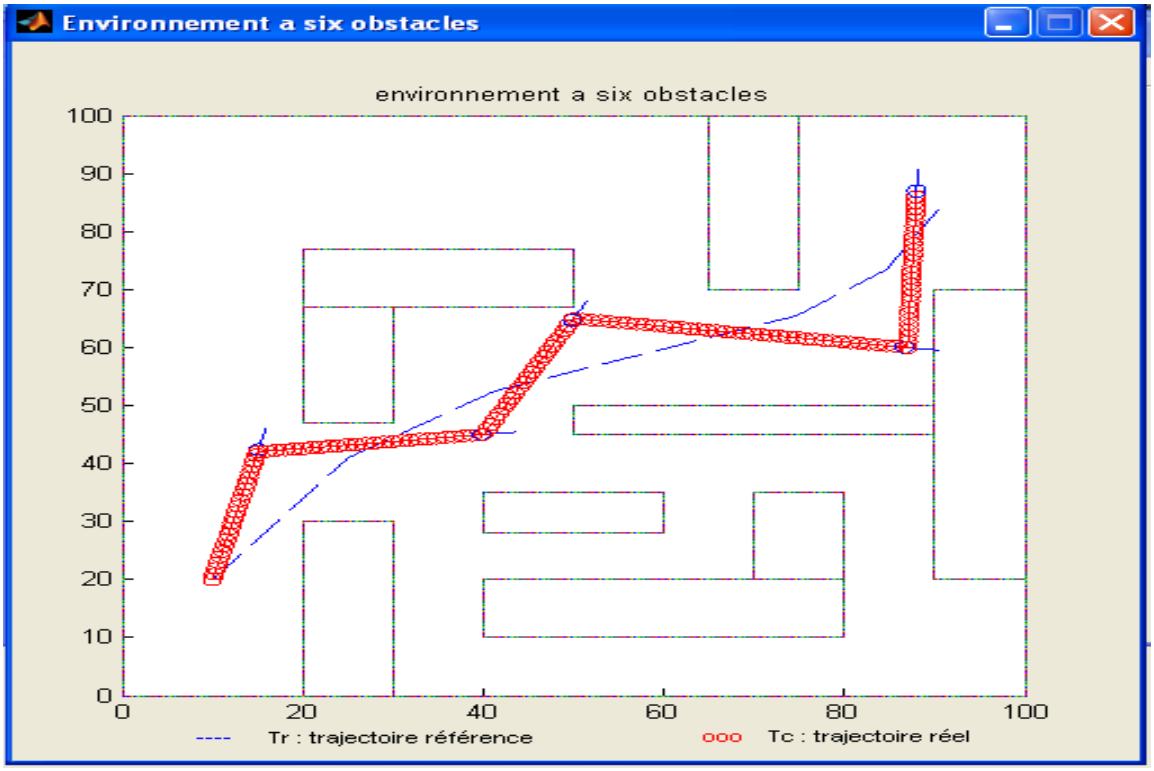


FIG.III.10 : L'environnement a six obstacles

❖ Le bouton environnement a sinusoïdale (Sans AG) :

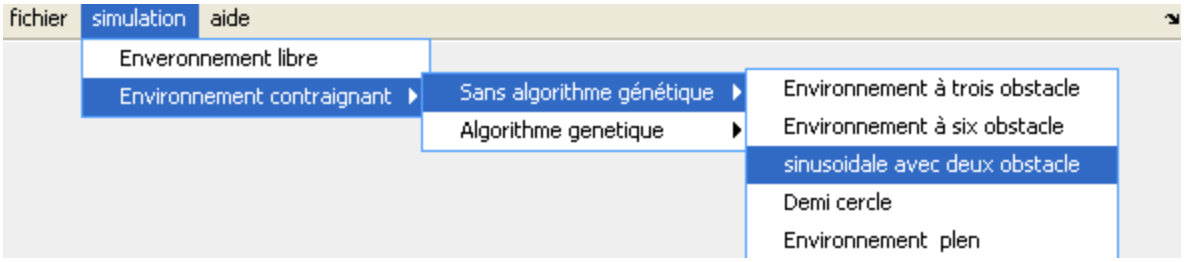


FIG.III.11 : le bouton environnement a sinusoïdale

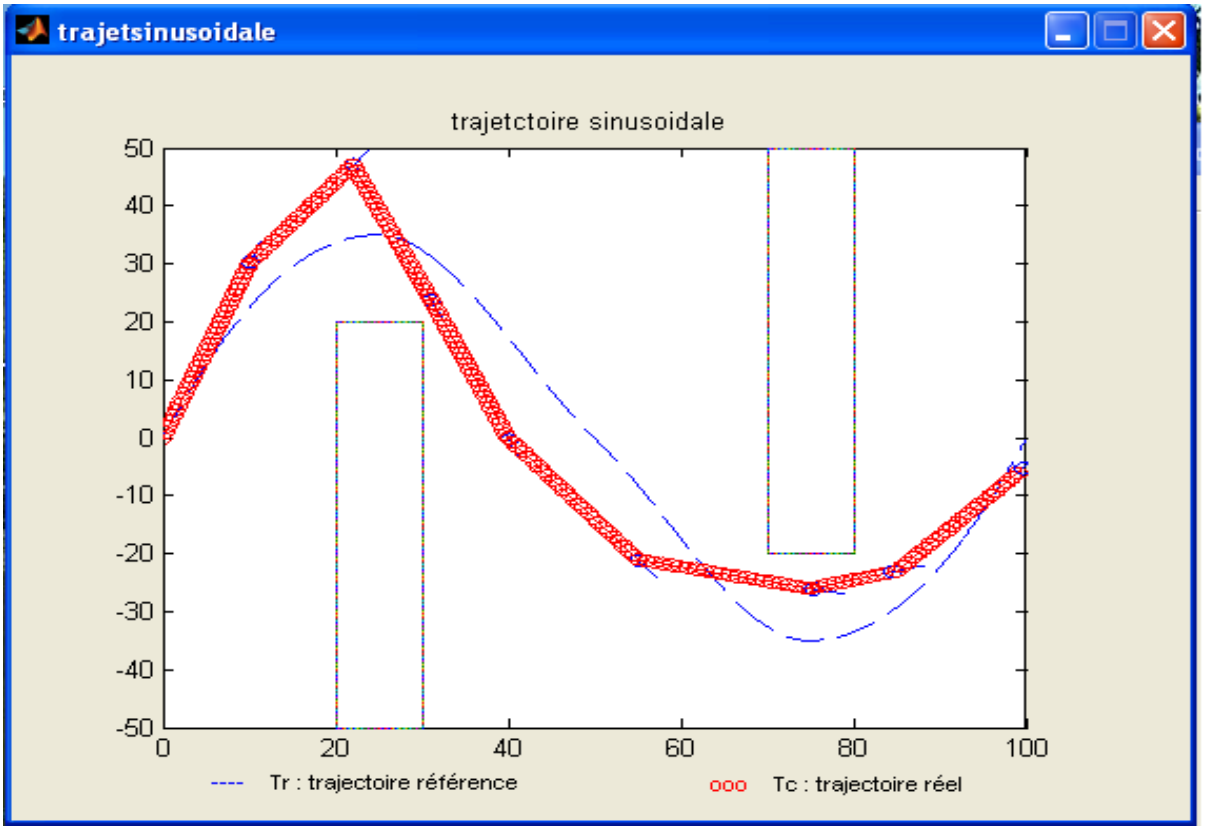


FIG.III.12 : L'environnement a sinusoïdale

❖ Le bouton environnement a demi-cercle (Sans AG) :

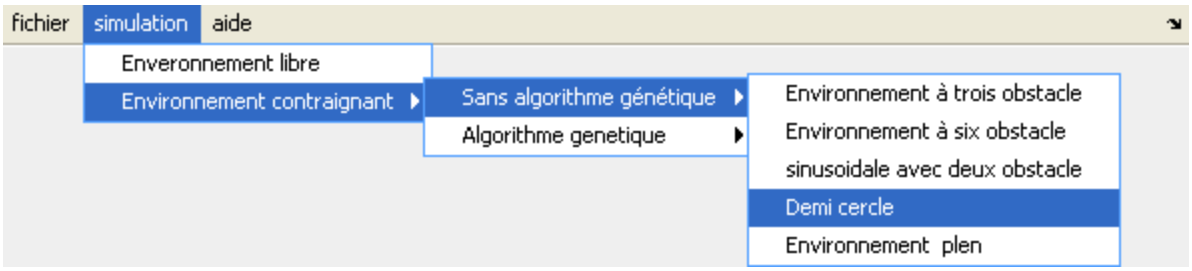


FIG.III.13 : le bouton environnement a demi-cercle

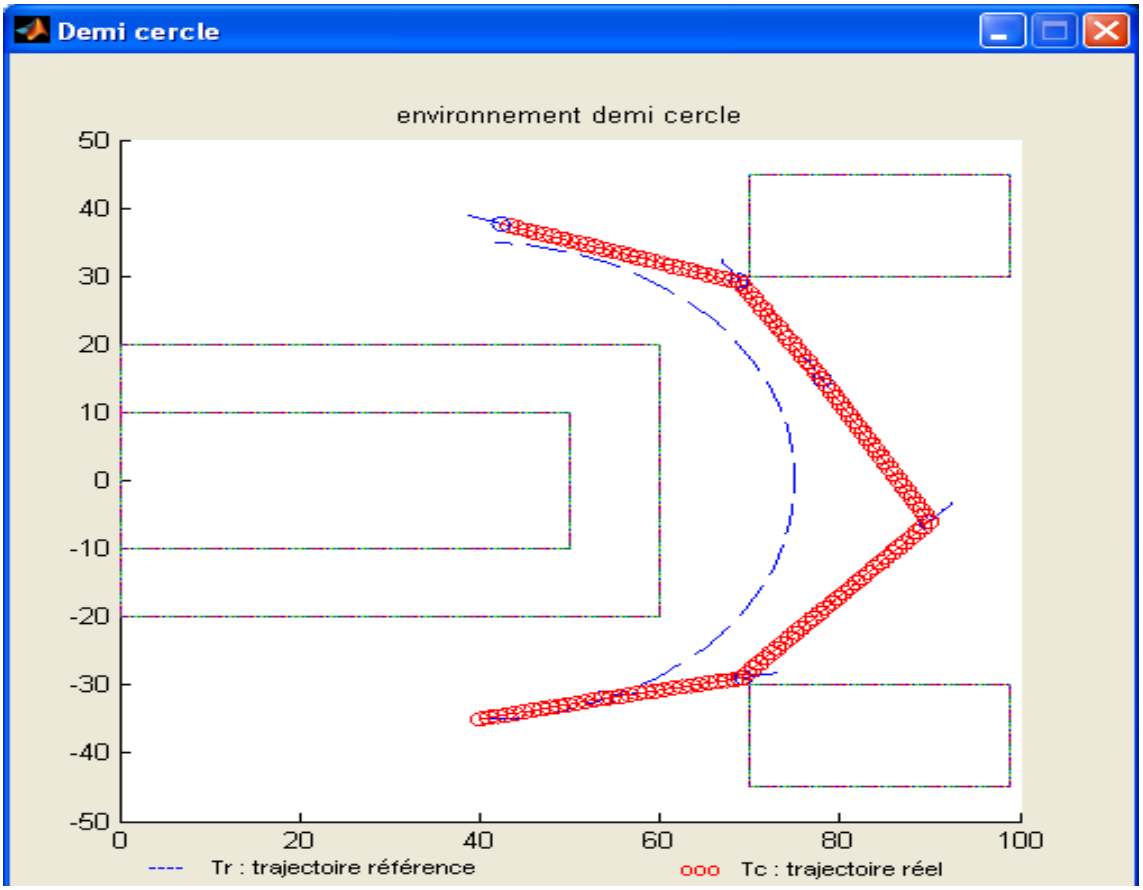


FIG.III.14 : L'environnement a demi-cercle

❖ Le bouton environnement a plein obstacles (Sans AG) :

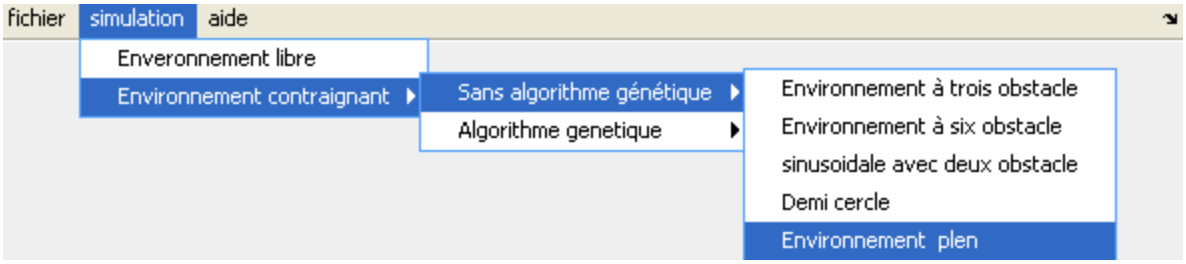


FIG.III.15 : le bouton environnement a plein obstacles

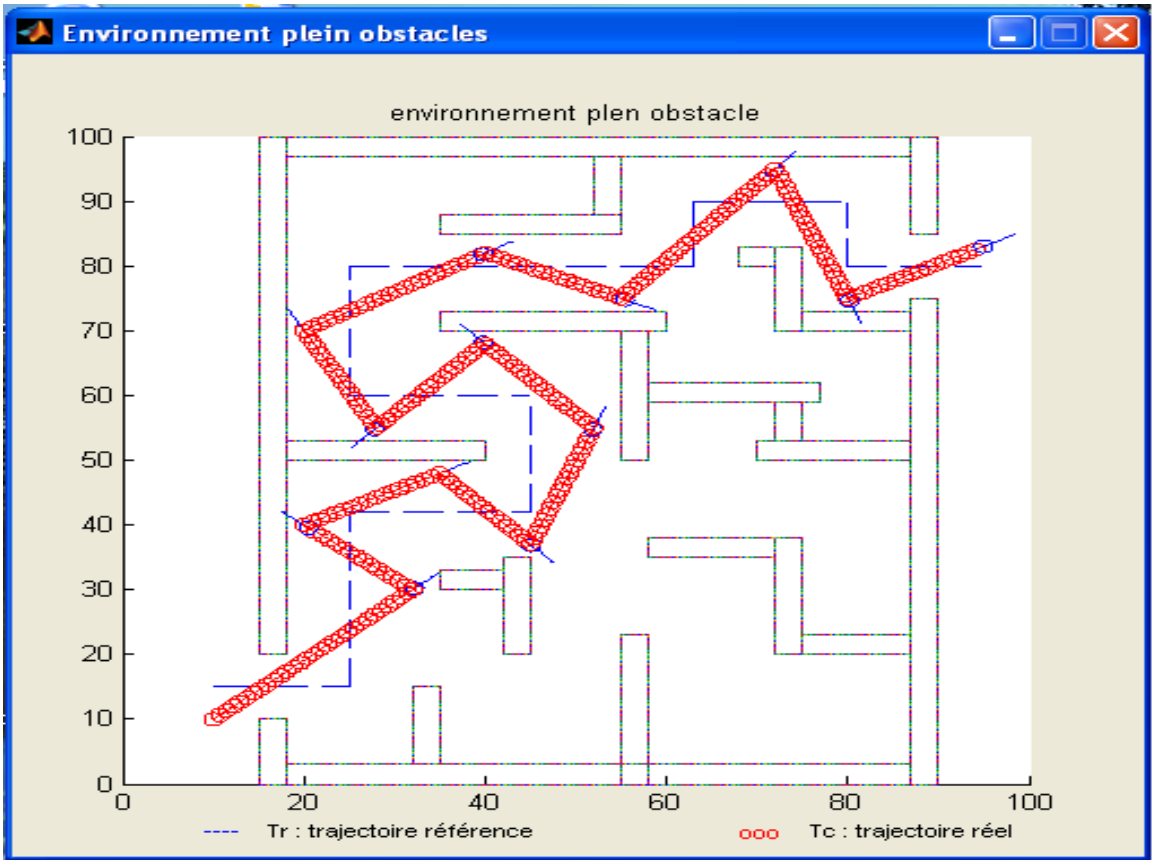


FIG.III.16 : L'environnement a plein obstacles

On observe que l'erreur entre la trajectoire du robot et la trajectoire de référence est grande.

Pour minimiser cet erreur en utilisé les algorithmes génétiques.

- Le bouton environnement a trois obstacles (AG)

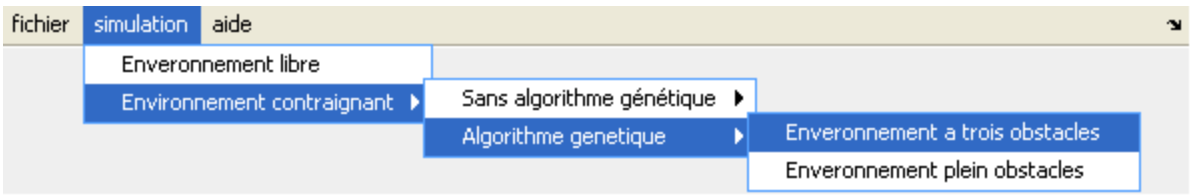


FIG.III.17 : le bouton environnement a trois obstacles (AG)

Ce bouton permet le passe à l'environnement a trois obstacles, le résultat du simulation est illustré par la figure suivante (Fig.III.18) :

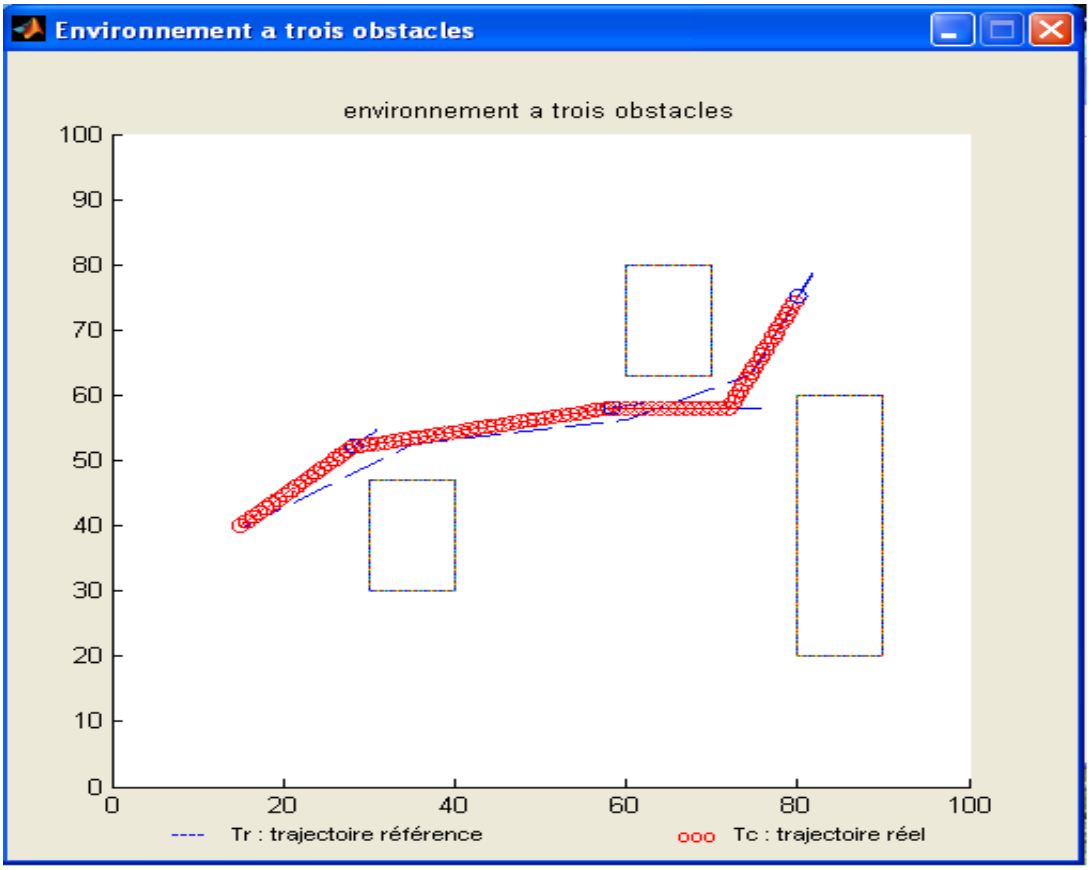


FIG.III.18 : L'environnement a trois obstacles

Dans cette figure on observe avec l'utilisation des algorithmes génétiques.

❖ Le bouton environnement à six obstacles :

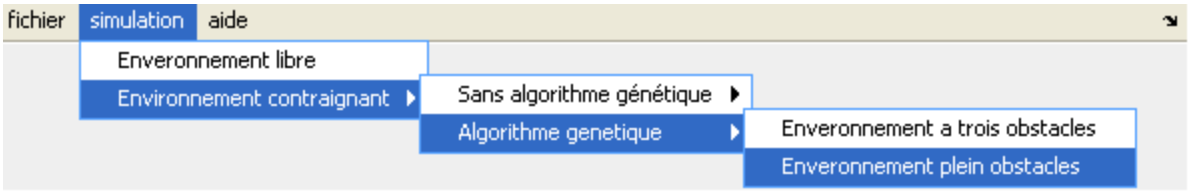


FIG.III.19 : le bouton environnement a six obstacles(AG).

Ce bouton donne le résultat de simulation des trajectoire dans un environnement à six obstacles qui sont illustrés par la figure suivante (fig.III.20) :

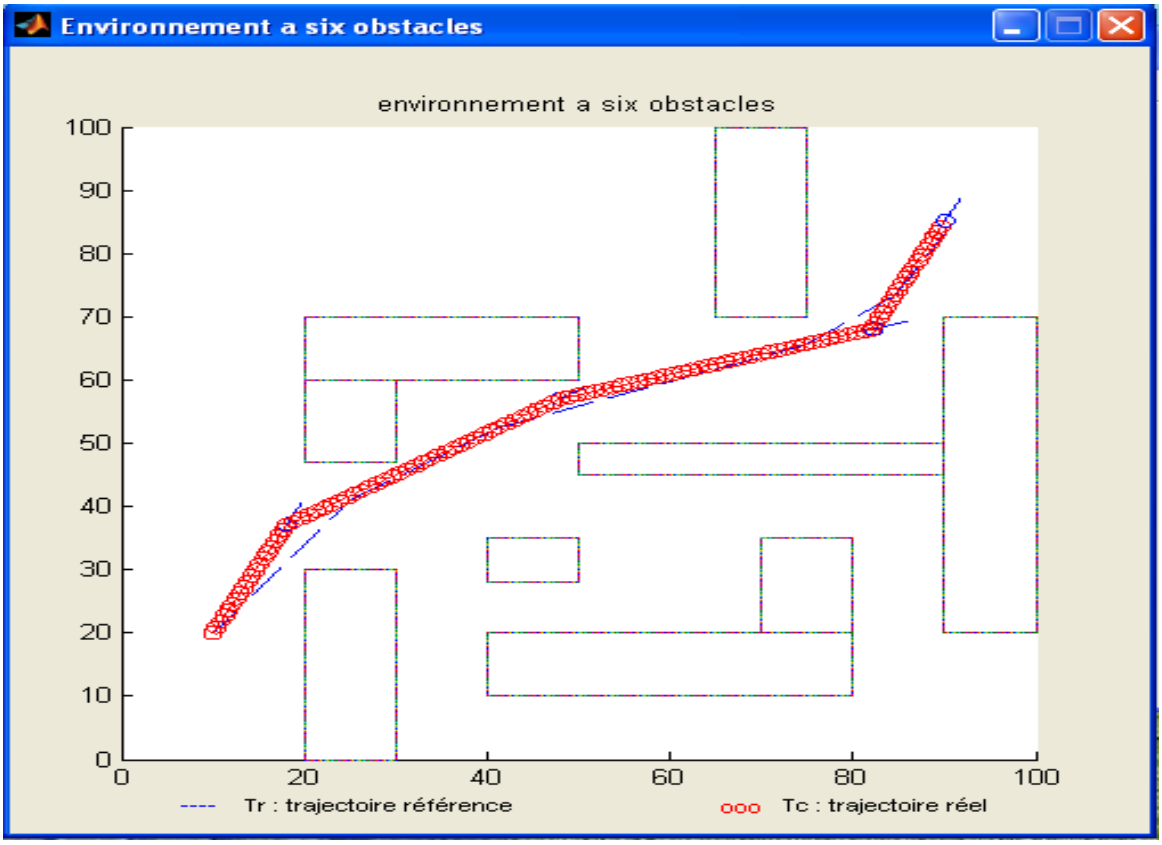


FIG.III.20 : L'environnement a six obstacles

Donc l'utilusation des algorithmes génétiques permet de minimiser l'erreur et donne des bons résultats comme indique les figures (**III.18)** et (**III.20)**.

- Le bouton aide:

  Permet d'afficher des informations générales sur l'application

FIG.III.21 : Le bouton aide.

- Le bouton a propos:

  Affiche des informations sur le projet

FIG.III.22 : Le bouton a propos.

III.6. Conclusion :

On essayant dans ce chapitre de présenter le simulateur que l'on a réalisé, notre logiciel est simple et flexible pour l'utilisateur, il comporte les méthodes et les fondements que nous avons étudiés et présentés dans notre projet, la commande du robot KHEPERA pour la suivi de trajectoire de référence en utilise des méthodes de l'intelligence artificiel qui est les algorithmes génétiques ; cette méthode donne des bon résultats pour la minimisation d'erreur entre la trajectoire du robot et trajectoire de référence.

CONCLUSION GENERALE

Le travail présenté dans ce mémoire avaient pour but l'étude de technique d'optimisation appliquée aux erreurs de suivi de trajectoire par le robot KHEPERA . Arrivé au ce terme, nous proposons de faire un rapide bilan de contenu :

on a commencer par la definition d'un robot mobile et particulièrement le robot " KHEPERA " qu'on a choisi comme un mini- robot permet de faciliter la recherche et le développement dans le cadre de robot mobile de type1, puis citer la description technique et le modèle cinématique de ce robot , nous avons utiliser dans le simulateur exclusivement : les algorithmes génétiques comme outil d'optimisation ,qui donne des résultats satisfaisants et précis, mais leur inconvenient est de prendre un temp considerable pendant la phase d'exécution (le grand nombre d'itération et le grand taille de population).

Le travail décrit dans ce memoire permet de :

* comprendre le principe des algorithmes génétiques et leurs utilisation .
* comprendre quelques notions de bases qui concerne la modélisation cinématique et les systèmes de perception pour les robots mobiles.
* Avoir une bonne maitrise de logiciel de programmation (MATLAB 7.1 et sa GUI).
* D'avoir une initiation dans le domaine de recherche.

et comme future perspective en peut utiliser des autre méthodes de commande d'intelligence artificial comme les réseau de neuron et la logique floue dans des autres environnements.

on ajoute un annexe pour expliquer les types des capteurs utilisées dans la robotique mobile et sont détailles. et un autre annexe pour la présentation générale de logicielle de programmation MATLAB7 .

Système de perception pour robot mobile :

I.1. Perception :

La notion de perception en robotique mobile est relative à la capacité du système à recueillir, traiter et mettre en forme des informations utiles au robot pour agir et réagir dans le monde qui l'entoure. Alors que pour des tâches de manipulation on peut considérer que l'environnement du robot est relativement structuré, ce n'est plus le cas lorsqu'il s'agit de naviguer de manière autonome dans des lieux très partiellement connus. Aussi, pour extraire les informations utiles à l'accomplissement de sa tache, il est nécessaire que le robot dispose de nombreux capteurs mesurant aussi bien son état interne que l'environnement dans lequel il évolue. Le choix des capteurs dépend bien évidemment de l'application envisagée. [1]

I.2. Les capteurs en robotique mobile :

Etude détaillée des capteurs =physique + électronique + traitement du signal + ……
Dans ce cours : uniquement principe de fonctionnement des capteurs embarqués pour la navigation d'un robot mobile.

I.2.1. Mesure de la rotation des roues :

Moteur +capteur = servomoteur

Configuration habituelle : roue motrice + dispositif de mesure de rotation à l'aide d'un capteur disposé sur l'axe lui-même.

Capteurs proprioceptifs.

Propriétés :

- ✓ peu différents des capteurs habituels en commande d'axe
- ✓ codeurs absolus inutiles
- ✓ grandeur de mesure = vitesse de rotation
- ✓ rarement mesure directe (génératrice tachymétrique)
- ✓ mesure indirecte par codeurs optiques (moindre coût, moindre encombrement, moindre entretien) : comptage d'impulsions (bruit de quantification aux basses vitesses)
- ✓ résolution fine ou synchrorésolveurs

I.2.2. Mesure de position et d'orientation :

a. Mesures de la position :

- **GPS :**Global Positioning System (GPS) : applications militaires, actuellement à la disposition du grand public.

Emissions synchronisées dans le temps perçues et recoupées au niveau du récepteur par triangulation.

Propriétés :

- ✓ précision brute de l'ordre de la quinzaine de mètres
- ✓ méthode différentielle à l'aide de deux récepteurs : précision de l'ordre du centimètre
- ✓ positionnement en extérieur
- ✓ faible précision, prix élevé : des systèmes multirobots
- ✓ mesure à des fréquences de l'ordre de 5 Hz : pas temps-réel : recalage de la position

b. Mesure d'orientation :

- **Gyromètres :**Gyromètres c'est un capteur proprioceptif pour mesurer l'orientation du corps sur lequel ils sont placés par rapport à un référentiel fixe, selon un ou deux axes.

  Mécaniques, optiques, à structure vibrante, etc.

Propriétés :

- ✓ gyromètres mécaniques et optiques : performances très supérieures à celles requises en robotique mobile, coût élevé (aéronautique et spatial)
- ✓ structure vibrante

- **Compas et boussoles :** information d'orientation par rapport à une référence fixe (nord magnétique typiquement).

Propriétés :

- ✓ compas électroniques capables de détecter le nord
- ✓ CMP03 : résolution de 3 à 4o environ, 32 _ 35 mm

I.2.3. Mesure de proximité et de distance :

- **Télémétrie :** toute technique de mesure de distance par des procédés acoustiques, optiques ou radioélectriques.

  Capteur = télémètre :

  - ✓ différentes techniques de mesure de distance (mesure du temps de vol d'une onde, triangulation)
  - ✓ différentes technologies.

- **Capteur infrarouge :** c'est un ensemble émetteur/récepteur utilisant des radiations non visibles.

  Propriétés :

  - ✓ faible portée, mesure très dégradée au-delà de un mètre : détecteurs de proximité ou de présence
  - ✓ Sensibilité aux conditions extérieures (lumière ambiante, spécularité des surfaces, température, pression)
  - ✓ cône de détection
  - ✓ alternance émission/réception : distance minimale.[18]106

- **Capteurs ultrasonores :** utilisent des vibrations sonores non perceptibles pour l'oreille humaine (20 kHz à 200 kHz).

  Propriétés :

  - ✓ distance maximale et distance effective de mesure à adapter à la plage de mesure
  - ✓ sensibilité à la densité de l'air (température, pression)
  - ✓ cône de détection
  - ✓ alternance émission/réception : distance minimale
  - ✓ fréquence maximale des mesures variable (mesures à 1; 5 m : toutes les 10 ms, à 30 m toute les 200 ms).

- **Télémètres laser :** mesure du temps de vol d'une impulsion émise par une diode laser faible puissance.

  Balayage : direction de mesure modifiée par rotation d'un miroir.

  Propriétés :

- ✓ bonne précision, télémètres les plus répandus
- ✓ bonne résolution angulaire
- ✓ distance maximale (conseillée) de mesure : 30 m
- ✓ balayage sur 100 à 180 degrés
- ✓ bonne stabilité en température
- ✓ plus encombrant
- ✓ mesure complète (balayage) en quelques secondes
- ✓ prix élevé

I.2.4. Vision par ordinateur :

❖ **Vision et robotique mobile :**

Vision par ordinateur : vision traditionnelle, stéréovision.

Vision omnidirectionnelle : mesure de la réflexion de l'environnement sur un miroir parabolique.

Propriétés :

- ✓ vision panoramique de la scène
- ✓ détection temps-réel
- ✓ difficultés technologiques : alignement caméra-miroir, régularité du miroir, géométrie de capteur rectangulaire à pas constant
- ✓ en développement [01]

I.2.3. classification des capteurs :

a)-Capteur interne (proprioceptifs) :

Dans les procédures mécaniques, ces capteurs définissent la configuration des divers axes du robot et mesurent essentiellement des positions de vitesses, des forces…etc., permettent ainsi un contrôle permanant de la bonne exécution du mouvement.

b)-Capteurs externe (extéroceptifs) :

Les capteurs externes servent pour les actions suivantes :

- ➢ Mesure des interactions entre le robot et l'environnement.

- Jouer un rôle majeur au niveau de commande dont le but est la vision de l'environnement.

Assure le fonctionnement et la sécurité.[07]

II. Dispositifs des contrôles du robot KHEPERA :

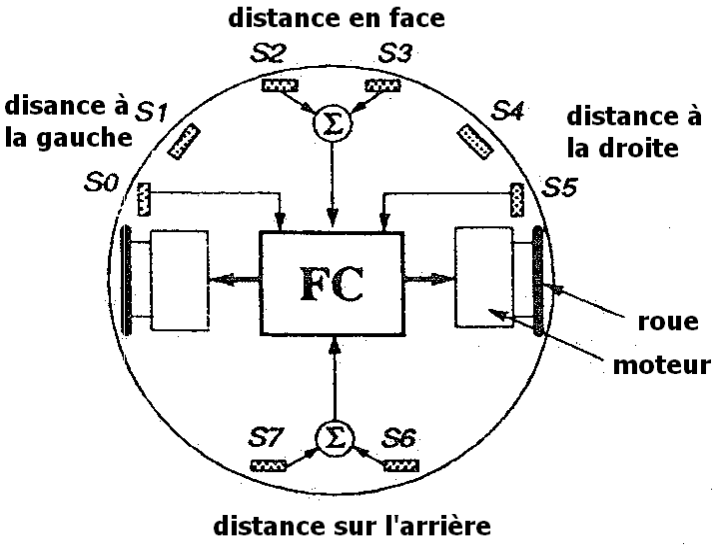


FIG.A.1 : Disposition des capteurs et actionneurs [08]

III. Systèmes réactifs et cognitifs :

Depuis le début de la robotique et ce jusqu'aux années 90, la majeure partie des systèmes robotisés étaient basée sur la même architecture (Figure .A.2). Les informations sur l'environnement sont stockées, analysées et modélisées avant que l'agent puisse prendre une décision. Ce type d'architecture est dit cognitif.

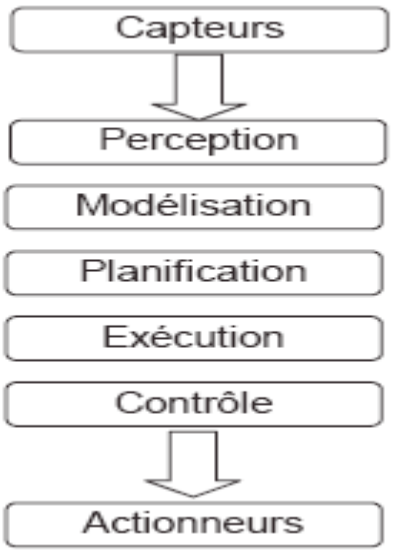


FIG .A.2 : Architecture traditionnelle de décomposition du programme de contrôle du robot en différents modules de fonctionnement

En 1986, Rodney A. Brooks propose une approche différente qui reste aujourd'hui encore une référence dans le domaine des systèmes multi-agents. Cette architecture, appelée " subsomption ", consiste à paralléliser les tâches. La Figure .A.3 décrit les différentes couches comportementales d'un agent.

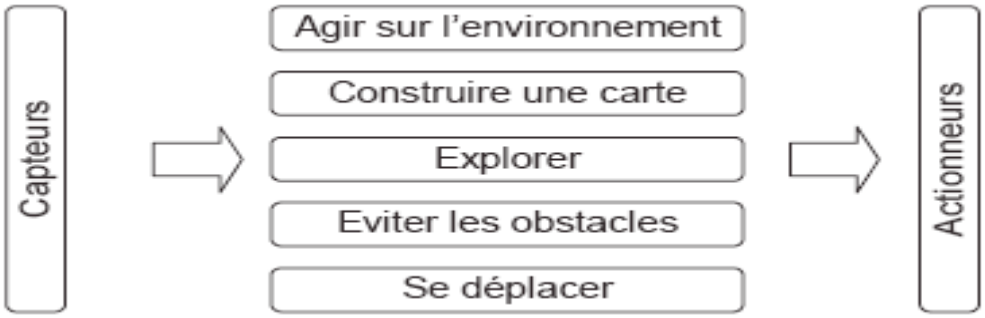


FIG.A.3 : Décomposition basée sur le comportement d'accomplissement de tâches

Chacune de ces couches relie les capteurs aux actionneurs et permet un comportement particulier ou une compétence spécifique, comme la locomotion, l'évitement d'obstacles ou la saisie d'objets. Ce type d'architecture permet notamment la décomposition d'une tâche complexe en plusieurs comportements réactifs. Ce type d'approche vise aussi à accroître la fiabilité du système. Dans l'architecture traditionnelle, si une panne survient sur l'un des modules, alors la panne se généralise à l'ensemble du système, chaque module étant essentiel au fonctionnement de l'ensemble. Dans l'architecture proposée par Rodney A. Brooks, même après la perte d'un module, le système peut continuer à fonctionner en mode dégradé, en inhibant ce module par exemple. [10]

I. Qu'est ce que MATLAB?

MATLAB est un langage hautes performances pour le calcul scientifique et technique. Il intègre la possibilité de calculs, de visualisation et de programmation dans un environnement très simple d'emploi. Les résultats sont exprimes sous une forme mathématique standard. L'utilisation typique est :

– Calcul scientifique
– Développement d'algorithmes
– Acquisition de données
– Modélisation et simulation
– Analyse de données, exploration et visualisation
– Graphisme scientifique
– Développement d'applications, interface graphique (gui) [14]

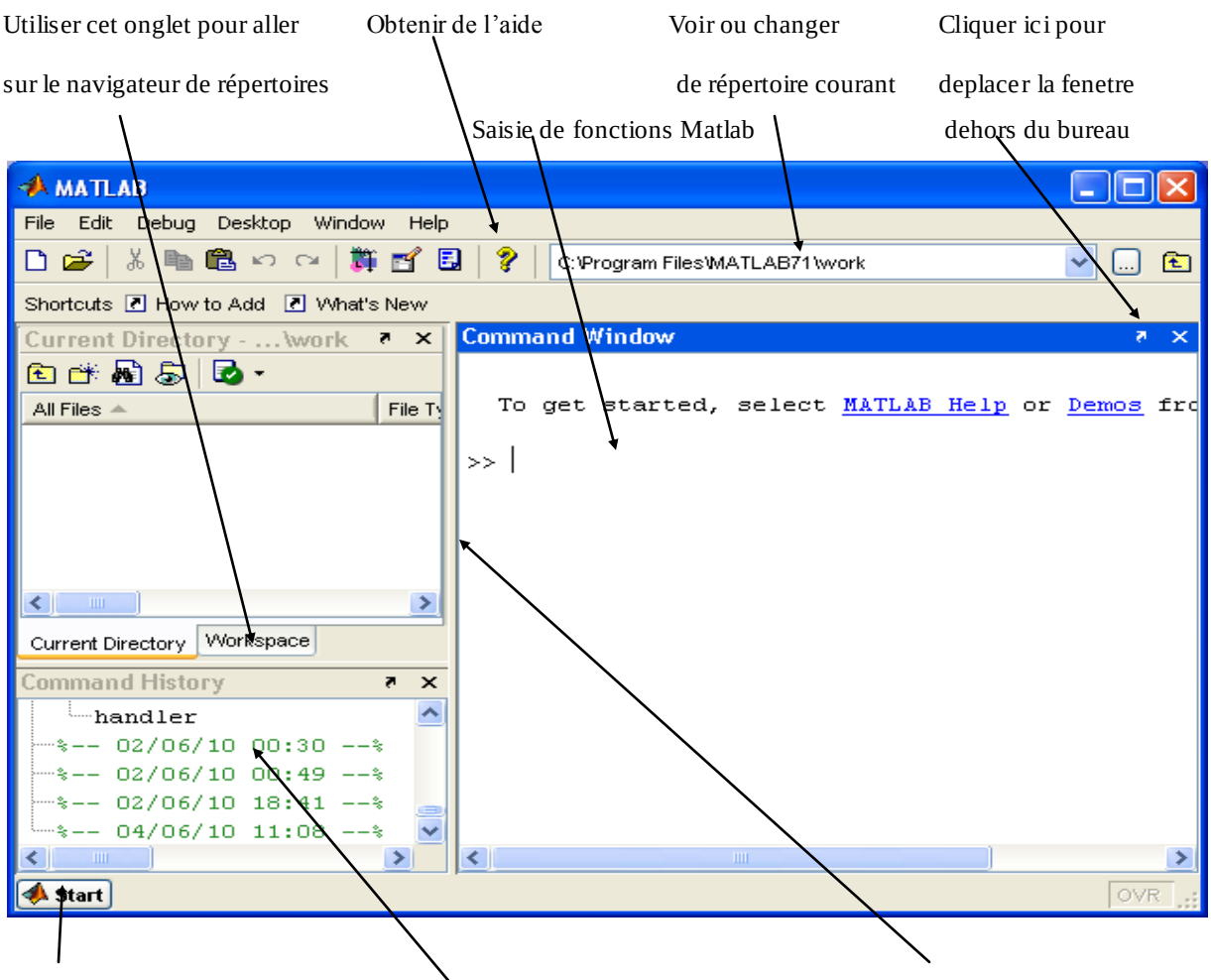


FIG.B.1 : Ecran MATLAB

I.1. Fenêtre Commande :

Dans cette fenêtre, l'usager donne les instructions et MATLAB retourne les résultats.

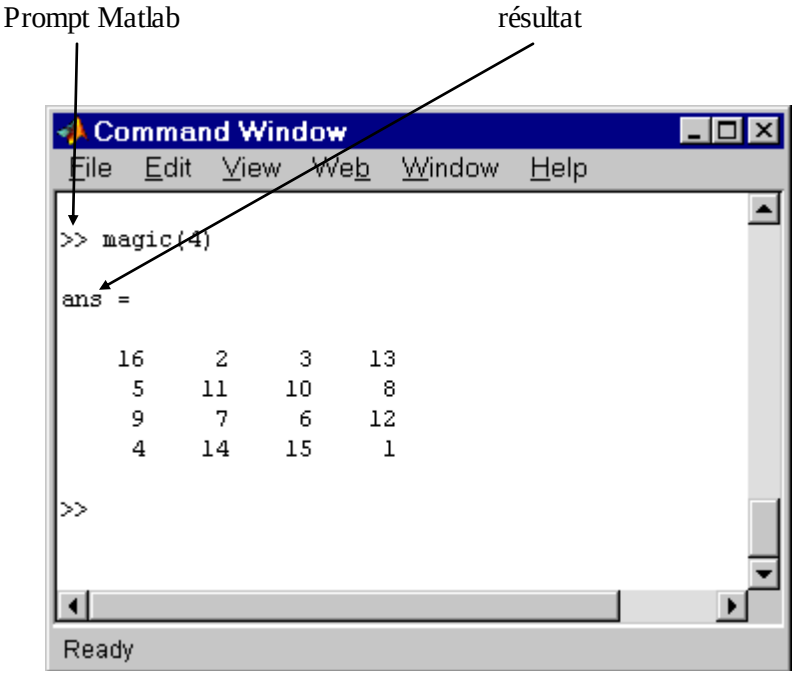


FIG.B.2 : Fenêtre de commande

I.2. Historique de commandes :

Les lignes tapées dans la fenêtre de commande sont automatiquement sauvegardées dans la fenêtre ”Command History”. On y voit donc les lignes précédemment tapées et il est possible de les copier ou d’en sélectionner un
groupe afin de l’exécuter.

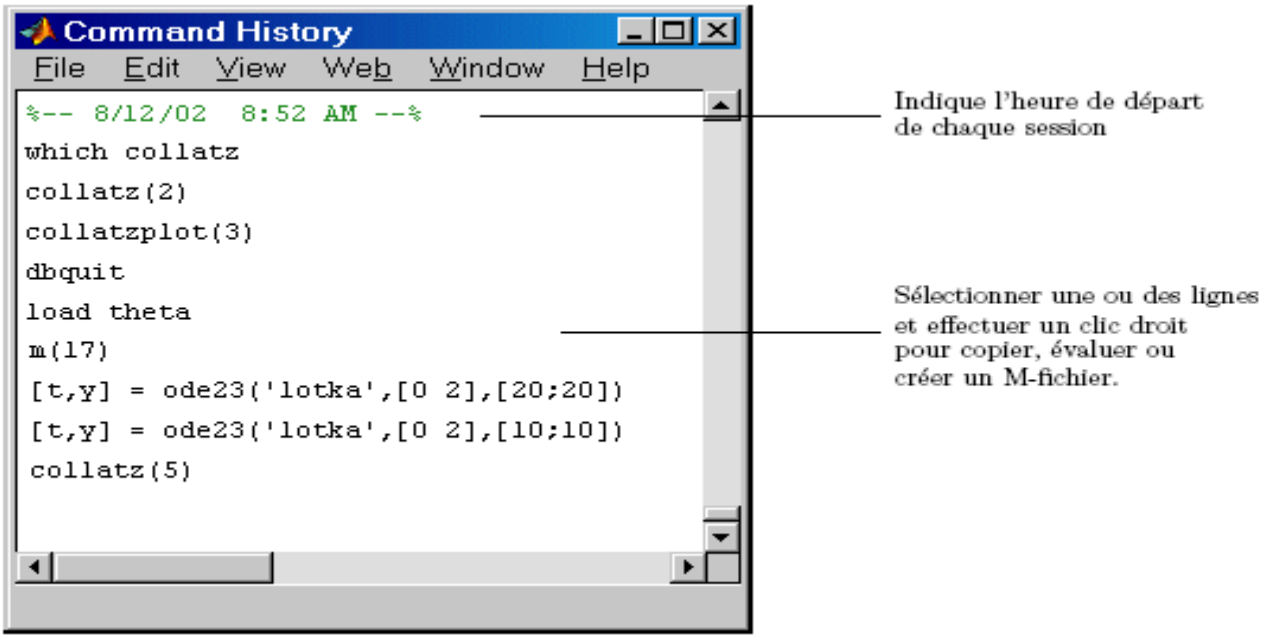


FIG.B.3 : Historique de commandes

II. Utilisation de Matlab (Principes généraux) :

– On travaille dans un répertoire de l'ordinateur, on se place donc dans le bon dossier des le début.

– Chaque fonction ma fonction est écrite dans un fichier ma fonction qui ne contient que cette fonction.

– Une ligne commençant par % n'est pas lue par Matlab, c'est un commentaire.

– Si une ligne de code finit par ; , elle n'affiche aucun résultat dans l'interface.

– On ne travaille que sur des entiers, des nombres décimaux et des matrices les contenant.

– Une fonction prend en entrée des arguments et renvoie un résultat. La syntaxe pour les fonctions est :

fonction [résulta] = ma fonction (arg1, arg2, arg3)

Une fonction ne renvoie rien d'autre que son résultat, donc toutes les lignes de codes de la fonction doivent finir par (;), Lorsque on appelle une fonction, cette fonction modifiera une copie de ses arguments et non pas les arguments originels.

Dans un programme, pensez à nommer intelligemment vos variables : vous devez être

III. Fenêtres Graphique GUI :

MATLAB trace les graphiques dans ces fenêtres graphiques.

Un GUI (Graphic User Interface) est constitué d'objets d'interface (unicontrols) dotés de méthodes et de propriétés programmables dans des scripts.

Pour décrire un GUI, Matlab utilise deux fichiers :

- Un fichier figure (d'extension .fig), qui contient le lay-out du GUI, ou disposition des objets d'interface, toute commande de représentation graphique déclenche la création d'une fenêtre **figure** nommée *figure1* contenant la représentation
- Un fichier script (d'extension .m), qui contient les comportements de l'interface.

FIG.B.4 : Exemple du fichier figure (.fig)

On peut également superposer des représentations graphiques sur une même figure en utilisant la commande.

hold on

Pour désactiver le mode superposé faire

hold off [13]

IV. Fishier SCRIPT :

Le fichier SCRIPT permet de lancer les mêmes opérations que celles écrites directement à l'invite MATLAB. Toutes les variables utilisées dans un SCRIPT sont disponibles à l'invite MATLAB.

Habituellement, on utilise les fichiers SCRIPT afin de :

- Initialiser le système (fonctions *clear*)
- Déclarer les variables
- Effectuer les opérations algébriques
- Appeler les fonctions
- Tracer les figures [06]

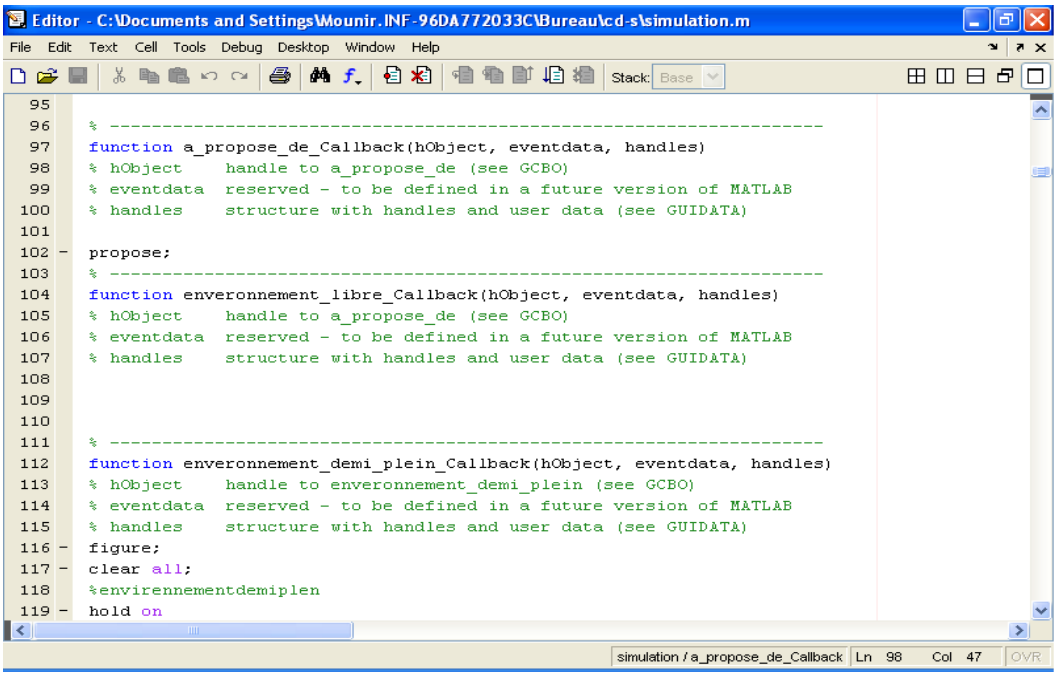


FIG.B.5 : Exemple du fichier script (.m)

V. L'éditeur de GUI et Menu Editor :

GUIDE (pour Graphic User Interface Development Editor) contient les outils pour :

1. disposer les objets d'interface (layout)

2. adapter les valeurs de leurs propriétés

3. programmer les comportements (Callbacks)

On exécute GUIDE depuis la fenêtre Matlab :

1. avec la commande : >> guide
2. à l'aide du raccourci dans la barre d'outils

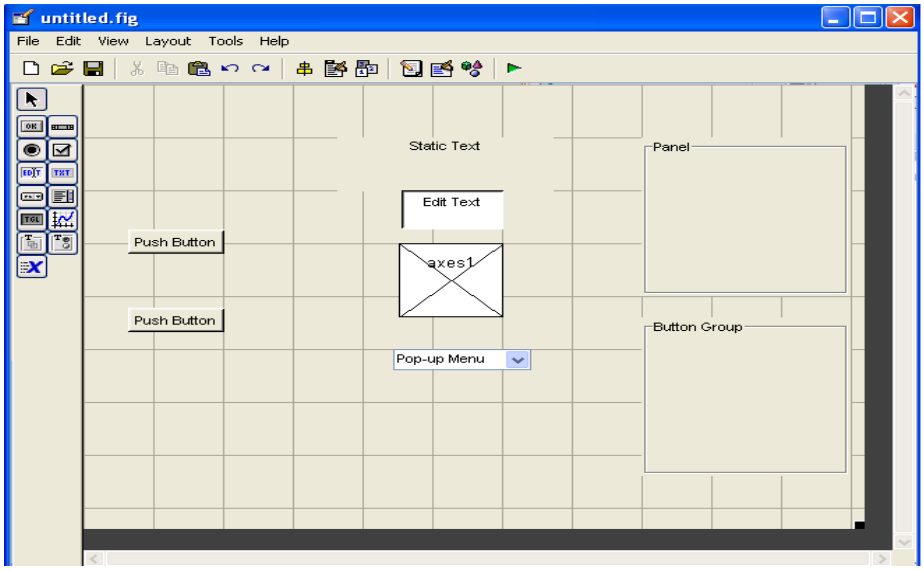


FIG.A.6 : Graphic User Interface

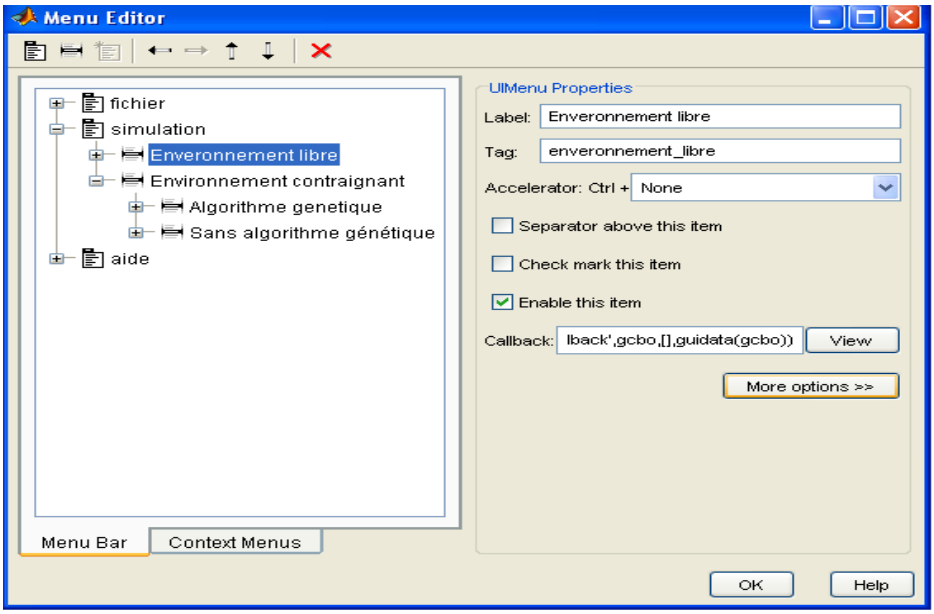


FIG.B.7 : Menu Editor

VI. La liste de propriétés (non exhaustive) :

les objets d'interface sont dotés de propriétés accessibles et modifiables.

- Name, Filename, Pointer, WindowStyle (figure)
- String, le texte qui apparaît (text, edit, …)
- Tag, est le nom de l'objet
- Style, son type, bouton, texte, panel …
- PaperPosition, PaperPositionMode, PaperSize,paperType

- ForegroundColor, …
- Max, Min, Step (slider)
- Value
- Enable,Visible
- Callback, pour coder les comportements
- ButtonDownFcn, ResizeFcn, CreateFcn, KeyPresFcn
- FontSize,FontAngle, FontWeight, FontName, FontUnite

L'inspecteur de propriétés est appelé par GUIDE pour modifier les propriétés des objets d'interface.

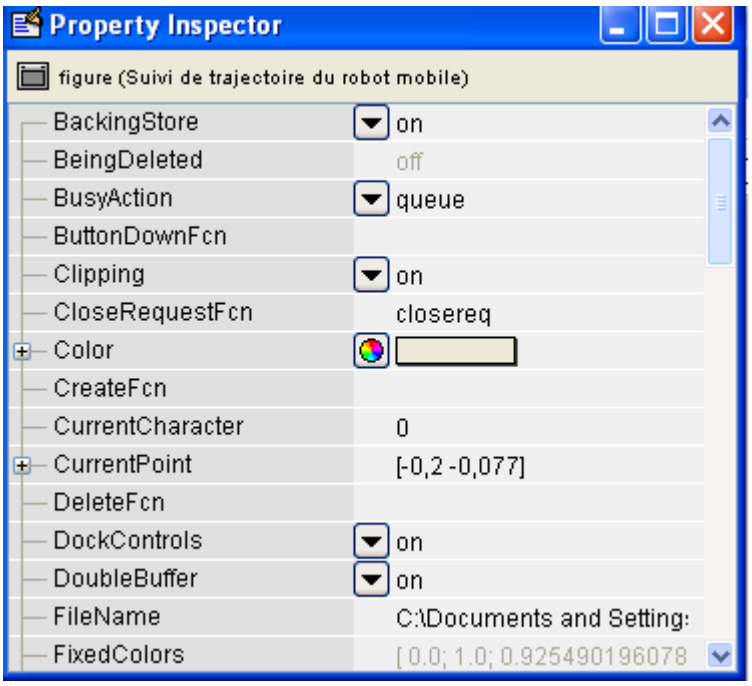


FIG.B.8 : L'inspecteur de propriétés

VII. Les propriétés des objets d'interface et Fonctions principales Options du Graphe :

Pour atteindre les propriétés des objets d'interface, on utilise des 'handler'.

À tout objet d'interface, Matlab peut associer des pointeurs, ou **handler,** pour l'accès aux propriétés.

Certains handlers sont définis par défaut :

- **gcf :** handler de la figure courante (get current figure)
- **gca :** axes courants de tracé (get current axes)
- **gcbf :** la figure cliquée (get callback figure)
- **gcbo :** l'objet qui appelle (celui sur lequel on a cliqué)
- Le handler 0 est la fenêtre interpréteur Matlab
- Le handler 1 est la figure 1, …

Voici des instructions qui utilisent un handler :

- **get et set :** respectivement pour lire et pour changer les valeurs des propriétés
- **findobj :** pour retrouver le handler d'un objet
- **propedit :** pour éditer les propriétés d'un objet.
- **Delete :** pour effacer un objet.

Fonctions principales

- **plot** **:** Graphe en 2D avec une échelle linéaire
- **figure, close** **:** Permet d'ouvrir ou de fermer une figure

Options du graphe

- **titre** **:** Définir le titre du graphe
- **xlabel** **:** affiche un intitulé pour l'axe des x
- **ylabel** **:** affiche un intitulé pour l'axe des y
- **zlabel** **:** affiche un intitulé pour l'axe des z
- **legend** **:** Ajouter une légende sur le graphe
- **text** **:** Permet d'ajouter du texte sur le graphe
- **axis** **:** Définir xmin, xmax, ymin et ymax du graphe.

VIII. Les principales "toolbox" disponibles sur le serveur de MatLab à l'Université :

MATLAB peut aussi être enrichi à l'aide de Toolbox (boites à outils) pour des problèmes spécifiques.

calfem - :Finite Element Toolbox

communications - Communication Toolbox

compiler - MATLAB Compiler

control - Control System Toolbox

ident - System Identification Toolbox

images - Image Processing Toolbox

local - Preferences

nnet - Neural Network Toolbox

optim - Optimisation Toolbox

signal - Signal Processing Toolbox

simulink - model/system-based design

stateflow - Logic and Command Toolbox

stateflow/sfdemos - Stateflow Demonstrations

stats - Statistics Toolbox

symbolic - Symbolic Math Toolbox

La liste détaillée dépend de l'environnement de travail. Elle est fournie par la commande ver. [12]

Bibliographie :

[01] : Bernard Bayle ; « Robtique mobile » ; Ecole Nationale Supérieure de Physique de Strasbourg; Université Louis Pasteur; année 2008–2009; bernard@eavr.ustrasbg.fr.

[02] : «Mobile robot» ;www.wikipedia.com.

[03] : David Filliat ; « La robotique mobile » cours C10-2 ENSTA ; 2 Octobre 2004.

[04] : Jean Paul Laumond ; « La robotique mobile » ; Paris hermès science publication ; 2001.

[05] : Éric Beaudry ; «Planification de tâches pour robotique mobile »; Mémoire présenté au Département d'informatique en vue de l'obtention du grade de Maître ès sciences (M.Sc.) Sherbrooke, Québec, Canada, mai 2006

[06] : Nicolas Hudon; «Initiation à Matlab » ; Ecole Polytechnique de Montréal.

[07] : Leonard de Vinci; «Modélisation et navigation d'un robot mobile » ; Codex Atlanticus folio 147 v.a.

[08] : sng hong lian; «fuzzy logique control of an obstacle avoidance robot » ; département électroniques et communication.

[09] : Gilles Mauris «Capteurs Ultrasonores''Intéligent"Application à la représentation symbolique de mesures de distance par codage flou » ; thèse de doctora ; Université de Savoie ; Le 30 juin 1992.

[10] : Philippe Lucidarme; «Apprentissage et adaptation pour des ensembles de robots réactifs coopérants »; Université Montpellier II; thèse de doctoral; Le 7novembre2003.

[11] : Thomas Vallée, Murat Yıldızoğlu; « Présentation des algorithmes génétiques et de leurs applications en économie »; Université de Nantes, Université Montesquieu Bordeaux IV ; Décembre 2003.

[12] : Alfred A. Manuel; «Eléments de MATLAB » ; Département de la Physique de la Matière Condensée ; Université de Genève .

[13] : P. Ciarlet, E. Lunéville; «Notes introductives à Matlab » .

[14] : Christophe Besse; «Introduction à Matlab » ; Université Paul Sabatier.

[15] : Saadaoui -A, Sebkhaoui -M; «Réalisation D'un Simulateur Des Algorithmes Génétiques Et De Champ Du Potentiel Artificiel Pour La Planification D'une

Trajectoire du Robot KHEPERA» ; Thèse d'Ingéniorat ; département d'informatique ; Université de Laghouat.

[16] : Souquet Amédée, Radet Francois-Gérard; «ALGORITHMES GENETIQUES»; le 21/06/2004.