الجمهـــوريـــــة الجزائـــريـــــة الديمقـــراطيـــة الشـعبيـــة
République Algérienne Démocratique et Populaire
وزارة التـــعلـــيـــــم العـــــالـــــــي والبـــحـــــــث العـلمـــــي
Ministère de l'Enseignement Supérieure et de la Recherche Scientifique

جامعة وهران 2 محمد بن أحمد
Université d'Oran 2 Mohamed Ben Ahmed

معهد الصيانة و الأمن الصناعي
Institut de Maintenance et de Sécurité Industrielle

Département de maintenance en instrumentation

MÉMOIRE

Pour l'obtention du diplôme de Master

Filière : Génie industriel
Spécialité : Génie industriel

Thème

Simulation d'une tache d'un robot industriel

Présenté et soutenu publiquement par :

Nom : Mebrek **Prénom** : Naima

Nom : Zaaf **Prénom** : Sarra

Devant le jury composé de :

Nom et Prénom	Grade	Etablissement	Qualité
Mr. HASSINI Abdelatif	PR	IMSI-Univ d'oran 2	**Président**
Mr. KACIMI Abderrahmane	MCB	IMSI-Univ d'oran 2	**Encadreur**
Mme. LAZREG Malika	M.A.A	IMSI-Univ d'oran 2	**Examinateur**

Année 2018/2019

الجمهـــوريـــــة الجزائـــريـــــة الديمقـــراطيـــة الشـعبيـــة
République Algérienne Démocratique et Populaire
وزارة التـــعلـــيـــــم العـــــالـــــــي والبـحـــــــث العـلمـــــي
Ministère de l'Enseignement Supérieure et de la Recherche Scientifique

جامعة وهران 2 محمد بن أحمد
Université d'Oran 2 Mohamed Ben Ahmed

معهد الصيانة و الأمن الصناعي
Institut de Maintenance et de Sécurité Industrielle

Sommaire

Chapitre I : Introduction à la robotique industrielle.

Chapitre II : Outils mathématiques pour la planification des taches industrielles

Chapitre III: Programmation graphique des taches avec Robot Studio

Chapitre VI : La simulation avec RobotStudio

Glossaire des acronymes

DDL	Degrés de liberté
MGI	Le modèle géométrique inverse
MGD	Le modèle géométrique direct
PEL	La programmation en ligne
PHL	La programmation hors ligne
CAO	Conception et Fabrication Assistée par Ordinateur
FAO	Fabrication Assistée par Ordinateur
SCARA	Selective Compliance Assembly Robot Arm

Liste des tableaux

Liste des figures

Chapitre I : Introduction à la robotique industrielle

Chapitre II : outils mathématiques pour la planification des taches industrielles

Chapitre III : Programmation graphique des taches avec RobotStudio

Chapitre IV : La simulation avec RobotStudio

Nomenclature

Symbole	Désignation
P	Un point de l'extrémité
R0	Un repère lié au bâti
C^n	corps
j	L'articulation
R_j	Le repère
Z_j	L'axe porté par l'axe de l'articulation
X_j	L'axe porté par la perpendiculaire
α_j	L'angle entre les axes Z_{j-1} et Z_j correspondant a une rotation autour de X_{j-1}
d_j	La distance entre Z_{j-1} et Z_j
θ_j	L'angle entre les axes X_{j-1} et X_j correspondant a une rotation autour de Z_j
r_j	La distance entre X_{j-1} et X_j
q_j	La variable articulaire
${}^{j-1}_{j}T$	La matrice de transformation

X	Les coordonnées opérationnelles
q	Le vecteur des variables articulaires
$\dot{q}$	La vitesse articulaire
J(q)	La matrice jacobéenne
dq	La différentielle articulaire
R^E	Le Repère
n	Le vecteur unitaire
n_T	La matrice de passage

Résumé

La robotique peut être définie comme l'ensemble des techniques et études tendant à concevoir des systèmes mécaniques, informatiques ou mixtes, capables de se substituer à l'homme dans ses fonctions motrices, sensorielles et intellectuelles.

Le projet de fin d'étude s'intéresse sur la planification des tâches pour un robot Industriel type ABB avec simulation par le logiciel de programmation graphique d'ABB « Robot Studio »

Objectif de notre travail est de programmer, planifier et simuler des tâches industrielles les plus fréquentes et employées d'un robot manipulateur ABB.

Abstract

Robotics can be defined as the set of techniques and studies tending to design mechanical systems, computer or mixed, capable of replacing the man in his motor, sensory and intellectual functions.

The end-of-project project focuses on the task planning for an ABB type industrial robot with simulation by ABB's "RobotStudio" graphical programming software.

The aim of our work is to program, plan and simulate the most frequent and used industrial tasks of an ABB robot manipulator.

Dédicacés

A l'homme de ma vie, mon exemple éternel, mon soutien moral et source de joie et de bonheur, celui qui s'est toujours sacrifié pour me voir réussir, que dieu te garde dans son vaste paradis, à toi mon père.

A la lumière de mes jours, la source de mes efforts, la flamme de mon cœur, ma vie et mon bonheur ; maman que j'adore.

Aux personnes dont j'ai bien aimé la présence dans ce jour, à mon frère Sohaib et ma sœur hadjer, je dédie ce travail dont le grand plaisir leurs revient en premier lieu pour leurs conseils, aides, et encouragements.

Aux personnes qui m'ont toujours aidé et encouragé, qui étaient toujours à mes côtés, et qui m'ont accompagnaient durant mon chemin d'études supérieures, mes aimables amis, collègues d'étude, et frères de cœur

Dédicaces

Je dédie ce modeste travail

A ma mère ma raison d'être, ma raison de vivre, la lanterne qui éclaire mon chemin, et m'illumine de douceur et d'amour.

A mon père, en signe d'amour, de reconnaissance et de gratitude pour tous les soutiens et les sacrifices dont il a fait preuve a mon égard.

A mes chères frères Daoud, Hadjira, Aicha, Nour El houda

A mes oncles surtout mon oncle Belkacem et tous la famille MEBREK.

A tous les amies et témoignage de l'amitié sincère qui nous a liées et des bons moments passés ensemble: a mon binôme Zaaf Sarra Fouka Nour el houda , Ziddan Cherifa,

et Les familles: MEBREK et ZAAF.

MEBREK NAIMA

REMERCIEMENTS

En préambule à ce mémoire nous remerciant ALLAH qui nous aide et nous donne la patience et le courage durant ces langues années d'étude.

Nous souhaitant adresser nos remerciements les plus sincères aux personnes qui nous ont apporté leur aide et qui ont contribué à l'élaboration de ce mémoire ainsi qu'à la réussite de cette formidable année universitaire.

Nous tenant à remercier sincèrement Monsieur, **KACIMI ABDERRAHMAN**, qui, en tant que directeur de mémoire, se sont toujours montrés à l'écoute et très disponible tout au long de la réalisation de ce mémoire, ainsi pour l'inspiration, l'aide et le temps qu'ils ont bien voulu nous consacrer et sans qui ce mémoire n'aurait jamais vu le jour. On n'oublie pas nos parents pour leur contribution, leur soutien et leur patience. Enfin, nous adressons nos plus sincères remerciements à tous nos proches et amis, qui nous ont toujours encouragée au cours de la réalisation de ce mémoire.

Merci à tous et à toutes.

Zaaf sarra

&

Mebrek Naima

Introduction générale

L'être humain depuis son existence cherche à faciliter sa vie et à concevoir des objets et des méthodes qui l'aident à exploiter l'environnement extérieur, la terre, la mer et l'espace. De nos jours, parmi ces objets on trouve les robots et plus précisément les robots manipulateurs.

Généralement, un robot manipulateur est considéré comme un système articulé rigide c'est pour qu'il est développé dans le cadre d'application industrielle ; Ce dernier est prouvé leur importance puisqu'il substitue efficacement l'homme dans la réalisation des tâches tel que la soudure dans les usines automobiles ou la palettisation, assemblage..…

La modélisation et la simulation des systèmes robotiques à l'aide de divers logiciels de programmation facilite le processus de conception, de construction et inspectant les robots dans le monde réel. Une simulation est importante pour les programmeurs de robot dans leur permettant d'évaluer et de prédire le comportement d'un robot, et en outre de vérifier et d'optimiser la planification de trajectoire de leur processus. En outre, cela permettra d'économiser le temps et l'argent, et jouer un rôle important dans l'évaluation de la fabrication d'automatisation. Être capable de simuler ouvre une large gamme d'options, en aidant à résoudre de nombreux problèmes créative. On peut étudier, concevoir, visualiser et tester un objet avant de faire une réalité.

Donc , Le robot IRB 1600 du fabriquant ABB est un robot fiable, rapide, précis, puissant, robuste, flexible et intelligent, avec lequel on peut réaliser une grande multitude d'opérations sans aucun problème et avec une grande performance. Ce robot conçu exclusivement pour les industries manufacturières qui utilisent des installations robotisées.

Le travail réalisé et présenté dans ce mémoire s'articule de la façon suivante :

- L'objet du premier chapitre est d'apportera quelques définitions de base et décrire les constituants technologiques d'un robot et définir les principaux termes du domaine.
- Dans le deuxième chapitre en présente quelques méthodes permettant d'établir les modèles géométriques et cinématiques pour, ces méthodes sont basées sur la détermination des paramètres de Denavit-Hartenberg.
- Dans le chapitre trois, décrit les différents types de Programmation et plus particulièrement la programmation graphique muni du langage de programmation robotique RAPID, des taches avec le logiciel RobotStudio.
- Et finalement dans le quatrième chapitre présentera la simulation d'une tâches planifiée du robot industriel IRB 1600 de la compagnie ABB à l'aide du logiciel RobotStudio.

I. Introduction

Un robot est un système mécanique poly-articulé mû par des actionneurs et commandé par un calculateur qui est destiné à effectuer une grande variété de tâches.

Selon la norme internationale ISO 8373, un robot industriel (figure I.1) est un << manipulateur multi-application reprogrammable commandé automatiquement, programmable sur trois axes ou plus, qui peut être fixé sur place ou mobile, destiné à être utilisé dans des applications d'automatisation industrielle >>. Selon cette même norme, un manipulateur est une << machine dont le mécanisme est généralement composé d'une série de segments, articulés ou coulissants l'un par rapport à l'autre, ayant pour but de saisir et/ou de déplacer des objets (pièces ou outils) généralement suivant plusieurs degrés de liberté>>. La partie extrême du manipulateur qui porte l'outil (préhenseur, pince de soudage, etc.) s'appelle l'effecteur du robot. [1]

(a) (b)

Figure (I.1) : Robots à six et cinq ddl : (a) robot sériel à six articulation et (b) robot Sériel à cinq articulations.

I.1. Définitions générales [2]

La robotique est une science pluridisciplinaire qui comprend la mécanique, l'automatique l'électrotechnique, le traitement de signal, l'informatique, communication.....etc. Un robot se compose de :

I.1.1. Le mécanisme

Structure plus au moins proche de celle du bras humain, on dit aussi manipulateur quand il ne s'agit pas d'un robot mobile.
Sa motorisation est réalisée par des actionneurs électriques, pneumatiques ou hydrauliques qui transmettent leur mouvement aux articulations par des systèmes appropriés. [2]

I.1.2. La perception

Permet de gérer les relations entre le robot et son environnement. Les organes de Perception sont des capteurs dits « proprioceptifs » lorsqu'ils mesurent l'état interne du robot (position et vitesses des articulations) ou « extéroceptifs » lorsqu'ils recueillent des informations sur l'environnement (détection de présence, mesure de distance, vision artificielle). [2]

I.1.3. La commande

Qui synthétise les consignes des asservissements pilotant les actionneurs. A partir de la fonction de perception et des ordres de l'utilisateur, elle permet d'engendrer les actions du robot. [2]

I.1.4. L'interface homme-machine

A travers laquelle l'utilisateur programme les taches que le robot doit exécuter.

I.1.5. Le poste de travail et les dispositifs perirobotique

Qui constituent l'environnement dans lequel évolue le robot. [2]

I.2. Constituants d'un robot [3]

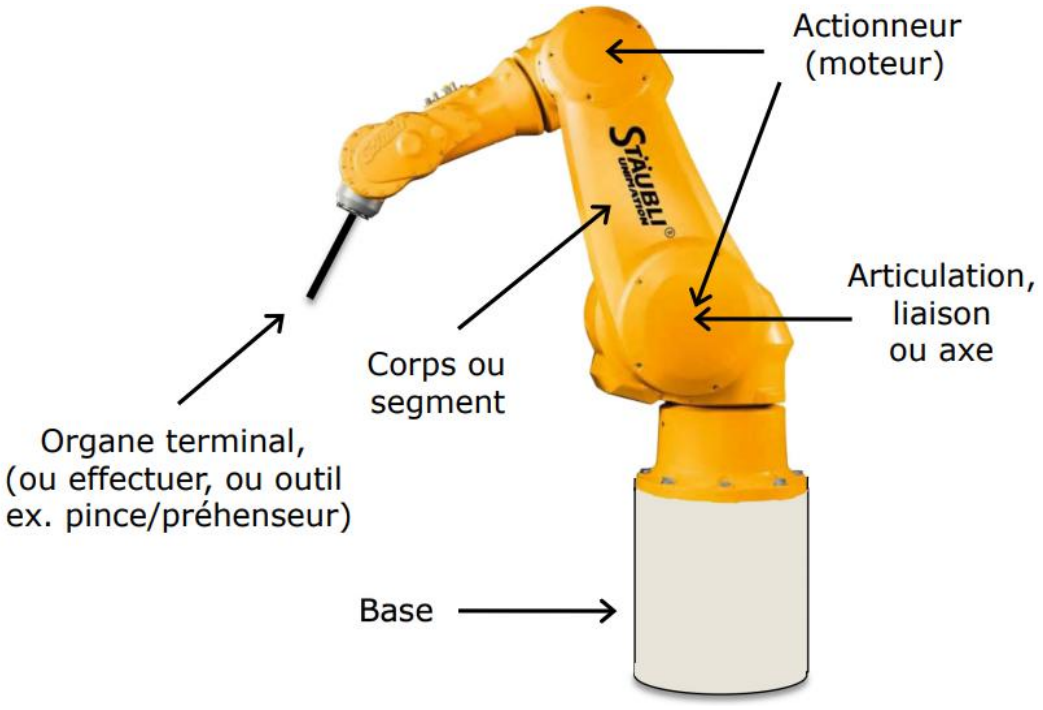


Figure (I.2) : Le vocabulaire de la robotique.

I.3. Caractérisation des robots [3]

I.3.1. Description de sa géométrie

Robot = système mécanique poly-articulé :

I.3.1.1. Articulation prismatique

Il s'agit d'une articulation de type glissière, notée P, réduisant le mouvement entre deux corps à une translation le long d'un axe commun. [4]

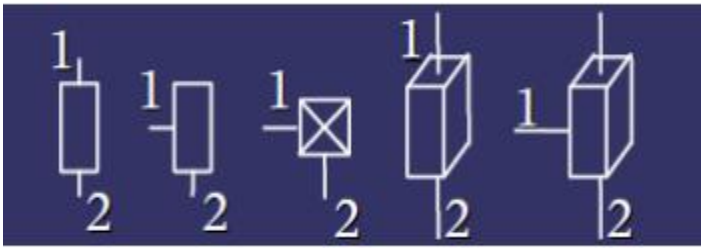


Figure (I.3) : Articulation prismatique.

I.3.1.2. Articulation rotoïde

Il s'agit d'une articulation de type pivot, notée R, réduisant le mouvement entre deux corps à une rotation autour d'un axe qui leur est commun. [4]

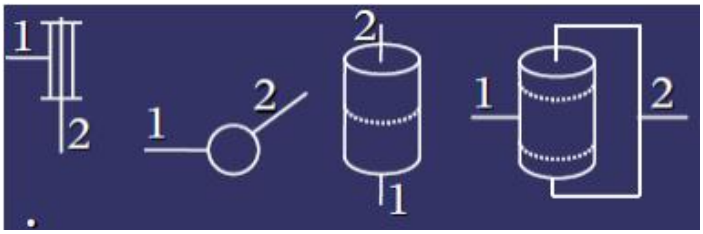


Figure (I.4) : Articulation rotoïde.

I.3.2. Caractéristiques géométriques

• Nombre d'axes (propulsai par un actionneur).

• Architecture (série ou parallèle).

• Chaînage des articulations.

• Nombre de degrés de liberté.

Exemples

– 3 axes, série, RRR, 3DL. [3]

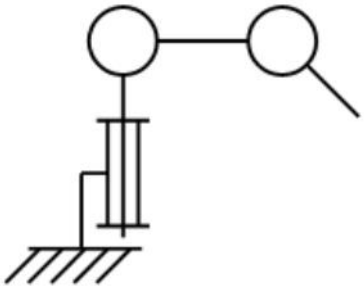


Figure (I.5) : Robot avec 3 axes, série, RRR et 3DL.

– 3 axes, série, PPP, 3DL.

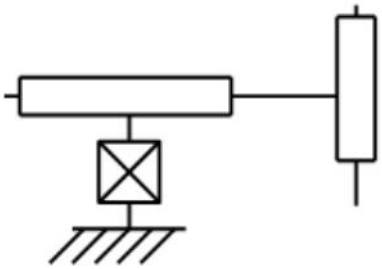


Figure (I.6) : Robot avec 3 axes, série, PPP, 3DL.

– 4 axes, parallèle, RP+RP, 3DL.

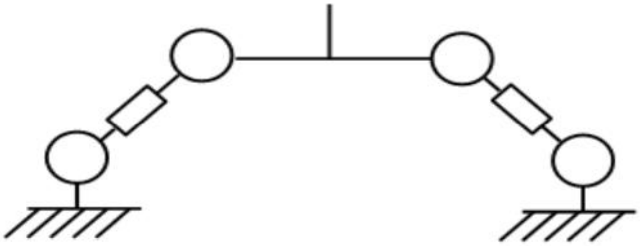


Figure (I.7) : Robot avec 4 axes, parallèle, RP+RP, 3DL.

I.3.3. Volume accessible par l'outil du robot

Ce volume dépend :

• de la géométrie du robot,

• de la longueur des segments,

• du débattement des articulations (limité par des butées) [3]

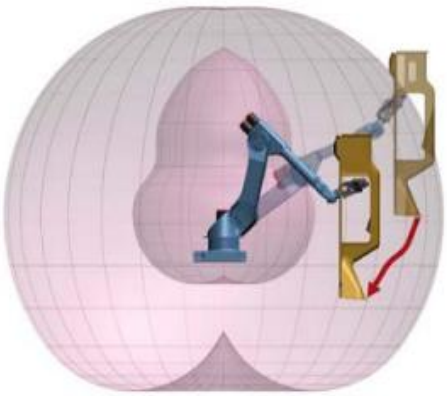


Figure (I.8) : Volume accessible par l'outil du robot.

I.3.4. Précision / Répétabilité

Positionnement absolu imprécis (>1 mm):

- Erreurs de modèle géométrique,
- Erreurs de quantification de la mesure de position,
- Flexibilités.

Répétabilité : la répétabilité d'un robot est l'erreur maximale de positionnement répété de l'outil en tout point de son espace de travail

• En général, la répétabilité < 0.1 mm. [3]

I.3.5. Performances dynamiques

I.3.5.1. Vitesse maximale :

Vitesse maximale de translation ou de rotation de chaque axe.

I.3.5.2. Accélération maximale :

Est donnée pour chaque axe dans la configuration la plus défavorable (inertie maximale, charge maximale).

Dépend fortement de l'inertie donc de la position du robot. [3]

I.3.6. Charge utile

• C'est la charge maximale que peut porter le robot sans dégrader la répétabilité et les performances dynamiques.

• La charge utile est nettement inférieure à la charge maximale que peut porter le robot qui est directement dépendante des actionneurs. [3]

I.3.7. Architecture des robots [2]

Un robot comporte 2 parties essentielles :

I.3.7.1. Le porteur

Le porteur représente l'essentiel du système mécanique articulé, il à pour rôle d'amener l'organe terminal dans une situation donnée imposée par la tache (la situation d'un corps peut être définie comme la position et l'orientation d'un repère attaché à ce corps par rapport à un repère de référence). [2]

I.3.7.2. Le poignet

Il est destiné à l'orientation de la pince ou de l'outil porté par le robot. La façon dont les liaisons motorisées sont reparties du bâti au poignet défini trois grandes classes d'architecture:

- Architecture série (ou chaine cinématique ouverte)
- Architecture parallèle (ou chaine cinématique multi-boucle
- Architecture mixte (série-parallèle ou parallèle-série). [2]

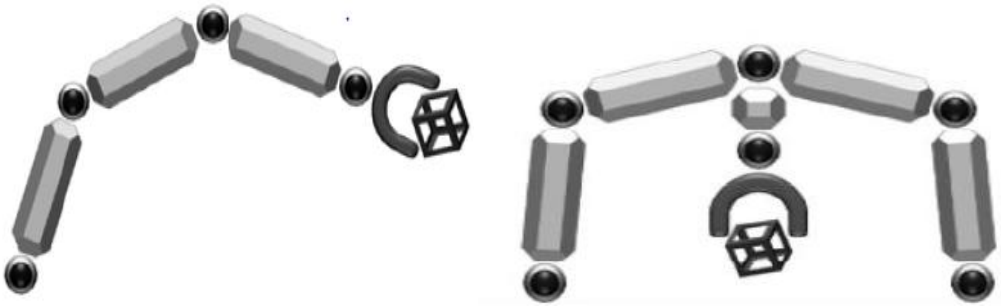


Figure (I.9) : Structure ouverte simple (a), structure arborescente (b).

I.4. Description et types de robots

Dans l'espace tridimensionnel, un corps rigide libre peut se déplacer selon six degrés de libertés (ddl) :

Trois translations et trois rotations. On utilise le terme pose pour désigner la localisation du corps par rapport à un référentiel. Une pose est composée d'une position et d'une orientation.

Pour placer un corps rigide n'importe où dans l'espace tridimensionnel, on a besoin d'un robot avec minimum six articulations motorisées, c.à.d. d'un robot à six ddl figure (I.1) (a). La grande majorité des robots industriels sont de type sériel. [5]

I.4.1. Le robot sériel

Un robot sériel est composé d'une série de segments reliés par des articulations motorisées rotoïde (en rotation) ou prismatiques (en translation). Dans certaines applications, on n'a pas besoin de déplacer les objets selon six ddl mais seulement selon cinq ou même quatre ou trois ddl. La figure (I.1) (b) illustre un robot sériel à cinq ddl utilisé pour la palettisation.

Le robot de la figure (1.9) est lui aussi utilisé pour la palettisation, mais il a seulement quatre ddl. [5]

Figure (I.10) : Robot sériel pour palettisation à quatre ddl.

I.4.2. Le robot parallèle

Dans un robot parallèle, l'effecteur est relié à la base via plusieurs < bras >, et la plupart des articulations ne sont pas motorisées. Les robots parallèles peuvent eux aussi avoir six, cinq, quatre, trois ou même deux ddl. Les robots parallèles à six ddl les plus connus sont les hexapodes, comme ceux qui déplacent les cockpits des simulateurs de vol. Les robots parallèles sont généralement plus rigides et plus rapides que les robots sériels. En revanche, ils sont beaucoup plus difficiles à étudier et il en existe de milliers d'architectures différentes. [5]

Figure (I.11): Un robot parallèle.

I.4.3. Le robot cartésien

3 articulations prismatiques dont les axes sont typiquement mutuellement orthogonaux

(PPP), 3 DDL :

- La structure cartésienne offre une très bonne rigidité mécanique et une grande précision
- Cependant, la structure présente une faible dextérité car toutes les articulations sont prismatiques
- Utilisation typique: manutention et assemblage. [6]

Figure (I.12) : un robot cartésien

I.4.4. Manipulateur SCARA

SCARA: Selective Compliance Assembly Robot Arm

- Deux articulations rotoïde et une articulation prismatique (RRP): tous les axes sont parallèles;
  3 DDL
- Rigidité élevée pour charges verticales et souplesse aux charges horizontales
- Bien adapté à des tâches de montage vertical et à la manipulation de petits objets
- Précis et très rapide (1er modèle: 1981) [6]

Figure (I.13): Robots SCARA

I.4.5. Les robots redondants à sept ddl

Enfin, il existe aussi des robots à sept ddl (c.à.d. avec sept articulations motorisées) comme le robot illustré à la figure (I.14) (a), mais ils sont rarement utilisés. L'avantage d'un tel robot redondant est l'existence d'une infinité de possibilités pour atteindre une pose désirée, ce qui permet au robot de contourner des obstacles. Beaucoup plus souvent, on monte un robot a six articulations sur un guide linéaire la figure (I.14) (b) ou sur une table pivotante, ce qui résulte aussi en un système robotique redondant.

Cependant, la raison principale n'est pas de contourner des obstacles mais d'augmenter l'espace de travail du robot (l'ensemble de poses que l'effecteur du robot peut atteindre). [5]

Figure (I.14): Robots redondants.

I.5.Historique des robots industriels

Un des premiers robots industriels à être conçu par l'américain Willard L. G. Pollard Jr. qui a fait une demande de brevet pour son invention en 1934 [7]. Il s'agit d'un robot parallèle à deux ddl destiné à l'application de peinture sur la carrosserie d'une automobile. Une licence de ce brevet à été vendue à la compagnie DeVilbiss en 1937.

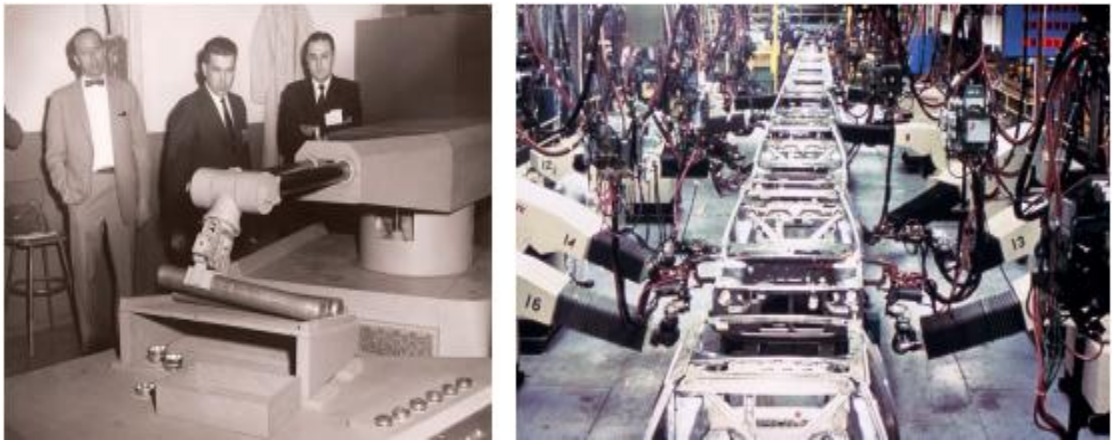
En 1941, DeVilbiss a fabriqué le premier robot industriel (un robot de peinture) sous la direction d'Harold Roselund. Ce robot n'était pas le robot de Pollard, mais un robot sériel inventé par M. Roselund lui-même [8].

De toute évidence, les inventions des messieurs Pollard et de M. Roselund n'ont pas eu l'effet désiré. C'est beaucoup plus tard, en 1954 que l'américain George Devol fait la demande d'un brevet pour un robot de transfert qui va donner naissance à la robotique industrielle, telle qu'on la connait aujourd'hui.

En 1956 M. Devol rencontre M. Joseph Engelberger [9], un jeune ingénieur qui travaillait dans l'industrie spatiale, et lui fait part de son invention. Engelberger devient rapidement passionné par l'invention de Devol et démarre la compagnie Unimation.

Entre temps, la compagnie AMF (American Machine and Foundry) introduit le robot Versatran, le premier robot cylindrique. En 1962, six robots Versatran ont été installés dans une usine de Ford. C'est en 1961 qu'Unimation développe leur premier prototype, l'Unimate, et le vend (à forte perte) à General Motors figure (I.15) (a). Ce robot a été utilisé pour assister une machine à coulée par injection.

En 1969, Unimation installe 26 robots soudeurs sur une ligne d'assemblage de Chevrolet Vega, chez General Motors figure (I.15) (b).



(a) (b)

Figure (I.15) : Les premiers robots industriels (a) le premier robot Unimate juste avant son entrée en fonction en 1961 et (b) la première ligne robotisée qui consiste en 26 robots soudeurs Unimate.

En 1969, Victor Scheinman développe le Standford Arm à l'Université Standford. Il s'agit de l'architecture qui est aujourd'hui utilisée par presque tous les robots sériels à six ddl.

En 1973, Scheinman démarre sa propre entreprise et commercialise son design sous le nom Vicarm. Unimation achète le Vicarm en 1977, l'améliore, et le renomme PUMA pour Programmable Universal Machine for Assembly figure (I.16). Le premier robot PUMA est installé en 1979, dans une usine de General Motors. Unimation finit par être vendu à l'entreprise suisse Staubli en 1989.

Figure (I.16) : Les trois premiers modèles du robot PUMA commercialisés par Unimation.

Aux débuts des années 1970, plusieurs grandes entreprises se lancent dans la fabrication de robots industriels [10]. Asea (aujourd'hui ABB), après avoir installé quelques 25 robots Unimate, développe son propre robot à six ddl en 1973, l'IRB 6, le premier robot à six ddl avec un parallélogramme figure (I.17) (a).

Le parallélogramme sert à placer le troisième moteur plus proche de la base et ainsi diminuer l'inertie du robot. La même année, le fabricant allemand KUKA développe son propre robot à six ddl, le FAMULUS.

En 1980, 19 000 robots industriels ont été fabriqués au Japon par quelques 150 fabricants, dont Kawasaki, Yaskawa, Kitachi, Mitsubishi Heavy Industries, Fanuc et Nachi. Le Japon devient le plus grand fabricant et utilisateur de robots industriels.

En 1978, Hiroshi Makino, à l'université de Yamanashi, invente le robot SCARA (Selective Compliance Assembly Robot Arm) [11]. Sankyo Seiki et Nitto Seiko sont les premiers fabricants a produire des versions commerciales de ce robot, en 1981 figure

(I.17)(c).

(a) (b) (c)

Figure (I.17) : Les premiers robots de la nouvelle ère de la robotique industrielle

En 2012 [12] quelque 159 346 robots industriels ont été installés à travers le monde.

En cinquante ans, plus de 2 470 000 robots industriels ont été installés, dont au moins la moitié sont encore en service (la durée de vie utile d'un robot industriel est environ douze ans).

Le club des fabricants de robots a peu changé en quarante ans (depuis l'entrée en jeu d'ABB, KUKA, Fanuc et Yaskawa). L'entreprise japonaise Fanuc a toujours été le plus important fabricant de robots, bien évidemment, les robots d'aujourd'hui sont nettement plus performants.

Les avances en vision robotique sont aussi spectaculaires. Enfin, de plus en plus la programmation des robots se fait à l'aide de logiciels de simulation très avancés. Par contre, il n'existe toujours pas de langage de programmation commun et la programmation d'un robot industriel reste assez difficile. De plus, même le plus petit robot industriel reste assez dangereux pour l'humain et nécessite l'installation de dispositifs de sécurité dispendieux.[5]

I .6. Utilisation des robots [13]

I .6.1. Tâches simples

- La grande majorité des robots est utilisée pour des tâches simples et répétitives.

- Les robots sont programmés une fois pour toute au cours de la procédure d'apprentissage.

-Critères de choix de la solution robotique:

- La tâche est assez simple pour être robotisée.
- Les critères de qualité sur la tâche sont importants.
- Pénibilité de la tâche (peinture, charge lourde, environnement hostile, ...).

Figure (I.18): Robots de peinture dans l'industrie Automobile.

Figure (I.19): Robots de soudure dans l'industrie Automobile.

I .6.2. Tâches complexes

C'est dans le cas où le robot effectue des taches sensibles et qui nécessite plus de précision et robustesse et on distingue notamment la robotique de service.

Figure (I.20): Robot pompiste

Figure (I.21): Robot de construction

Parmi les applications les plus communes, ABB recense le soudage à l'arc, le soudage par point, la manutention, le chargement/déchargement de machines outils, la mise en peinture, le transfert de pièces (palettisation, emboîtage, etc.), l'assemblage, l'enlèvement de matière (ébavurage, polissage, etc.), et l'application de colle ou de scellant. Les robots industriels sont

aussi utilisés dans des endroits inhabituels tels que les parcs d'amusement, les hôpitaux, et même les restaurants. [5]

I.7. Le robot industriel IRB 1600 de la compagnie ABB

Le robot industriel que nous allons utiliser est le modèle IRB 1600 du fabricant ABB.

Il est conçu spécifiquement pour les industries manufacturières qui utilisent des installations robotisées flexibles. Le robot a une structure ouverte spécialement adaptée a une utilisation souple. Le système est composé d'un manipulateur sériel à six ddl et d'un contrôleur IRC5. [5]

Figure (I.22) : Le manipulateur IRB 1600 et le contrôleur IRC5

I.7.1 Le manipulateur

Le modèle 3D du manipulateur est montré à la figure (I.23), il s'agit d'un manipulateur à six articulations rotoïde actionnées par des moteurs AC. La lecture de l'angle de rotation de chaque articulation est fournie par un résolveur directement monté sur l'arbre de chacun des moteurs. La rotation de chaque articulation est connue de façon absolue sur un tour de moteur, mais un système électronique mémorise le nombre de tours. Ce système est situé dans la base du robot et protégé par une batterie lors de la mise hors tension du système. [5]

Figure (I.23) : Modèle 3D du robot IRB 1600.

I.7.2. Espace de travail et performance du robot

Nous allons appeler l'ensemble de poses que l'effecteur du robot peut atteindre l'espace de travail. L'enveloppe de travail d'un robot sériel avec des articulations rotoïde est définie par les longueurs des segments, la disposition des axes des articulations, les limites des articulations et les interférences mécaniques entre les différents segments. La plage de déplacement et la vitesse maximale pour chaque articulation sont montrées au tableau (I.1) : [5]

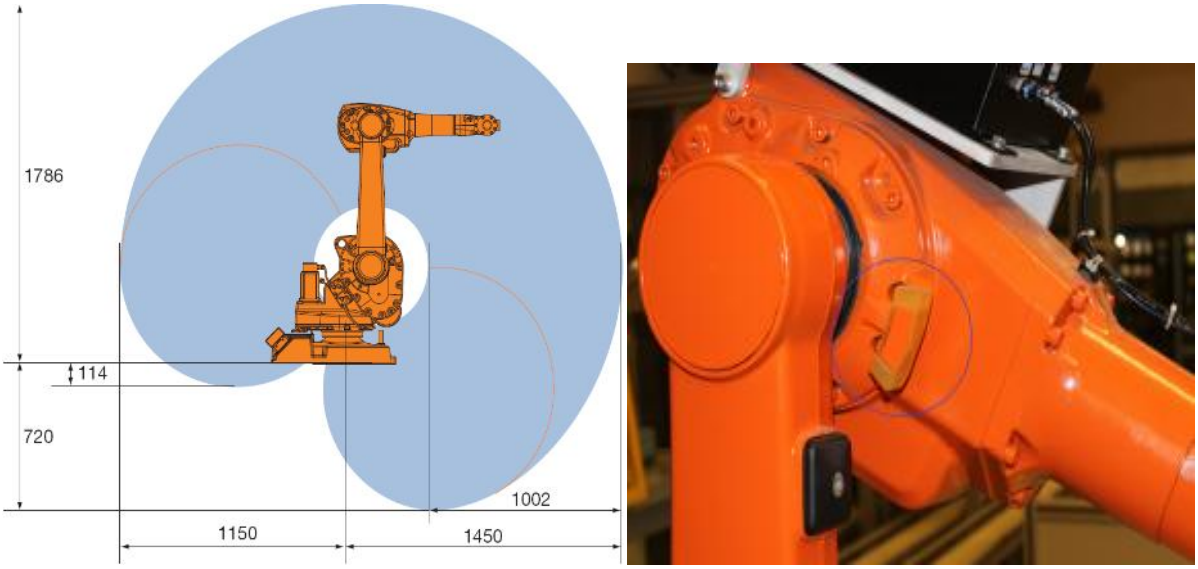
Articulation	Plage (par défaut)	Vitesse
1	+ ou - 180 °	150°/S
2	-90° à 150°	160°/S
3	-245° à 65 °	170°/S
4	+ou - 200°	320°/S
5	+ou - 115°	400°/S
6	+ ou - 400 °	460°/S

Tableau (I.1): Capacité des articulations du robot IRB1600.

La figure (I.24) (a) montre une section verticale de ce qu'on appelle souvent l'enveloppe de travail du robot IRB 1600. Il s'agit simplement de la zone atteignable par le centre du poignet du robot (le point d'intersection des axes des trois dernières articulations). L'espace de travail du robot est quant a lui une entité géométrique à six dimensions fort complexe et ne peut pas être représenté graphiquement. De plus, l'espace de travail du robot dépend de l'outil utilisé.

Les articulations des robots au local A-0610 ont des plages de travail plus restrictives que celles fournies par le fabricant afin de réduire le risque de collisions. Il s'agit simplement de

butées programmées dans le contrôleur. Cependant, afin d'être conforme aux normes et éviter les bris, les robots possèdent sur leurs articulations des butées mécaniques, comme illustré à la figure (I.24) (b). [5]



(a) (b)

Figure (I.24) : (a) Section verticale de l'enveloppe de travail du robot IRB 1600 et (b) une butée mécanique sur ce même robot.

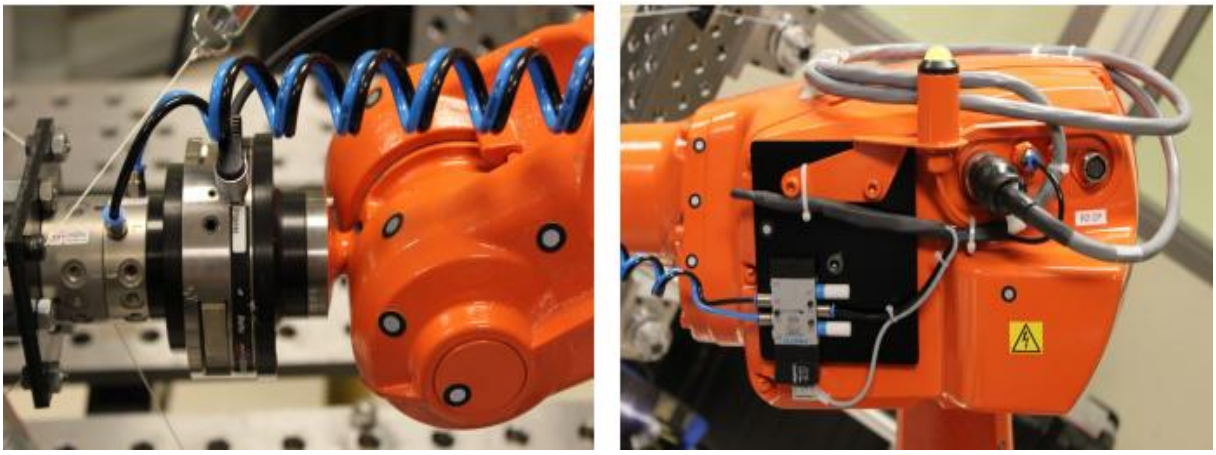
I.7.3. Description des composants principaux

I.7.3.1. La bride de montage

Par définition, un robot industriel est une machine multifonction et rarement vendu avec de l'outillage. Par conséquent, l'extrémité de chaque robot industriel est munie d'une bride de montage sur laquelle on peut fixer un outil tel qu'un préhenseur ou une pince de soudage par point. Généralement, une bride de montage consiste en une plaque circulaire avec au moins quatre trous filetés et des trous de localisation. Lorsque le fabricant d'un robot industriel parle de la charge utile du robot (5 kg dans le cas du robot IRB 1600), il s'agit de la masse maximale de tout ce qui est fixé à sa bride. La figure(I.25) (a) montre un autre exemple de changeur d'outil monté sur le robot IRB 1600. [5]

I.7.3.2. Les conduits pour l'effecteur

Le plus souvent, l'outil installé sur la bride de montage aura besoin de conduits électriques et/ou pneumatiques. Ainsi, la plupart des robots industriels auront quelques conduits électriques et pneumatiques qui partent de la base du robot, passent à l'intérieur des membrures et ressortant vers l'extrémité du bras. Dans le cas du robot IRB 1600, il s'agit d'un conduit pneumatique (maximum 8 bars) et de 28 conduits électriques (maximum 50 mA par conduit) comme montré à la figure (I.25) (b). Par la suite, il en revient au concepteur de la cellule robotique de faire le lien entre les connecteurs de raccord et l'outil. La plus grande difficulté ici est de mettre en place une conduite flexible répondant à tous les besoins de mouvement sans toutefois risquer de coincer et/ou se faire arracher durant un mouvement.[5]



(a) (b)

Figure (I.25): L'effecteur du robot IRB 1600

I.7.3.3. La base

La base du robot doit être fixée sur une surface très rigide. Il est important de s'assurer de répondre aux normes du fabricant pour la méthode de fixation, afin d'assurer la sécurité. Le robot génère des forces très élevées lors d'un arrêt d'urgence. Le torque généré peut être plusieurs fois plus élevé que le torque produit durant le fonctionnement normal. [5]

I.7.3.4. Freins électriques

Pour des raisons de sécurité, il est strictement interdit d'actionner les boutons de relâchement mécanique des freins figure(I.26) des robots IRB 1600. Notez que le poids des deux bras est d'environ 50 kg. En plus de danger de blessures graves, il est également possible de briser le robot, son effecteur et les équipements autour de lui (glissoire, etc..).[5]

Figure (I.26): Bouton pour relâcher les freins à ne jamais actionner.

I.7.4. La répétabilité

Les robots industriels sont capables de répéter un mouvement avec très peu de variation. Lorsqu'un robot essaie d'amener son effecteur à une même pose plusieurs fois, il existe une formule statistique pour calculer ce qu'on appelle la répétabilité. Dans le cas de la répétabilité en position, on peut dire de façon simpliste que la répétabilité est l'erreur maximale en position lorsque le robot va essayer d'amener son effecteur à une pose, pour une deuxième fois. Si le chemin emprunté est toujours le même, on parle de la répétabilité unidirectionnelle. Sinon, on parle de la répétabilité multidirectionnelle.

ABB spécifie que la répétabilité du robot IRB 1600 est de ±0.050 mm. Au fait, il s'agit de la répétabilité unidirectionnelle. [5]

I.7.5. Le contrôleur IRC5 et le FlexPendant

I.7.5.1. Le contrôleur IRC5

Le contrôleur IRC5 est le plus récent contrôleur de robot de la compagnie ABB. Il peut contrôler jusqu'à quatre robots en mode synchrone ou non. ABB utilise un seul contrôleur pour tous ses modèles de robots. Le format du boitier est offert en plusieurs modèles et les variateurs dans le contrôleur sont évidemment différents d'un robot à l'autre.

Le contrôleur utilise un système d'opération en temps réel appelé RobotWare qui pourra interpréter et exécuter les programmes implantés. Le contrôleur est principalement composé des items suivants figure (I.27) :

- Un ordinateur industriel de type Pentium 4 cadencé à 1.4 GHz, protégé par une carte Compact Flash de 256 Mo. Cette carte s'assure de préserver la mémoire lors d'une perte de tension dans le système. Ceci permet au robot de ne perdre aucune information importante, et ce même s'il était en pleine exécution d'un mouvement.
- Une carte de contrôle des axes (Motorola PowerPC 250 MHz) ;
- Un groupe puissance permettant le contrôle des moteurs (adapté au type de robot) ;
- Une carte d'entrées/sorties de type TOR 24 Vdc permettant l'interaction entre les équipements sur la table, l'effecteur et le robot. [5]

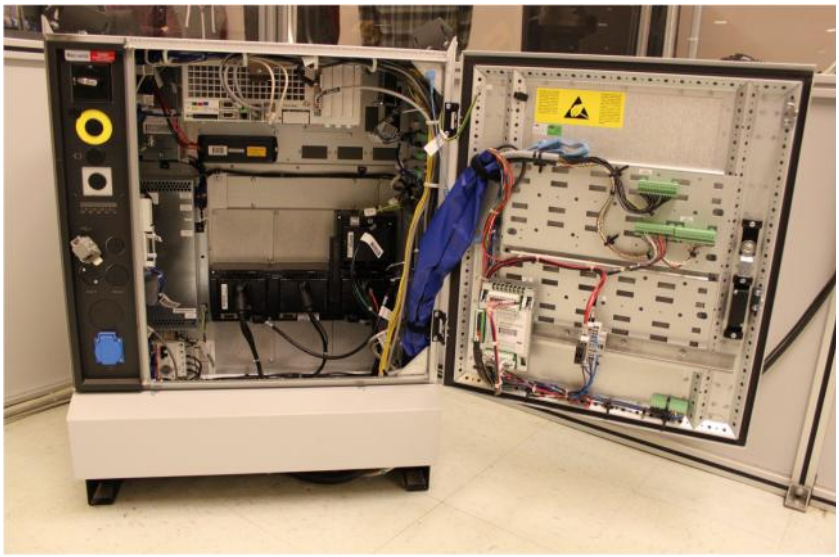


Figure (I.27): Contrôleur IRC5 standard de la compagnie ABB

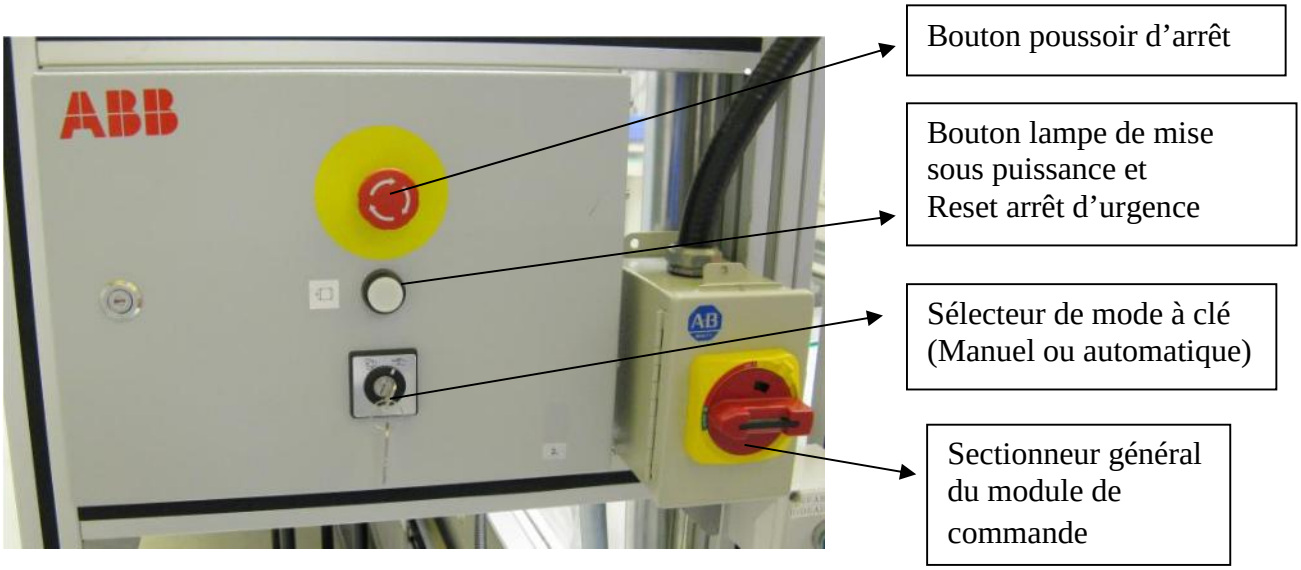


Figure (I.28) : panneau opérateur externe d'un des quatre robots au laboratoire A-0610.

I.7.5.2. Le boitier de commande

Le boitier de commande appelé FlexPendant par ABB permet à l'opérateur d'interagir avec le contrôleur du robot figure (I.29). Le FlexPendant est en fait un ordinateur avec écran tactile de sept pouces avec une manette 3D et des boutons. Il communique avec le contrôleur par un réseau Ethernet privé et fonctionne sous un système d'exploitation WindowsCE. Ceci a deux avantages : Il est possible de personnaliser les écrans en fonction du projet avec l'option ScreenMaker et voire même ne pas avoir le boitier relié en tout temps avec le robot.[5]

Figure (I.29): Boiter de commande FlexPendant de ABB.

Voici quelques règles et informations concernant le FlexPendant :

- ✓ Le boitier de commande est relié au contrôleur par un câble multibrin relativement fragile. Il est important de ne pas le forcer, le coincer, l'écraser sous les souliers ou encore tirer dessus.
- ✓ Il est interdit d'utiliser des objets pour toucher l'écran tactile. Il faut plutôt utiliser les doigts.
- ✓ Dans le coin du boitier de commande, nous retrouvons un arrêt d'urgence de type bouton < pousser tourner >. C'est ce bouton qui garantit la complète sécurité du programmeur quand il entre dans une cellule.
- ✓ Sur le coté du boitier de commande, nous retrouvons une gâchette < homme mort>. Cette gâchette est composée de deux interrupteurs, à trois positions, en parallèle. La position du milieu active le robot.
- ✓ Pour piloter le robot en mode manuel, il faut utiliser la manette 3D.
- ✓ Enfin, nous retrouvons des boutons de contrôle sur le devant du boitier : départ, arrêt, avancement pas-à-pas ou recul pas-à-pas. Il est important de comprendre que l'arrêt est contrôlé par le système et fait en douceur, alors que l'arrêt d'urgence et la gâchette < homme-mort > arrêtent le système de manière brusque en activant immédiatement les freins, ce qui use le robot. [5]

I.8. Les avantages de la robotisation industrielle

Les avantages de la robotisation d'une entreprise se déclinent autour de 3 critères principaux :

I.8.1. Facteurs économiques

•Réduction des coûts de main-d'œuvre :

- ✓ un robot est capable de travailler en 3x8 de façon **constante** et peut réaliser à lui seul les tâches de différents opérateurs
- ✓ Après programmation, un robot peut **travailler seul** la nuit et le week-end

•Flexibilité de la gestion de production :

- ✓ Un robot s'adapte à différentes tâches et peut donc aisément être affecté à **des opérations multiples**, selon les impératifs de l'entreprise

•Amélioration de la qualité :

- ✓ Les robots industriels ont la capacité de reproduire une même **tâche répétitive sans dégradation des performances**

I.8.2. Facteurs humains

- Augmentation de la sécurité sur le poste de travail, flexibilité, régularité et qualité conduisent à l'optimisation de la gestion de production et à la compression des délais de fabrication et de livraison
- Le robot soulage les opérateurs sur les postes de travail contraignants, fatigants, dangereux qui peuvent impacter leur santé, par exemple : les TMS (troubles musculo-squelettiques).

I.8.3. Facteurs environnementaux

- Grâce à l'optimisation des procès, la robotisation d'une entreprise permet une **économie énergétique** substantielle, notamment en terme de:

✓ Matières premières
✓ Déchets
✓ Rejets nocifs pour l'environnement

Conclusion

Dans ce chapitre on a présenté quelques définitions du robot industriel, et leur historique en industrie et les différents champs d'applications. Pour compléter ces généralités, nous présentons dans le chapitre suivant les notions et les outils mathématiques mis en œuvre pour la modélisation des robots.

II. Introduction

La conception et la commande des systèmes mécanique articulés nécessitent le calcul d'un modèle mathématique pour le but de le représenter dans l'environnement où il évolue. Cette modélisation consiste à exprimer d'une part :

- La situation de l'organe terminal et la transformation entre l'espace articulaire et l'espace opérationnel, c'est le modèle géométrique.

-La vitesse de l'organe terminal, on parle du modèle cinématique.

- Les couples et les forces exercés par les actionneurs de l'autre part, c'est le modèle dynamique.

II.1 La modélisation

Un manipulateur est constitué de deux sous-ensembles distincts : un (ou plusieurs) organe terminal et une structure mécanique articulée. L'organe terminal est le dispositif d'interaction, fixé à l'extrémité mobile de la structure mécanique. Il est désigné par différentes appellation : préhenseur, outil, effecteur, organe terminal (OT) ou pince lorsqu'il s'agit d'une pince, la structure cinématique articulée est une chaine cinématique de corps, généralement rigides, assemblées par des articulations. Le rôle de la structure mécanique articulée est d'amener l'organe terminal à un emplacement de position et d'orientation donnée, selon des requêtes spécifique de vitesse et d'accélération.

II.1.1 Modélisation géométrique

La modélisation des robots de façon systématique et automatique exige une méthode adéquate pour la description de leur morphologie. Plusieurs méthodes et notations ont été proposées. La plus répandue est celle de Denavit-Hardenberg [14]. Dans les années 1950, les messieurs Jacques Denavit et Richard Hartenberg ont eu l'excellente idée de proposer une méthode simple et systématique pour placer des référentiels sur chaque lien d'un mécanisme sériel qui facilite énormément le calcul des matrices de transformation homogène. [15]

II.1.1.1. Méthode de Denavit-Hartenberg

II.1.1.1.1.Notations

On numérote les solides par ordre croissant en partant du socle. Ainsi le robot est composé de n+1corps, notés C0,…, Cn et de n articulations ($n \geq 1$). Le corps C0 désigne le socle (la base) du robot, le corps Cn le corps portant l'organe terminal.

Le repère Rj est lié au corps Cj du robot.
La variable de l'articulation j, qui lie le corps Cj au corps $Cj-1$, est notée qj. [16]

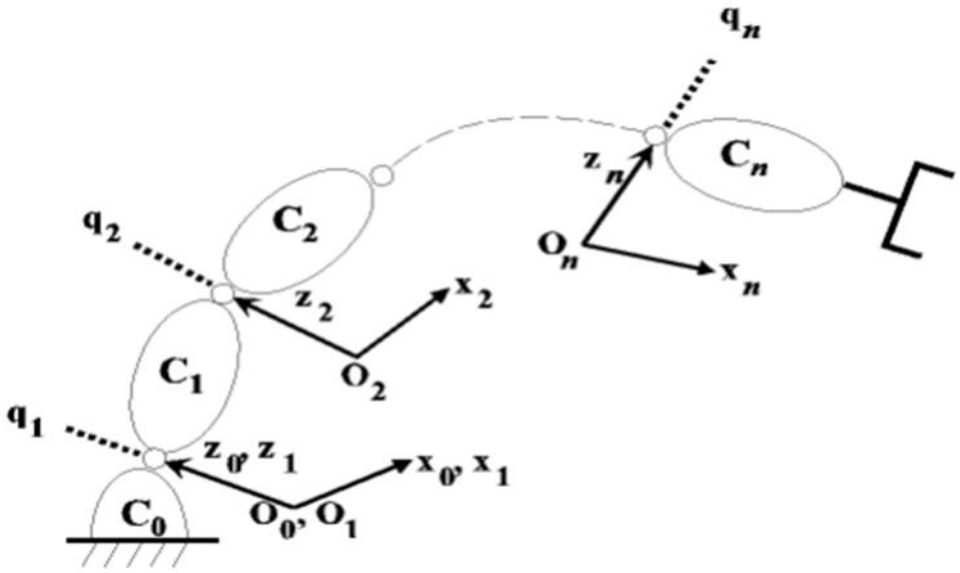


Figure (II.1) : Chaine ouverte simple

II.1.1.1.2. Principe

-Fixer des repères à chaque corps du robot.

-Calculer les matrices homogènes entre chaque corps.

-Calculer la matrice homogène entre base et organe terminal.

II.1.1.1.3. Hypothèse

On suppose que le robot est constitué d'un chainage de n+1 corps liés entre eux par n articulations rotoïde ou prismatiques. A chaque corps, on associe un repère Ri.

Les repères sont numérotes de 0 à n. La i ème articulation, dont la position est notée *qi*, est le point qui relie les corps i-1 et i.

Le repère Rj fixé au corps Cj, est défini de sorte que :

-L'axe Zj est porté par l'axe de l'articulation j.

-L'axe Xj est porté par la perpendiculaire commune aux axes Zj et Zj+1 .Si les axes Zj et Zj+1 sont parallèles ou colinéaires, le choix de Xi n'est pas unique.

II.1.1.1.4 .Les paramètres de Denavit-Hartenberg

Le repère Rj fixé au corps Cj est défini de telle sorte que :

– Zj est porté par l'axe articulaire j.

– Xj est porté par la perpendiculaire commune aux axes Zj et Zj+1.

Si Zj et Zj+1 sont parallèles ou colinéaires, le choix de Xj n'est pas unique. Dans ce cas, des considérations de symétrie ou de simplicité permettent alors un choix rationnel.

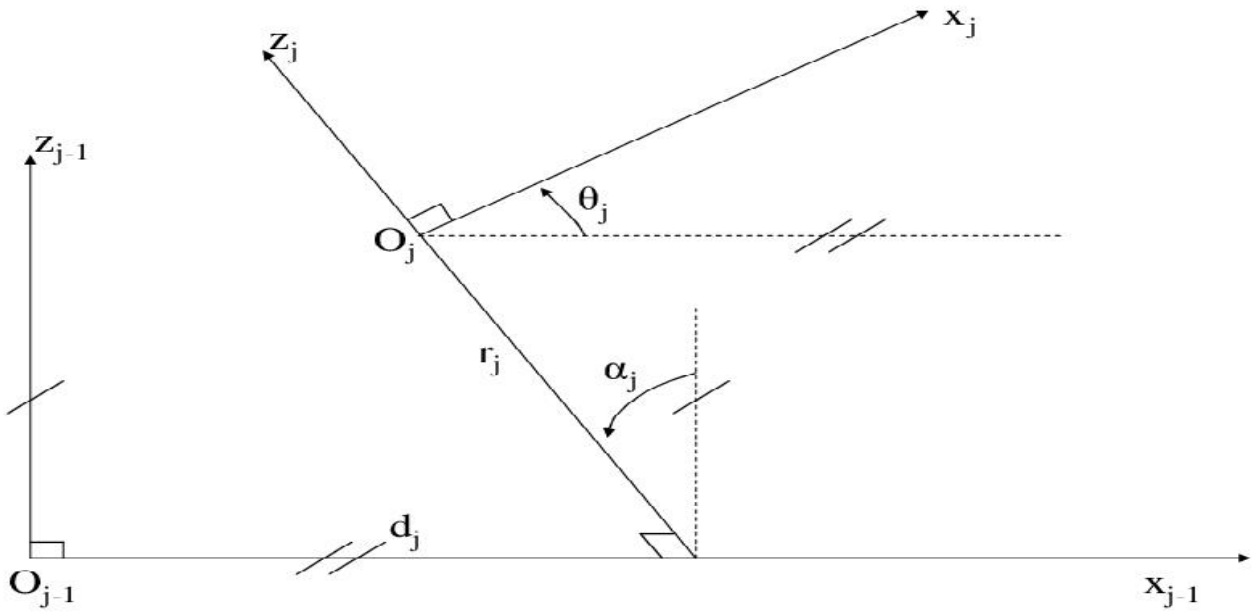
Le passage du repère Rj−1 au repère Rj, s'exprime en fonction de 4 paramètres suivants:

– αj est l'angle entre les axes Zj−1 et Zj correspondant à une rotation autour de Xj−1.

– dj est la distance entre les axes Zj−1 et Zj le long de Xj−1.

– θj est l'angle entre les axes Xj−1 et Xj correspondant à une rotation autour de Zj.

– rj est la distance entre les axes Xj−1 et Xj correspondant à une rotation autour de Zj. [16]



Figure(II.2) : Paramètres géométriques dans le cas d'une structure ouverte simple

La variable articulaire qj (associée à la j ème articulation) est soit :

– θj si l'articulation est de type rotoïde (σj = 0)

– rj si l'articulation est de type prismatique (σj = 1)

On a donc : $qj = \sigma j * \theta j + \sigma j * rj$ (II.1)

La matrice de transformation homogène définissant le repère Rj dans le repère Rj−1 est donnée par :

$$Tj^{j-1} = Rot(x, \alpha j).Trans(x, dj).Rot(z, \theta j).Trans(z, rj) \quad \text{(II.2)}$$

$$Tj^{j-1} = \begin{pmatrix} \cos .(\bar{}j) & -\sin(\bar{}j) & 0 & dj \\ \cos(\sim j).\sin(\bar{}j) & \cos(\sim j).\cos(\bar{}j) & -\sin(\sim j) & -rj.\sin(\sim j) \\ \sin(\sim j).\sin(\bar{}j) & \sin(\sim j).\cos(\bar{\cup}j) & \cos(\sim j) & rj.\cos(\sim j) \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad \text{(II.3)}$$

On remarque que la matrice de rotation (3x3) Aj^{j-1}peut être obtenue par :

$$Aj^{j-1} = Rot(x, \alpha j).Rot(z, \theta j) \quad \text{(II. 4)}$$

La matrice de transformation définissant R j-1 dans R j est donnée par :

$$Tj = \begin{pmatrix} & & & -dj.\cos(\bar{}j) \\ & A^{t}_{j-1,j} & & dj.\sin(\bar{}j) \\ & & & -rj \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad \text{(II.5)}$$

II.1.1.2.Modèle géométrique direct

Le modèle géométrique direct (MGD) d'un bras manipulateur exprime la situation de son Organe Terminal (OT) en fonction de sa configuration. Donc il exprime la position instantanée de l'OT par rapport au repère fixe lié au bâti.

$$X_i = f(q_i) \quad \text{(II.6)}$$

Le modèle géométrique direct (MDG) il peut être représenté par la matrice de passage T_n^0

$$Tn^{0} = T_1(q_1)T_2{}^{1}(q_2).....T_n{}^{n-1}(q_n) \quad \text{(II.7)}$$

Le modèle géométrique direct du robot peut être représenté par la relation:

$$\mathrm{X} = f(q) \quad \text{(II.8)}$$

q étant le vecteur des variables articulaires tel que :

$$q = [q_1, q_2, ..., q_n]^T \quad \text{(II.9)}$$

Les coordonnées opérationnelles sont définies par:

$$X = \lfloor P_x, P_y, P_z, \sim, \quad, ¥ \rfloor \quad \text{(II.10)}$$

II.1.1.3.Modèle géométrique inverse

Le modèle géométrique inverse (MGI) est l'ensemble des relations inverses à celles du modèle direct. Ce modèle permet d'exprimer les variables articulaires *q* du bras manipulateur en fonction des coordonnées opérationnelles *X* exigées pour l'exécution d'une tâche donnée.

$$q = f^{-1}(X) \quad \text{(II.11)}$$

Pour le calcul du modèle géométrique inverse, on distingue parmi les méthodes analytiques les deux méthodes suivantes :

– la méthode de Paul qui traite séparément chaque cas particulier et convient pour la plupart des robots industriels,

– la méthode de Peiper qui permet de résoudre le problème pour les robots à six degrés de liberté possédant trois articulations rotoïde d'axes concourants ou trois articulations prismatiques

– la méthode générale de Raghavan et Roth, donnant la solution générale des robots à six articulations à partir d'un polynôme de degré au plus égal à 16. Lorsqu'il n'est pas possible de trouver une forme explicite du modèle géométrique inverse, on peut calculer une solution particulière par des procédures numériques, Whitney, Ournier, Featherstone, Wolovich, Goldenberg, Sciavicco. On ne présente dans ce paragraphe que la méthode de Paul. [6]

II.1.1.3.1.Principe de la méthode de Paul

Considérons un robot manipulateur dont la matrice de transformation homogène a pour expression :

$$T_n^{\ 0} = T_1^{\ 0}(q_1)T_2^{\ 1}(q_2).....T_n^{\ n-1}(q_n) \qquad \text{(II.7)}$$

Soit U_0la situation désirée telle que :

$$U_0 = \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad \text{(II.12)}$$

On cherche à résoudre le système d'équations suivant :

$$U_0 = T_1^{\ 0}(q_1)T_2^{\ 1}(q_2).....T_n^{\ n-1}(q_n) \qquad \text{(II.13)}$$

Pour trouver les solutions de l'équation (II.14), Paul à proposé une méthode qui consiste à pré multiplier successivement les deux membres de l'équation (II.13) par les matrices T_{i-1}^{i} pour i variant de 1 à n-1, opérations qui permettent d'isoler et d'identifier l'une après l'autre les variables articulaires que l'on recherche.

$$T_0^1(q_1)U_0 = T_2^{\ 1}(q_2).....T_n^{\ n-1}(q_n) \qquad \text{(II.14)}$$

Le terme de droite est fonction des variables q_2 ……q_n . Le terme de gauche n'est fonction que des éléments de U_0et de la variable q_2 ;

– identification terme à terme des deux membres de l'équation (II.13). On se ramène à un système d'une ou de deux équations fonction de q_1 uniquement, dont la structure appartient à un type particulier parmi une dizaine de types possibles ; [6]

La succession des équations permettant le calcul de tous les $q_i$est la suivante :

$$U_i = T_n^{\ i} = T^{i}_{\ i-1}U_{i-1} \qquad \text{(II.15)}$$

II.2.1. Passage d'un MGD vers un MGI

MGD est la situation de l'OT du robot (sériel) en fonction de sa configuration. C'est un ensemble de relation en utilisant l'une des méthodes cités précédemment. L'inversion des

relations permet de déduire le MGI (système d'équations non linéaire) dont la résolution est la détermination des coordonnées articulaires.

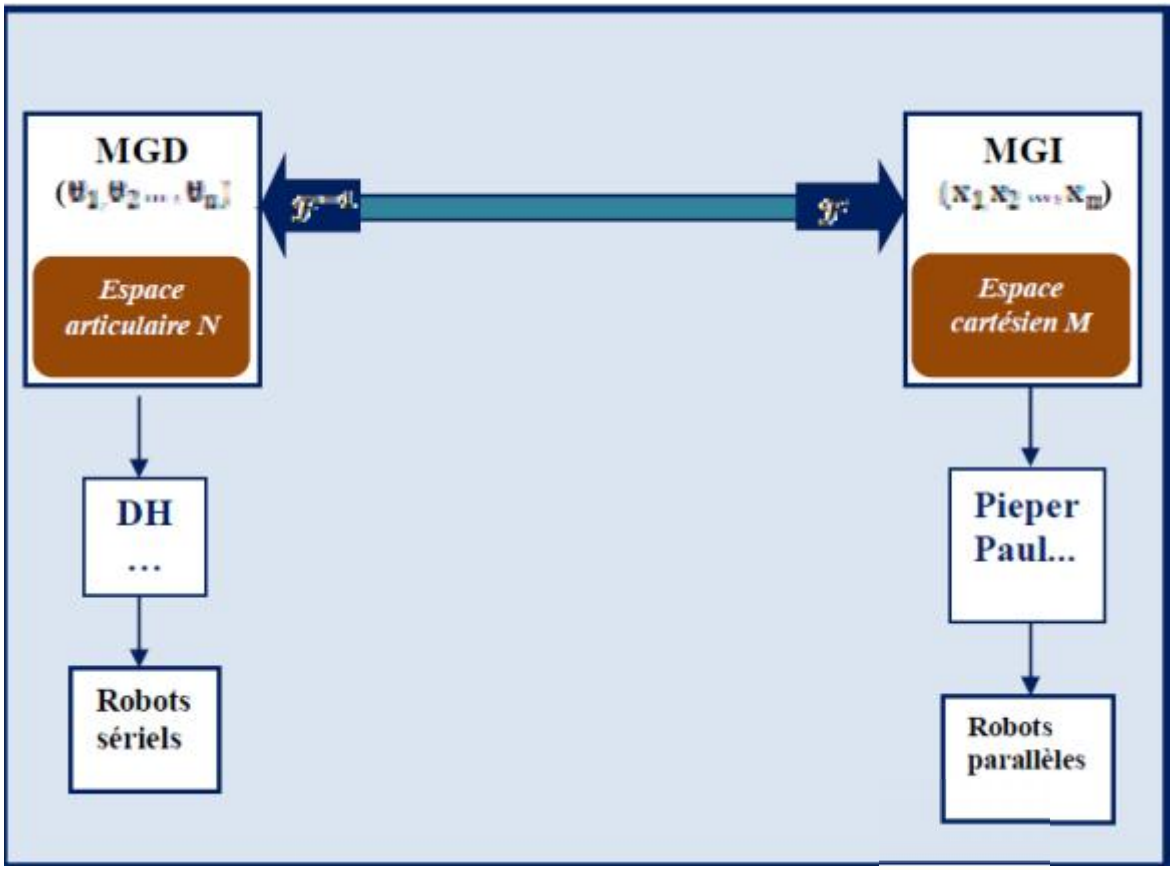


Figure (II.3) : Passage entre les model géométrique ($D \leftrightarrow I$)

II. 3 .Le modèle cinématique

II.3.1. Le modèle cinématique directe

Le modèle cinématique complète le modèle géométrique, en écrivant les relations entre les vitesses des variables articulaires et les vitesses de l'organe terminal. La propriété intéressante du modèle cinématique est sa linéarité par rapport aux vitesses. Ainsi il est plus simple à manipuler que le modèle géométrique et permet d'utiliser des propriétés qui découlent de la résolution des systèmes linéaires. [17]

Le modèle cinématique direct d'un robot -manipulateur décrit les vitesses des coordonnées opérationnelles en fonction des vitesses articulaires. Il est noté :

$$\dot{x} = J(q)\,\dot{q} \qquad \text{(II.16)}$$

Ou J(q) désigne la matrice jacobéenne de dimension (m×n) du mécanisme, égale à $\frac{\partial y}{\partial x}$ et fonction de la configuration articulaire q. La même matrice jacobienne intervient dans le calcul du modèle différentiel direct qui donne la variation élémentaire dX des coordonnées opérationnelles en fonction des variations élémentaires des coordonnées articulaires dq, soit :

$$dX = J(q)dq \qquad \text{(II.17)}$$

II.3.1.1. Intérêts de la matrice jacobienne

-elle est à la base du modèle différentiel inverse, permettant de calculer une solution locale des variables articulaires q connaissant les coordonnées opérationnelles X ;

– en statique, on utilise le jacobienne pour établir la relation liant les efforts exercés par l'organe terminal sur l'environnement aux forces et couples des actionneurs ;

– elle facilite le calcul des singularités et de la dimension de l'espace opérationnel accessible du robot. [6]

II.3.1.2 .Matrice jacobienne cinématique

Le calcul de la matrice jacobienne peut se faire en dérivant le MGD, $X = f(q)$à partir de la relation suivante :

$$J_{IJ} = \frac{\partial f_i(q)}{\partial q_j} \tag{II.18}$$

i=1,…. m ; j=1,……, n

Ou J_{IJ} l'élément (i, j) de la matrice Jacobienne J.

Le modèle cinématique direct peut être mis sous la forme :

$$\begin{bmatrix} V_n \\ W_n \end{bmatrix} = J_n \dot{q} \tag{II.19}$$

Cette équation peut être exprimée comme :

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ W_x \\ W_Y \\ w_z \end{bmatrix} = \begin{bmatrix} J_{I1} & J_{i2} & . & . & . & J_{in} \\ J_{A1} & J_{A2} & . & . & . & J_{An} \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ . \\ . \\ \dot{q}_n \end{bmatrix} \tag{II.20}$$

II.3.2. Modèle cinématique inverse

Afin de suivre le contour à une vitesse constante, ou toute vitesse prescrite, on doit connaitre la relation qui existe entre la vitesse de l'outil est les vitesses des différentes articulations (MCD).

La détermination des vitesses articulaires à partir de la vitesse de l'organe terminal est conceptuellement simple étant donné que la relation de vitesse est linéaire. Ainsi les vitesses articulaires sont trouvées via le Jacobienne Inverse J^{-1} .comme suit

$$\dot{q} = J^{-1} \dot{X} \quad \text{(II.21)}$$

II.4 Modèle dynamique

II.4.1 Modèle dynamique direct

Le modèle dynamique est la relation entre les efforts appliqués aux actionneurs et les positions, les vitesses et les accélérations articulaires. Les expressions des modèles dynamiques sont utilisées pour des applications telles que la simulation, le dimensionnement des actionneurs, l'identification des paramètres inertiels ou la commande. [17]

Le modèle dynamique par application de formalisme de Lagrange est donné par l'équation suivante :

$$f(q, \dot{q}, \ddot{q}, \Gamma, F_D) = 0 \quad \text{(II.22)}$$

-q le vecteur de coordonnées articulaires ;

-$\dot{q}$vecteur des vitesses articulaires ;

-$\ddot{q}$vecteur des accélérations articulaires ;

-$F_D$le vecteur des efforts appliqués sur le système.

II.4.1.1 Formalisme de Lagrange

Le formalisme de Lagrange décrit les équations du mouvement, lorsque l'effort extérieur sur l'organe terminal est supposé nul, par l'équation suivante :

$$\Gamma_i = \frac{d}{dt}\frac{\partial l}{\partial \dot{q}_i} - \frac{\partial l}{\partial q} \quad \text{i=1,…. n} \quad \text{(II.23)}$$

Avec :

- Γ : lagrangien du système égal à E – U ;
- E : énergie cinétique totale du système ;
-U : énergie potentielle totale du système.

L'énergie cinétique du système est une fonction quadratique des vitesses articulaires :

$$E = \frac{1}{2} \dot{q}^T A \dot{q} \quad \text{(II.24)}$$

II. 4. 2. Modèle dynamique inverse

L'établissement du modèle dynamique inverse d'un robot manipulateur basé sur le formalisme de Lagrange est particulièrement intéressant, ce modèle est donné par la relation suivante :

$$\frac{d}{d}\left(\frac{\partial T_{Si}}{\partial q_i}\right) - \left(\frac{\partial T_{Si}}{\partial q_i}\right) = G_i + T_i \quad \text{(II.25)}$$

-q_i: variable articulaire ;

- G_I : Force de gravité relative à θi ;

-T_i : Force articulaire ;

-T_{si}: Énergie cinétique du solide.

Le modèle dynamique pour l'intégralité du système (n solides) est exprimé par la relation suivante :

$$\sum_{i=1}^{n} \frac{d}{d}\left(\frac{\partial T_{Si}}{\partial q_i}\right) - \left(\frac{\partial T_{Si}}{\partial q_i}\right) = G_i + T_i \qquad \text{(II.26)}$$

III.5.Génération de mouvement (trajectoire)

La tâche de déplacement d'un robot est spécifiée en définissant un chemin que le robot doit suivre. Un chemin est une séquence de points définis soit dans l'espace des tâches (opérationnel) (afin de situer l'organe terminal), soit dans l'espace des configurations (articulaire) du robot (afin d'indiquer les valeurs des paramètres de liaison). [16]

Ces points peuvent être :

- programmés par apprentissage,
- issus d'une base de données d'un système de CAO, …

Le problème de la génération de mouvement est de calculer les séquences souhaitées (consigne) de variables articulaires ou de variables liées à l'organe terminal qui assurent le passage du robot par le chemin désiré.

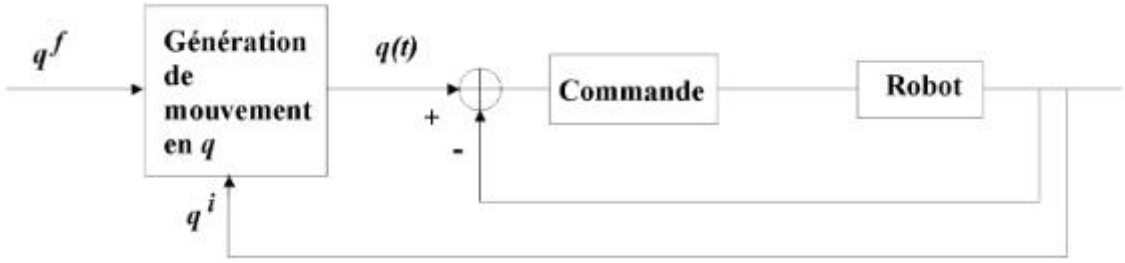
Les trajectoires d'un robot peuvent être classifiées comme suit :

- les mouvements entre 2 points avec des mouvements libres entre les points,
- les mouvements entre 2 points via une séquence de points intermédiaires désirés, spécifiés notamment pour éviter les obstacles ; la trajectoire est libre entre les points intermédiaires,
- les mouvements entre 2 points, la trajectoire étant contrainte entre les points,
- les mouvements entre 2 points via des points intermédiaires, la trajectoire étant contrainte entre les points intermédiaires.

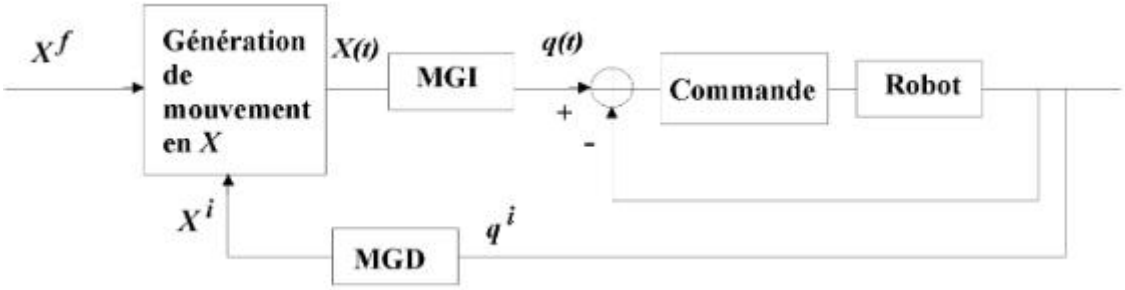
Dans les deux premiers cas, la génération de mouvement peut se faire directement dans l'espace des configurations : elle se traduit par une séquence de positions articulaires constituant les consignes des asservissements.

Dans les deux derniers cas, la trajectoire étant fixée à tout instant dans l'espace opérationnel, il est préférable de raisonner dans cet espace. La loi de commande engendrée doit ensuite être transformée en consignes articulaires par le changeur de coordonnées.

La génération de mouvement dans l'espace articulaire et génération de mouvement dans l'espace opérationnel – sont schématisées sur les figures suivantes. [16]



Génération de mouvement dans l'espace articulé



Génération de mouvement dans l'espace opérationnel

- La génération de mouvement dans l'espace articulaire présente plusieurs avantages :
  - le mouvement est minimal sur chaque articulation,
  - elle nécessite moins de calcul en ligne (au sens où il n'y a pas de changeur de coordonnées),
  - le mouvement n'est pas affecté par le passage sur les configurations singulières,
  - les contraintes de vitesse et de couples maximaux sont connues avec précision puisqu'elles correspondent aux limites physiques des actionneurs.

- La génération de mouvement dans l'espace opérationnel permet de contrôler la géométrie de la trajectoire (mouvement rectiligne par exemple). Par contre :
  - elle implique la transformation en coordonnées articulaires de chaque point de la trajectoire,
  - elle peut être mise en échec lorsque la trajectoire calculée passe par une position singulière,
  - elle peut être mise en échec chaque fois que les points de la trajectoire engendrée ne sont pas dans le volume accessible du robot ou chaque fois que la trajectoire impose une reconfiguration du mécanisme (changement d'aspect en cours de trajectoire). [16]

Conclusion

Pour modéliser un robot à n articulations, il faut représenter le comportement de ce robot sous la forme d'un modèle, une telle démarche s'appelle la modélisation, d'une manière générale, on recherche toujours le modèle le plus simple qui permet d'expliquer, de manière satisfaisante, le comportement du processus dans son domaine d'application; les modèles de transformation entre l'espace opérationnel et l'espace articulaire.

III. Introduction

Dans le processus continu, le robot est manipulé par une série de commandes et de fonctions stockées dans un fichier dit «programme robot ». La trajectoire du robot, et par conséquence de la torche, doit être programmée avant son exécution. Deux modes de programmation des trajectoires robot sont couramment utilisés : la programmation en ligne et la programmation hors-ligne. La méthode principale de programmation en ligne est la programmation par apprentissage. Il existe deux méthodes pour la programmation hors-ligne : la programmation par langage et la programmation graphique à partir d'un système de CFAO (Conception et Fabrication Assistée par Ordinateur). [18]

III.1 La programmation en ligne (PEL)

La programmation en ligne est toujours la méthode de programmation la plus utilisée dans les applications industrielles. Il s'agit de la programmation par apprentissage. Son principe est de montrer manuellement au robot ce qu'il doit faire par l'opérateur. Cette technique est très pratique car il permet de réaliser la programmation rapidement et facilement. En plus, elle peut être maîtrisée par un personnel non qualifié. On estime à 80% le nombre de robots qui sont programmés de cette façon dans les applications industrielles. [19]

Figure (III.1) : La programmation en ligne

Une trajectoire effectuée par l'organe terminal du robot est constituée d'un certain nombre de points précis, placés à des endroits judicieusement choisis par l'opérateur.. Pour ce faire, l'opérateur dispose d'un pupitre d'apprentissage comportant les commandes manuelles et permettant de déplacer le robot axe par axe ou amener le CDO. Lorsque la position souhaitée est atteinte, elle est mémorisée dans une série d'instructions de mouvement (Figure III). Il est facile de réaliser et de programmer en concordance avec les positions réelles des équipements et des pièces. [18]

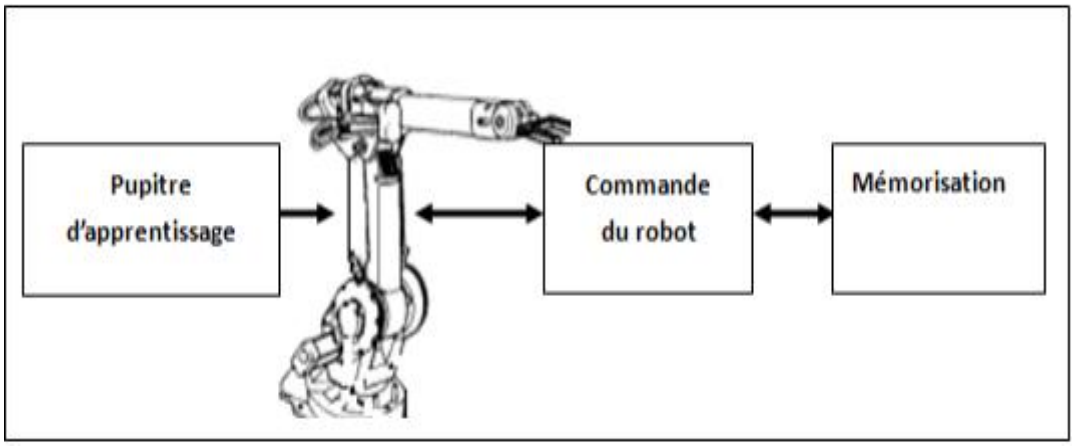


Figure (III.2) : Programmation par apprentissage

La programmation en ligne est aujourd'hui le mode de programmation le plus utilisé

industriellement, il existe deux méthodes de programmation en ligne: l'apprentissage direct et l'apprentissage indirect point à point.

III.1.1. L'apprentissage direct

Le principe de cette méthode de programmation consiste à mémoriser, lors de l'apprentissage, l'ensemble des configurations articulaires du robot pendant l'exécution de la tache ou pendant une simulation de celle-ci. L'opérateur manipule directement le robot à l'aide d'un pupitre d'apprentissage pour lui "apprendre" la tache à accomplir. L'acquisition de la trajectoire est continue. Ce type de programmation est simple et demande peu de formation pour le programmeur. Cette méthode est plutôt utilisée pour des applications qui exigent des mouvements de précision limitée, dans la projection sur les pièces simples ou le domaine de la peinture. [20]

III.1.2. L'apprentissage indirect point par point

Le principe utilisé par cette méthode est également basé sur une démonstration matérielle de la tache à réaliser. L'opérateur déplace le robot en mode manuel, utilisant pour cela le pupitre de commande à touche. Cependant, la configuration du robot n'est enregistrée qu'en certains points caractéristiques de la trajectoire. La liaison entre ces différents points est spécifiée par les caractéristiques de la trajectoire entre ces points (linéaire, articulaire voire circulaire) et par la définition d'une vitesse de déplacement spécifiée par l'opérateur. [20]

III.2 Programmation hors-ligne

Pour éviter les arrêts de production, on choisira la programmation hors ligne sur un logiciel PC où les points de la trajectoire sont calculés ou sont issus d'une simulation.

Dans ce cas, les logiciels proposés permettent :

• de simuler graphiquement sur PC le programme robot (vue du robot en mouvement à l'écran, vérification du temps de cycle).

• de créer ou d'éditer les programmes.

• de transférer des programmes robot du PC à l'armoire de commande. [21]

La programmation hors-ligne correspond à la génération d'un programme robot à exécuter sur site à partir des données CAO (Conception Assistée par Ordinateur) générées avec un logiciel sur ordinateur notamment le logiciel virtuel Robot Studio équipé d'une bibliothèque du robot. Ce type de programmation est en train de devenir une tendance pour des nombreuses applications robotiques. [19]

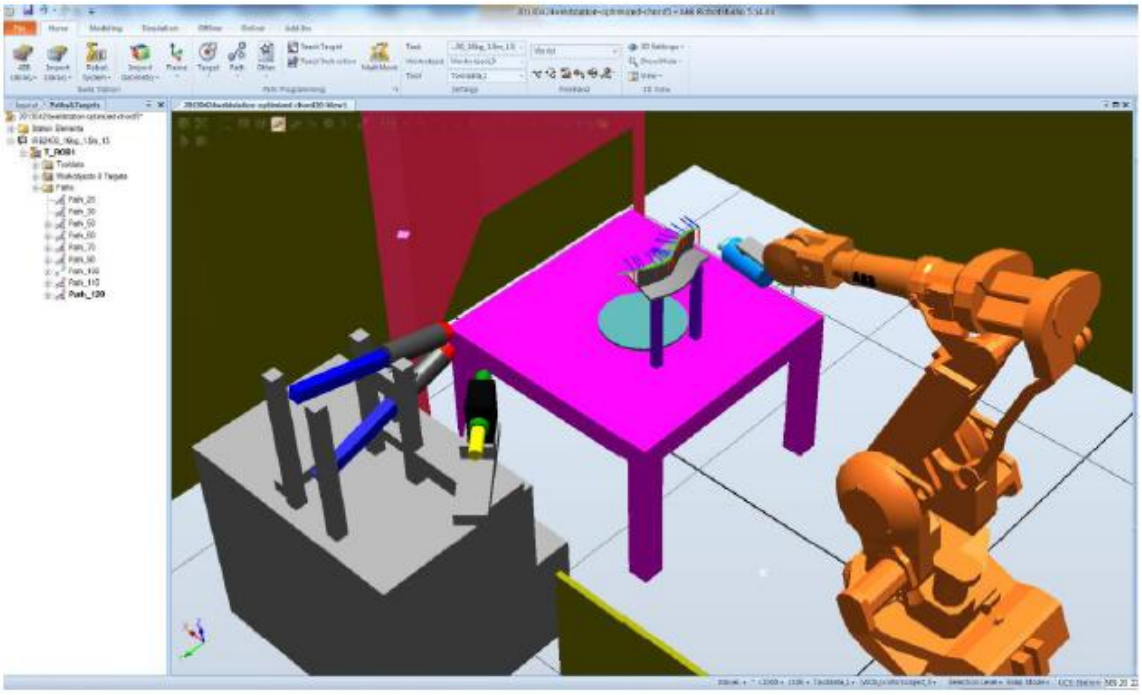


Figure (III.3) : Programmation hors ligne

Les caractéristiques de la programmation hors-ligne doivent permettre :

- D'assurer la sécurité humaine: l'environnement virtuel est une réplique exacte de la cellule du robot, donc le programme du robot peut être conçu dans le bureau de l'opérateur, ce qui permet de l'éloigner de lieux ou peut subsister un quelconque danger.
- De diminuer le temps de cycle: le programme du robot peut être préparé à l'avance, ou être conçu malgré l'indisponibilité du robot (en production par exemple), ce qui réduit le temps de préparation.
- De diminuer le risque de collision entre l'outil de travail, le robot et la pièce dans la cellule : le programme du robot peut être simulé dans le logiciel de programmation hors-ligne pour visualiser la détection de la collision avant l'exécution dans la cellule sur site.
- De faciliter la programmation du robot : grâce à la duplication de la cellule de robot, le programme préparé par une interface graphique du logiciel facilite le processus de programmation, et une même tâche peut être affectée à plusieurs robots. [19]

III.2.1.La programmation par langage robotique

Les langages de programmation des robots sont proches des langages informatique classique. Ils proposent des instructions particulières pour commander les mouvements des robots. Ils sont performants pour définir des positions par calculs ou pour implanter des instructions conditionnelles ou des boucles. [20]

Le principal inconvénient de ces langages est qu'ils restent généralement spécifiques à chaque marque de robot; donc il n'existe pas un langage de programmation universel chaque fabricant de robots a son propre langage et son environnement de développement Par ex. staubli (langage *VAL3*), KUKA (langage *KRL*), ABB (langage *RAPID*). Dans notre étude nous allons utiliser le langage RAPID. [22]

Donc la programmation par apprentissage qui fait référence à un langage «gestuel», la programmation par langage est basée sur un langage «textuel». [18]

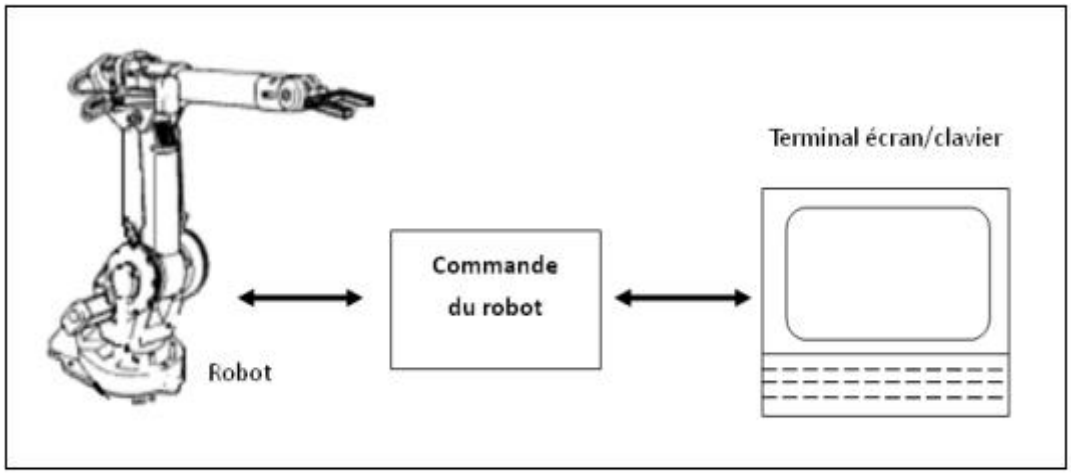


Figure (III.4) : Programmation par langage

III.2.2.La programmation graphique

Basée sur l'utilisation des langages de programmation et des simulateurs graphiques, les systèmes de programmation graphique tendent à réduire le temps d'occupation du robot pour la programmation des tâches. Il est intéressant en effet de ne pas arrêter le travail en cours du robot. Le principe est donc de préparer la tâche sans utiliser le robot lui-même, puis de charger le programme ainsi réalisé, dans la mémoire du système de commande du robot (Figure III). Pour obtenir une certaine visibilité de l'exécution future de la tâche, on peut faire appel à un système de CAO robotique qui permet de simuler les mouvements du robot, d'éviter les collisions et donc de prévoir au mieux le bon déroulement réel de la future tâche en vue tridimensionnelle. Le système CAO est de plus un système de programmation unique pouvant être utilisé pour plusieurs robots : cela permet éventuellement de réduire le coût du travail. [18]

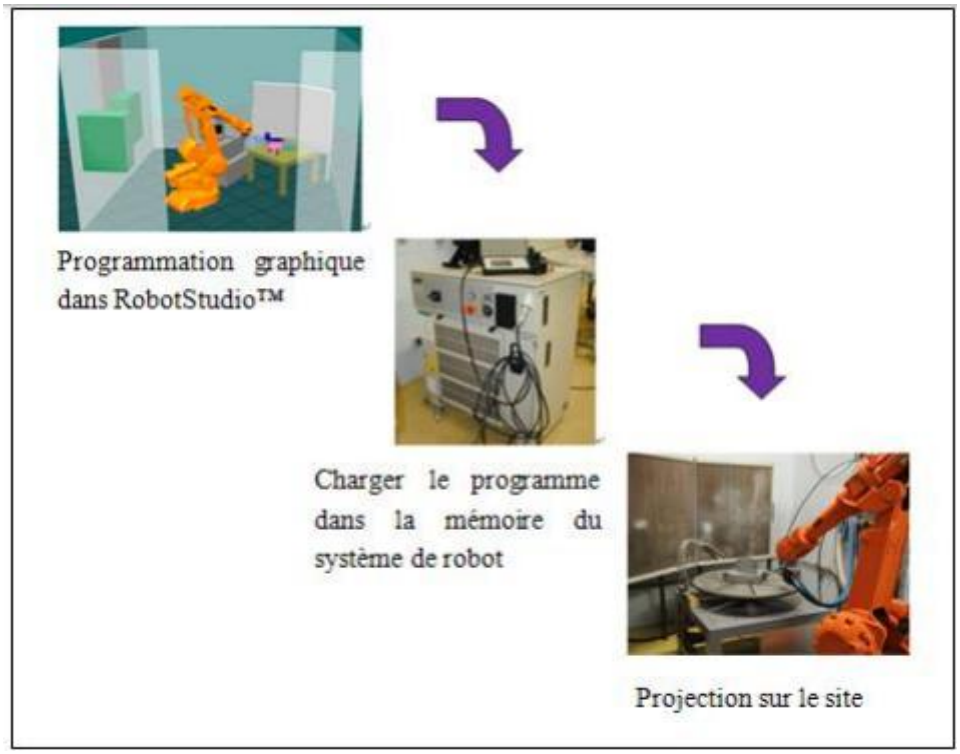


Figure (III.5) : Programmation graphique.

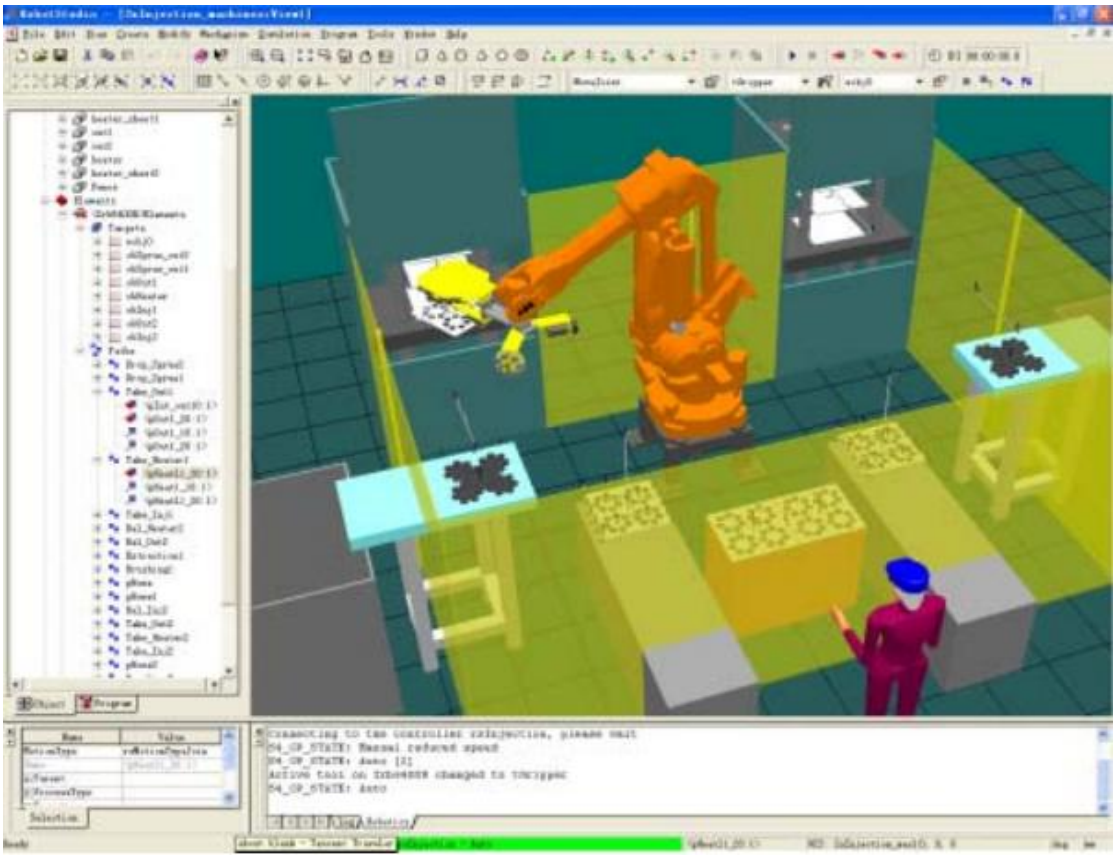


Figure (III.6) : Programmation graphique.

Au niveau de leur mise en œuvre, les principaux avantages de ces systèmes de programmation sont les suivants :

- La programmation des robots peut se faire sur des sites CAO déjà largement implantés pour d'autres types d'études (notamment en construction automobile, chantiers navals, aéronautique…).
- Grâce aux possibilités de travail en mode interactif, la formation nécessaire pour utiliser les sites de CAO est simple et de relativement courte durée. On pourra donc utiliser pleinement l'expérience et les compétences des techniciens d'atelier ayant une bonne connaissance des robots et de leurs modes de travail. [18]

Dans le cadre de cette étude, le logiciel de CFAO robotique dit Robot Studio a été utilisé en raison de sa bonne compatibilité entre les robots virtuels et les robots réels.

III.3. la CAO

III.3.1.Définition de la CAO

Nous pouvons définir la Conception Assistée par Ordinateur (CAO) par l'ensemble des outils logiciels et des techniques informatiques qui permettent d'assister les concepteurs dans la conception et la mise au point d'un produit.

A titre d'exemple de logiciels de CAO on peut citer Auto CAD, SolidWorks, Catia, SolidConcept, SolidEdge, ProEngineer, ACIS….

Un logiciel de CAO se compose généralement de quatre parties majeures qui peuvent être organisées comme suit : [23]

III.3.1.1. Le modeleur géométrique

Il représente "la planche à dessin". Nous trouvons dans cette partie les composants géométriques essentiels: points, droites, cercles, ellipses, plans, sphères, cylindres, cônes, courbes de Bézier ou B-Splines, surfaces NURBS, surfaces de révolution, surfaces de balayage, etc. Il intègre également les composants topologiques: sommets, faces, arêtes, orientations, coïncidences, adjacences, intersections, soustractions, unions, etc.… [23]

III.3.1.2. L'outil de visualisation. [23]

III.3.1.3. Un certain nombre d'applications

Nous retrouvons le calcul des grandeurs géométriques (distances, inerties, volumes, masses, etc.), les fonctions métiers: assemblage de pièces, production de plans, simulation d'usinage, moulage, fraisage, etc. [23]

III.3.1.4. Un contrôleur:

Il gère et manipule les intersections entre les trois outils cités précédemment. [23]

III.3.2. Logiciels CAO en robotique [24]

Dans le cas général, les systèmes CAO en Robotique ont deux objectifs majeurs.

- Aide à la conception et à la simulation ainsi que l'optimisation des cellules robotisées, au moyen des outils dont ils disposent pour la modélisation des robots et de leurs environnements. Ces systèmes permettent de générer des trajectoires, d'analyser les mouvements, de détecter les collisions et d'évaluer des temps de cycle.
- Les systèmes CAO en robotique permettent de créer hors ligne des programmes téléchargeables et exécutables par les robots manipulateurs conçus.
- Ces systèmes de CAO ont été développés principalement pour simuler des robots industriels ayant des structures sérielles. La simulation d'un robot existant dans une cellule de production complète permet aux concepteurs de réduire de manière très considérable le temps de conception d'un produit et d'améliorer sa qualité. De nos jours, le comportement des mouvements des systèmes robotisés simulés par un de ces logiciels est conforme à la réalité avec une grande précision.

III.4. Logiciel de programmation hors-ligne RobotStudio

III.4.1. Définition de RobotStudio

Robot Studio est un logiciel de programmation hors-ligne fourni par l'entreprise ABB. Celui-ci est spécialement conçu pour la réalisation des tâches robotiques telles que la programmation hors-ligne, la simulation de la trajectoire et la détection de collision.

RobotStudio™ est basé sur la technologie de Virtual Controller d'ABB, qui exécute le même logiciel intégré dans les robots réels en production. Cette technique permet de construire graphiquement un site robotisé virtuel comme le site réel. Les mouvements du robot sont donc spécifiés à l'aide de ce logiciel par l'utilisateur, et ensuite les codes exécutables sont formés automatiquement et préparés à être téléchargé vers le vrai robot. [19]

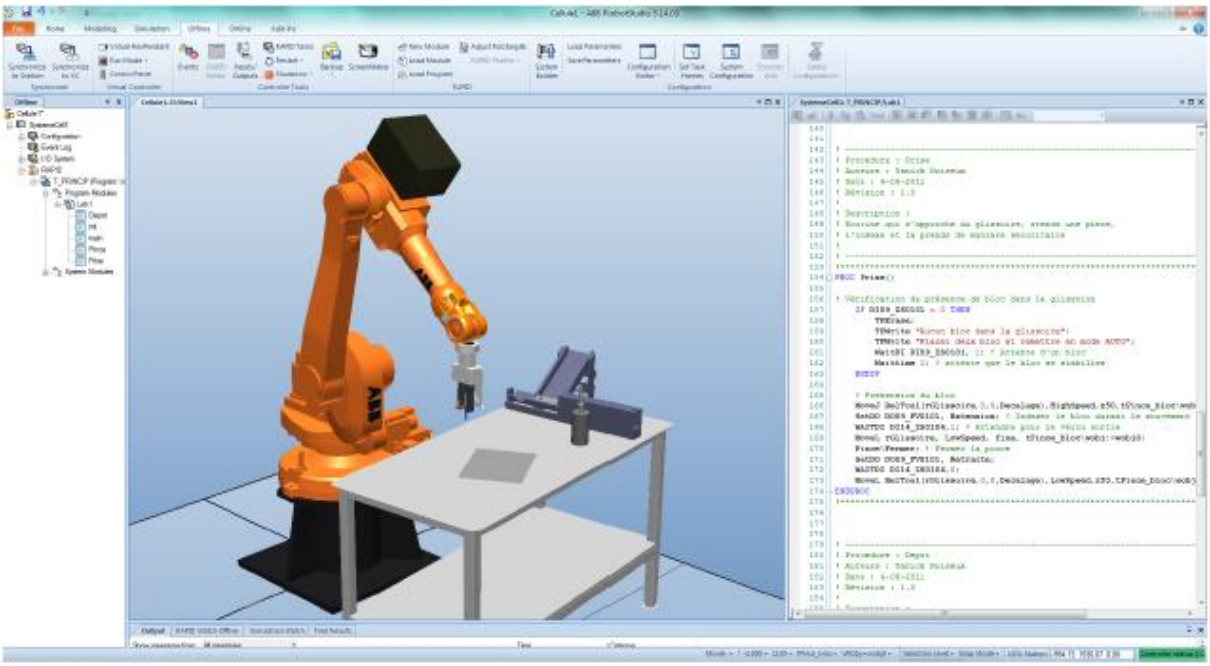


Figure (III.7) : Programmation hors ligne dans Robot Studio avec une cellule virtuelle.

III.4.2. Les avantages du RobotStudio

- Diminution des risques de programmation : la génération de la trajectoire, la simulation des paramètres cinématiques et la détection de collisions dans l'environnement graphique assurent la préservation de la pièce et la sécurité de l'opérateur.
- Augmentation du rendement global : le programme se construit à partir d'un ordinateur de bureau et peut être préparé en avance bien que le robot est encore en production.
- Haute compatibilité des donnés géométriques : les modèles géométriques couramment utilisés tels que IGES, STEP, VRML, VDAFS, ACIS et CATIA peuvent être importés directement dans Robot Studio comme les composants numériques pour la modélisation.
- Bibliothèque de système du robot : tous les types de modèles robots des produits ABB sont disponibles. [19]

III.4.3. Site robotisé virtuel et génération de la trajectoire du robot

Modélisation du site robotisé consiste à générer de façon virtuelle une station qui inclue tous les outillages essentiels pour la génération de la trajectoire du robot. Une station robot contient notamment un ou plusieurs robots, les outils, les positionneurs et d'autres équipements nécessaires. Le procédé de modélisation d'équipements du robot doit prendre en compte la précision de la trajectoire, car il peut introduire directement des erreurs non négligeables lors de la synchronisation du programme sur site. La création d'une station à partir d'un système existant est fournie par Robot Studio™. Cela permet de générer une nouvelle station virtuelle qui comporte les modèles du robot et les contrôleurs du robot correspondants. [19]

III.5. La programmation

On peut programmer le robot à partir du FlexPendant ou de Robot Studio. Sur le FlexPendant, les instructions et les arguments sont pris dans des listes d'alternatives adaptées. Dans Robot Studio, les programmes sont développés en langage texte.

III.5.1.Le langage RAPID

Le langage de programmation des robots ABB s'appelle RAPID. Un programme en RAPID est un ou plusieurs fichiers de type texte (ASCII), non compiles. Le contrôleur du robot interprète le code, tout en validant la syntaxe du programme en entier. Un programme contenant des erreurs de syntaxe peut être chargé dans le contrôleur, mais ne peut pas être exécuté.

Il existe deux types de fichiers. Le premier type est le plus important : les modules (.mod) qui contiennent le programme. Cela ne fait aucune différence pour le contrôleur du robot si le programme est écrit dans un seul ou dans plusieurs modules. L'utilisation de plusieurs modules se justifie uniquement dans la plus grande facilité de saisi et de réutilisation des modules par le programmeur. Par contre, il ne peut y avoir qu'un seul programme actif dans le contrôleur du robot ; c'est-`a-dire qu'un seul des modules peut contenir une procédure

appelée « main ». Lors du chargement dans le contrôleur des différents modules, le contrôleur assumera automatiquement l'ensemble de modules comme un programme. Les modules peuvent ˆêtre chargés s´séparément, un par un, ou en utilisant le deuxième type de fichier : le programme (.pgf), qui est simplement un fichier de type XML qui spécifie la liste des modules à charger. Le contrôleur amorce le programme en partant par la procédure « main ». Celle-ci doit être au début d'un module, juste après la d´déclaration des variables globales à la tâche. [5]

III.5.2. Syntaxe

Le contrôleur n'est pas sensible à la case, ce qui signifie par exemple que dans un module, les variables « p1 » et « P1 » seront considérées les mêmes. Le langage RAPID n'est pas à structure imposée, non plus. Il est donc possible d'utiliser plusieurs espaces, tabulations et retours de chariot, afin de faire une belle mise en page. Enfin, c'est le point d'exclamation qui est utilisé pour les commentaires.

La plupart des instructions exigent un point-virgule à fin. Le programmeur a donc toute la lassitude nécessaire afin de rendre son programme lisible, bien documenté, et facile à déverminer et à corriger. [5]

III.5.3.Type de variables de base

Le langage RAPID offre la possibilité au programmeur de se créer des registres mémoires afin de stocker des informations pertinentes au déroulement d'un projet. Son approche est similaire à un langage de programmation informatique. La déclaration est structurée et les registres que l'on désire accessibles de partout doivent être déclarés dans le début du module avant toutes instructions.

Le registre mémoire ce définit en trois types :

- CONST : Une constante représente une valeur statique lors de l'exécution du programme. Elle peut seulement recevoir une nouvelle valeur manuellement que ce soit lors d'une programmation en ligne ou hors-ligne.
- VAR : Une variable représente une valeur dynamique que le programme peut modifier lors de son exécution. Elle peut recevoir une nouvelle valeur en tout temps. Celle-ci n'est réinitialisée que lorsqu'on demande au contrôleur de réinitialiser un programme (opération appelée « PP to Main »).
- PERS : Une persistante peut être d´écrite comme étant une variable « persistante » dans le temps. Lorsqu'on modifie sa valeur, sa définition est automatiquement mise à jour par le contrôleur afin de retenir toujours la dernière valeur.

III.5.4. Les enregistrements

Un enregistrement est composé d'un mélange de données atomiques et/ou d'enregistrements. Chaque composante d'un enregistrement a un nom et peut être adressée en utilisant le nom de l'enregistrement suivi par le séparateur «. » suivi par le nom de la composante, etc. Les enregistrements les plus importants sont les poses (pose), les référentiels

(wobjdata), la pose de l'effecteur du robot combinée à la configuration du robot (robtarget) et les définitions d'outil (tooldata).

Les enregistrements les plus importants sont les poses (pose), les référentiels (wobjdata), la pose de l'effecteur du robot combinée à la configuration du robot (robtarget) et les définitions d'outil (tooldata). [5]

III.5.4.1. La pose

Une pose est une représentation mathématique d'un référentiel par rapport à un autre. Dans RAPID, un enregistrement de type pose est composé d'un enregistrement de type position (pos) et d'un enregistrement de type orientation (orient), par exemple [[0, 0, 0], [1, 0, 0, 0]].

III.5.4.2. La wobjdata

L'enregistrement wobjdata (« work object data ») permet de définir un référentiel de travail. Il est très rare, voire quasi impossible, d'avoir un référentiel de robot parfaitement aligné avec son milieu de travail. Il est donc souvent nécessaire de se définir un référentiel par rapport à un objet et/ou une surface de travail. De plus, les cellules ´étant parfois complexes, il n'est pas rare d'avoir plus d'un lieu de travail. Dans le cas d'un robot ABB, les enregistrements de type wobjdata doivent toujours être déclarés globaux et persistantes (PERS). L'enregistrement wobjdata est une structure complexe composée des éléments suivants :

- robhold (bool) : Sert à définir si le robot tient l'outil, ou si l'outil est fixe.
- ufprog (bool) : Sert à définir si le référentiel se déplace dans l'espace ou s'il est fixe.
- uframe (pose) : Sert à définir la pose du référentiel du lieu de travail (< user frame >) par rapport au référentiel de l'atelier wobj0 (figure III). La valeur de cette composante provient le plus souvent d'un enseignement manuel à l'aide du FlexPendant ou de la fonction DefFrame.
- oframe (pose) : Sert à définir la pose de l'objet sur lequel on va travailler (< object frame >) par rapport au uframe (figure III). Souvent, cette composante est laissée à sa valeur par sans effet, soit [[0; 0; 0]; [1; 0; 0; 0]] (c'est-à-dire que le référentiel oframe coïncide avec le référentiel uframe).

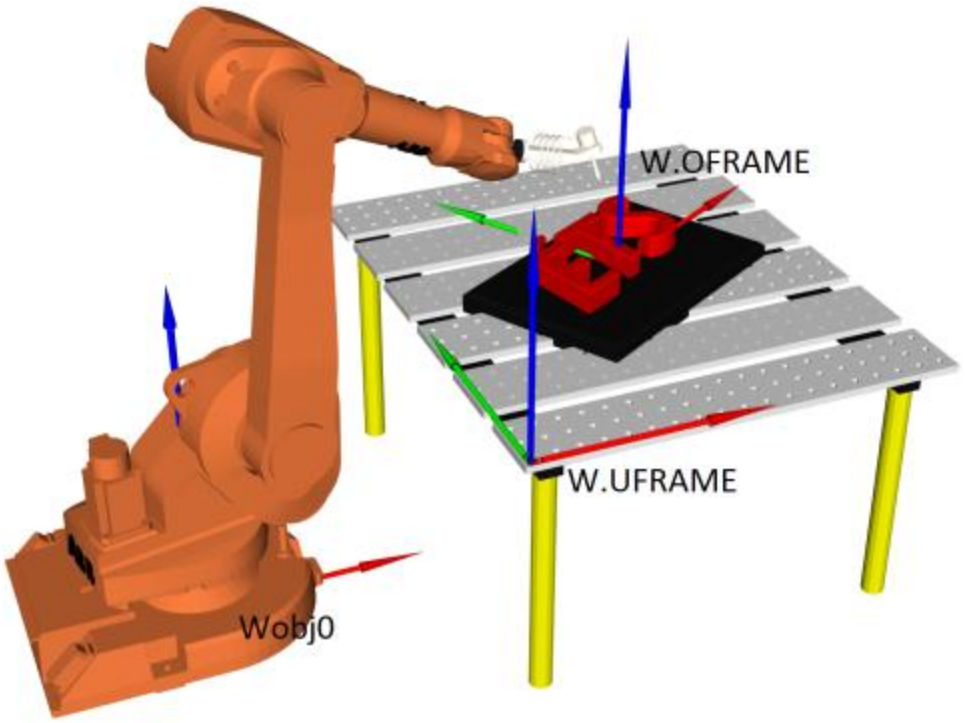


Figure (III.8) : Référentiels faisant partie d'un enregistrement de type workoject,

III.5.4.3.Le robtarget

L'enregistrement de type robtarget est utilisé pour représenter une pose de l'outil du robot par rapport à un référentiel (wobjdata) non spécifie, ainsi que la configuration du robot et les positions des moteurs supplémentaires (tel qu'un guide linéaire ou une table pivotante).

- Trans (pos) : Contient les coordonnées x, y et z (en mm) d'un référentiel d'outil non spécifié par rapport à un référentiel de travail (wobjdata) non spécifié.
- rot (orient) : Contient le quaternion exprimant l'orientation du même référentiel d'outil par rapport au même référentiel de travail (wobjdata).
- robconf (confdata) : Permet de définir la valeur du cadran des articulations 1, 4 et 6, ainsi que le numéro de configuration (de 0 à 7, dans le cas des robots sériels à 6 ddl).
- extax (extjoint) : Permets d'interagir avec les articulations externes du robot.

III.5.4.4. La tooldata

L'enregistrement de type tooldata permet de définir un référentiel sur un outil ainsi que les caractéristiques physiques de l'outil.

L'outil de chaque robot est fixé à la bride du robot. En plus de contenir un référentiel physique, l'enregistrement va également contenir le poids, le centre de gravité et les moments d'inertie de l'outil. Il est important de définir ceux-ci si on veut optimiser la précision et la vitesse du robot. L'enregistrement de type tooldata est une structure composée principalement des trois niveaux suivants :

- robhold (bool) : Sert à définir si le robot tient l'outil.
- tframe (pose) : Permet de définir la pose du référentiel de l'outil par rapport au référentiel de la bride.
- tload (loaddata) : Permet de définir la charge physique de l'outil et ainsi permettre au robot de connaitre les forces physiques générées par l'outil lors de son d´emplacement et ainsi optimiser sa vitesse et sa trajectoire.

III.5.5. Les commandes de déplacement

La commande de déplacement est de déplacer l'outil du robot d'une manière spécifique. On doit toujours spécifier un robtarget comme cible. Il existe trois types de déplacements dont la cible est un robtarget.

III.5.5.1. Types de déplacement

- **Déplacement linéaire**

Dans un déplacement linéaire, l'origine du référentiel de l'outil spécifie suit une trajectoire linéaire. Ce type de déplacement est très contraignant, car il oblige souvent le robot à réaliser des mouvements complexes et parfois brusques.

Un déplacement linéaire peut également être limite par ce qu'on appelle des singularités. Quand un programmeur rencontre une singularité dans un mouvement linéaire, il est fréquent que la seule solution soit un changement physique de la cellule de travail afin que le robot ne repasse plus sur cette singularité. Dans le cas d'un robot ABB, la somme des mouvements pour chaque paire d'articulations 1 et 4, 1 et 6, 3 et 4 ou 3 et 6, ne doit pas dépasser 180° .

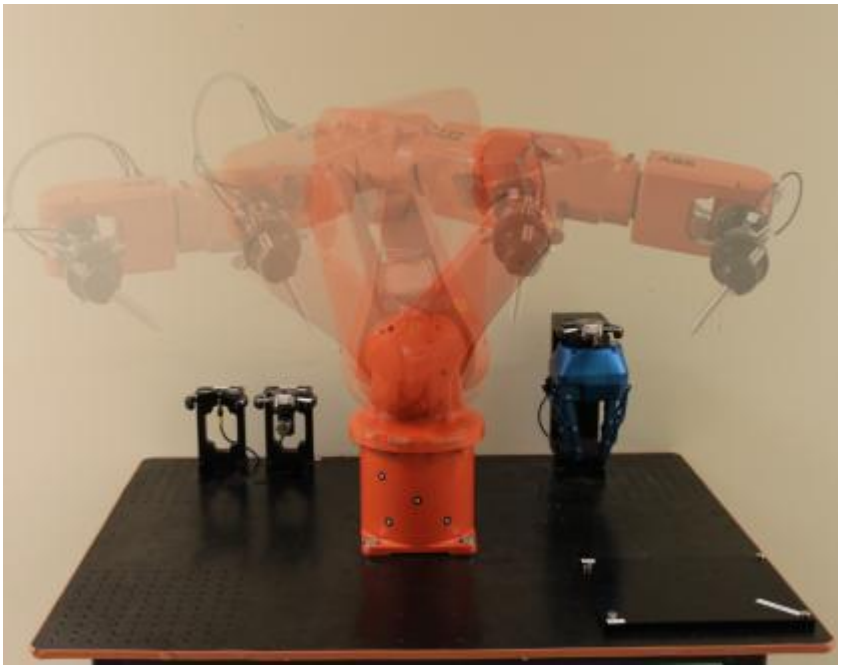


Figure (III.9) : Déplacement linéaire de l'outil (commande MoveL).

Les paramètres de base d'une commande de linéaire sont comme suit :
1. MoveL : Commande pour un déplacement linéaire de l'endroit o'u il se trouve vers le robtarget spécifié.
2. robtarget : Valeur venant d'une variable, fonction ou autre qui indique la destination de l'outil.
3. speeddata : Vitesse de croisière maximale que l'outil peut d'atteindre durant le déplacement.
4. zonedata : Précision d'arrêt ou d'approche de l'origine du référentiel de l'outil sur un robtarget `à atteindre.
5. tooldata : Le référentiel de l'outil et les caractéristiques physiques de l'outil.
6. wobjdata : Paramètre optionnel qui défini par rapport `a quel référentiel de travail on veut positionner le référentiel de l'outil 'a la pose définie dans le robtarget. Si ce paramètre est omis, le référentiel par défaut est celui de l'atelier (appelée wobj0).

- **Déplacement articulaire**

Dans un déplacement articulaire, la trajectoire suivie par l'effecteur du robot est difficilement prévisible .Ce type de déplacement offre une grande liberté de mouvement sans risque de singularité durant le déplacement. Le robot réalise un déplacement articulaire en définissant les valeurs des articulations à la configuration où il se trouve, et ensuite en calculant les valeurs de celles-ci à la destination finale. Une fois le déplacement nécessaire de chaque articulation connu, le robot démarre toutes les articulations en même temps, a différentes vitesses. [5]

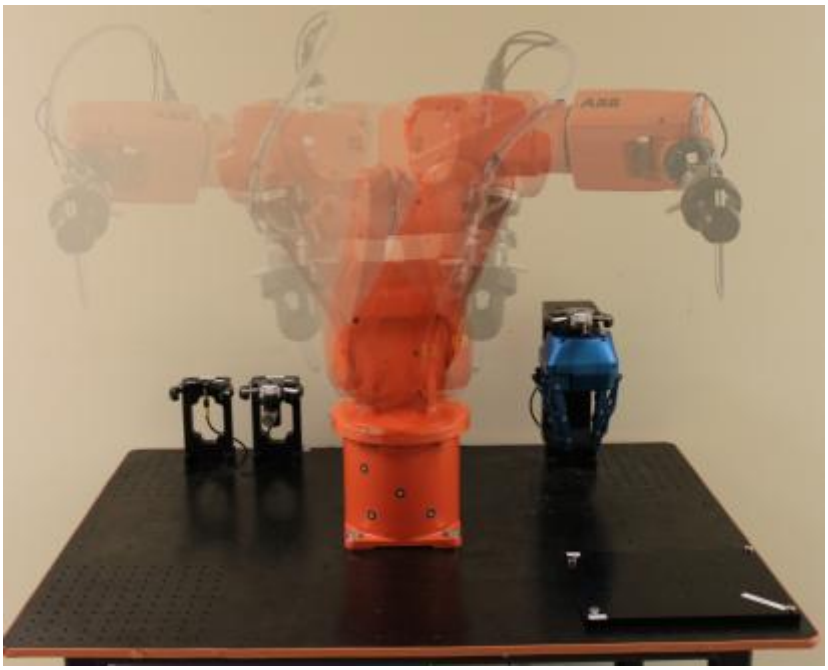


Figure (III.10) : Déplacement articulaire de l'outil (commande MoveJ).

III.5.6. Vitesse et précision du déplacement

III.5.6.1. Vitesse

La vitesse est un enregistrement de type speeddata qui s'évalue toujours au niveau du référentiel de l'outil. Il existe dans le contrôleur plusieurs constantes de type speeddata déjà définies (v1, v10, v20, v50, v100, etc.).[5]

III.5.6.2. L'interpolation

Est un enregistrement de type zone data et définit la tolérance en position à respecter par rapport au robtraget spécifie, dans un déplacement, Cette fonctionnalité a pour but d'assurer la fluidité des parcours et d'optimisation les temps de cycles.

L'enregistrement de type zone data est composé de plusieurs valeurs qui définissent le comportement de l'outil lors de l'arrivée dans la zone d'interpolation. Les trois premiers paramètres nous sont utiles au laboratoire, et les quatre derniers peuvent contenir une valeur arbitraire.[5]

1. **finep :** Variable booléenne définissant si le robot doit s'arrêter ;
2. **pzone-tcp :** rayon en mm de la sphère ou l'origine du référentiel de l'outil peut commencer l'interpolation vers le prochain trajet ;
3. **pzone-ori** : rayon en mm de la sphère ou l'origine du référentiel doit se trouver pour que l'orientation de l'outil puisse commencer l'interpolation vers le prochain trajet. Ce rayon doit être égal ou plus grand que le rayon précédant. [5]

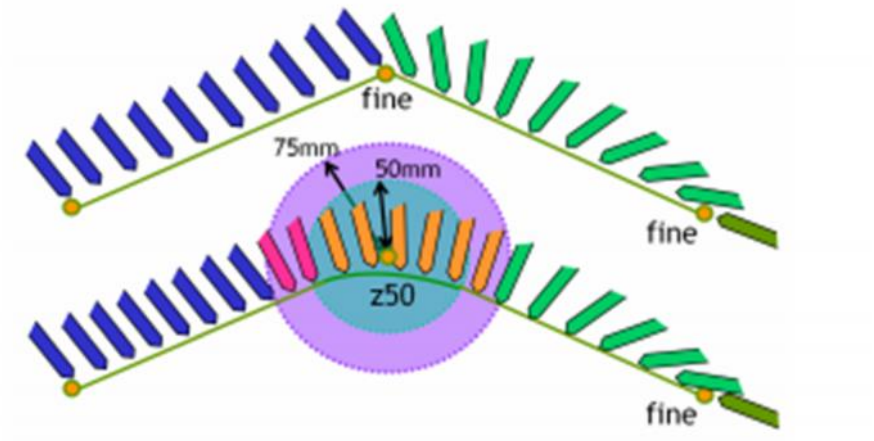


Figure (III.11): Interpolation entre deux déplacements linéaires.

Conclusion

Dans ce chapitre on a démontré la programmation graphique et les commandes de déplacement qui ont une importance dans la programmation et la planification des taches.

IV. Introduction

Les systèmes à convoyeurs sont indispensables dans l'industrie (manutention, stockage, déstockage, palettisation...etc.), ils nécessitent une bonne et fiable programmation pour assurer leur fonctionnement. Une des applications les plus courantes de la robotique industrielle est **le soudage**.

Et certaines entreprises proposent des robots de **palettisation** dédiés. Ces palettiseurs robustes sont conçus et fabriqués pour fournir un service ininterrompu et des temps de cycle plus rapides. Dotés d'un logiciel adaptable ne nécessitant aucune reprogrammation et capables d'assurer la manutention de charges particulièrement volumineuses, ces palettiseurs constituent la solution idéale pour un nombre illimité d'opérations de palettisation complexes, mixtes ou standard. Donc les prévisions pour ce type d'applications sont en croissance significative étant donné que les robots deviennent de plus en plus faciles à manipuler.

Cependant la programmation hors ligne est le meilleur moyen d'augmenter le retour sur investissement des systèmes robotisés. L'un des logiciels les plus utilisés pour la programmation des taches industrielles est le logiciel ABB de simulation et de programmation hors ligne RobotStudio permet la programmation à partir d'un ordinateur de bureau sans interrompre la production.

Dans ce chapitre on va faire la programmation hors ligne et graphique de deux robots industrielles de type IRB 1600 de la compagnie ABB, à l'aide du logiciel RobotStudio. Ces deux robots travaillent en synchronisation, le premier robot fait la tache de soudure d'un objet sur un convoyeur et l'autre fait la tache de palettisation.

IV.1.Programmation des taches

Dans cette partie nous allons créer et programmer à l'aide du logiciel RobotStudio, un système de deux convoyeurs, un robot soudeur et un robot palettiseur, les étapes a suivre sont les suivantes:

- Tout d'abord on commence par créer une Cellule vide.

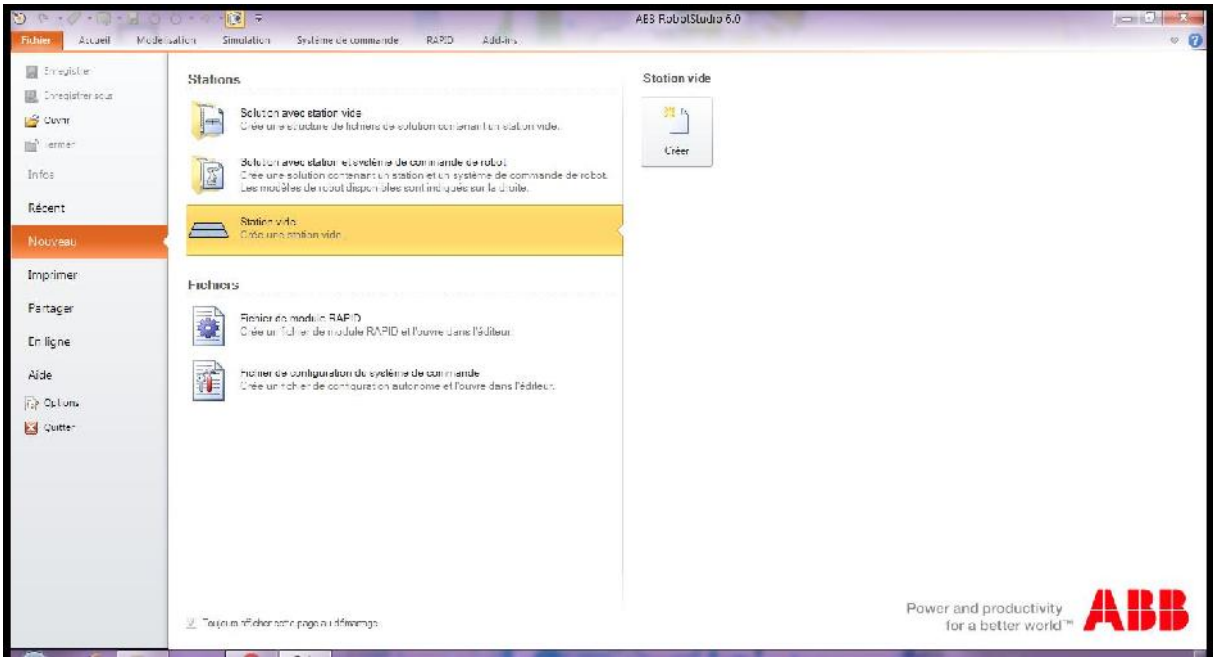


Figure (IV.1): Création d'une station vide.

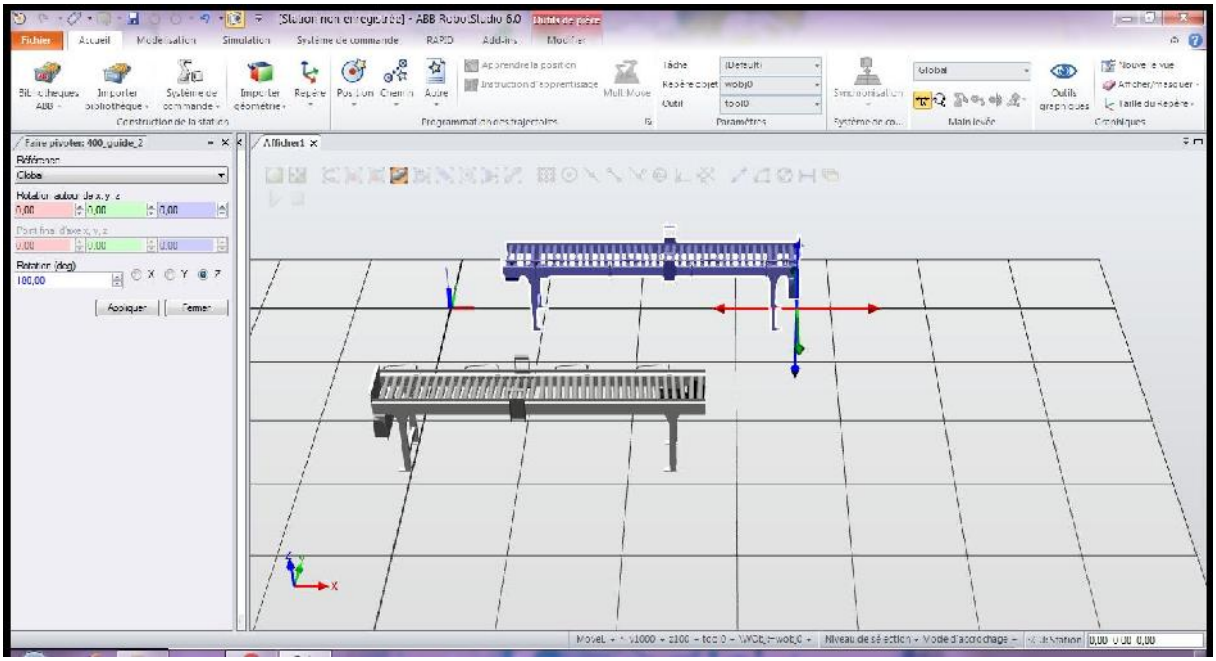


Figure (IV.2): Création les deux convoyeurs.

- Après avoir créé les deux convoyeurs, on à besoin de créer un objet a partir de l'instruction modélisation ⟶ solide ⟶ boite avec des dimensions 200.

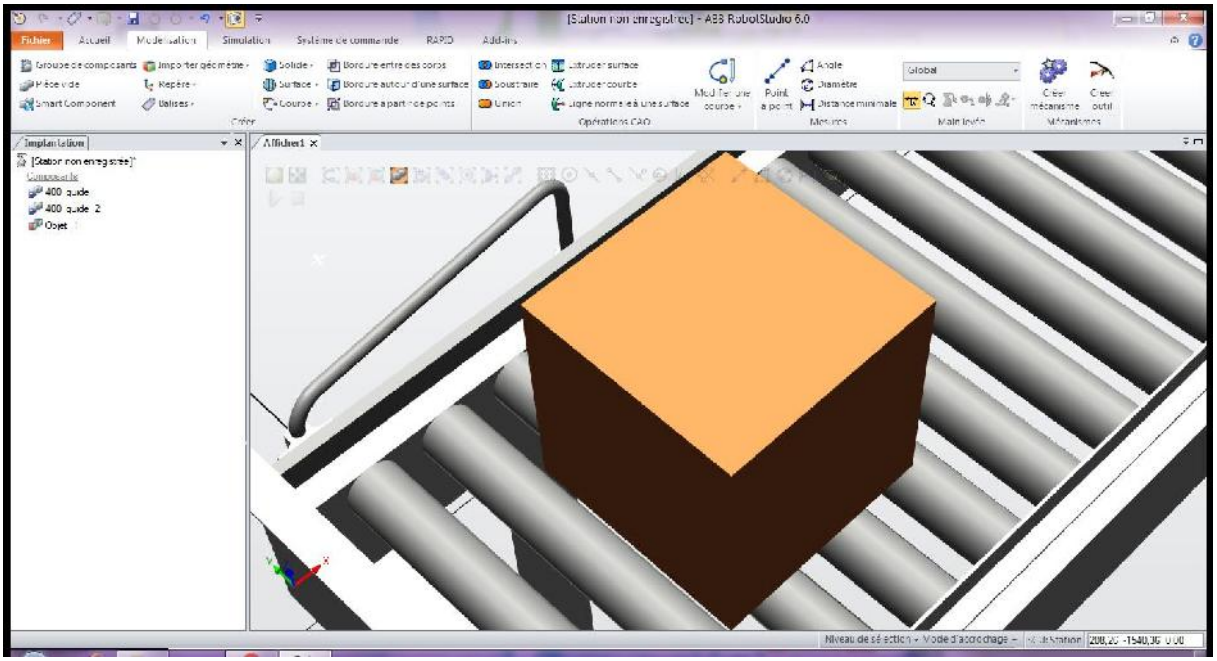


Figure (IV.3): Création de l'objet.

- Créer une position de l'objet à partir de l'instruction simulateur d'E/S.

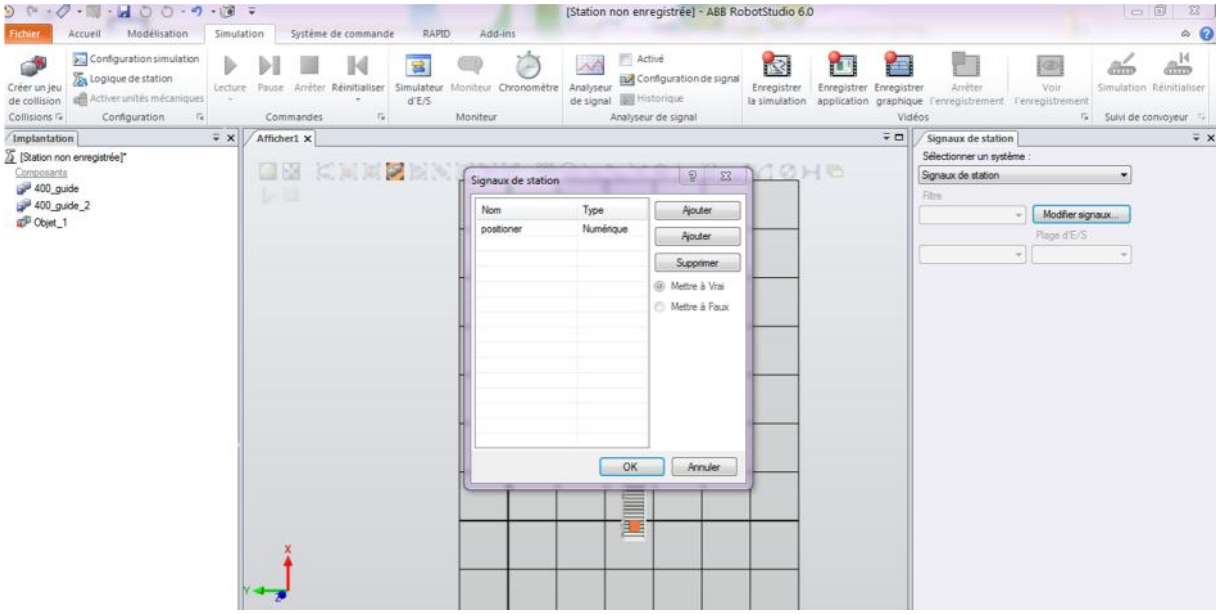


Figure (IV.4): Ajouter un signal numérique pour objet.

- Logique de station ⟶ modifier les signaux ⟶ ajouter(Numérique)⟶ nom signal (positionneur).
- On déduire la position et l'orientation de l'objet à partir de logique de station ajouter composant positionneur.

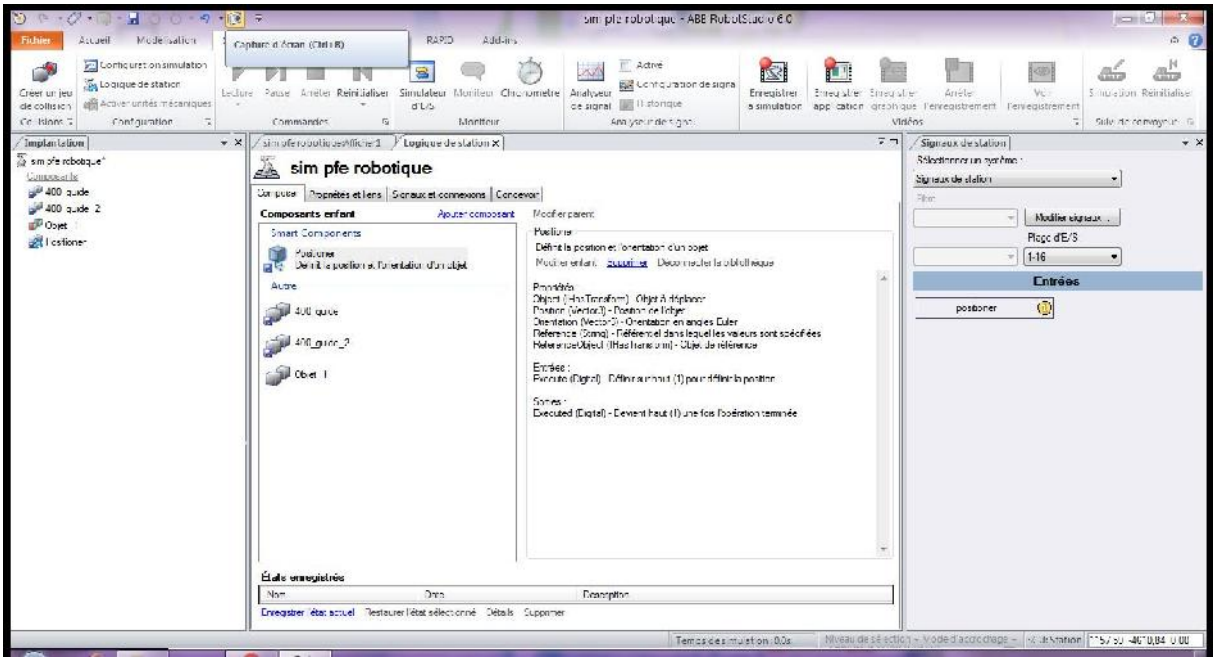


Figure (IV.5): Ajouter un composant (Positionneur).

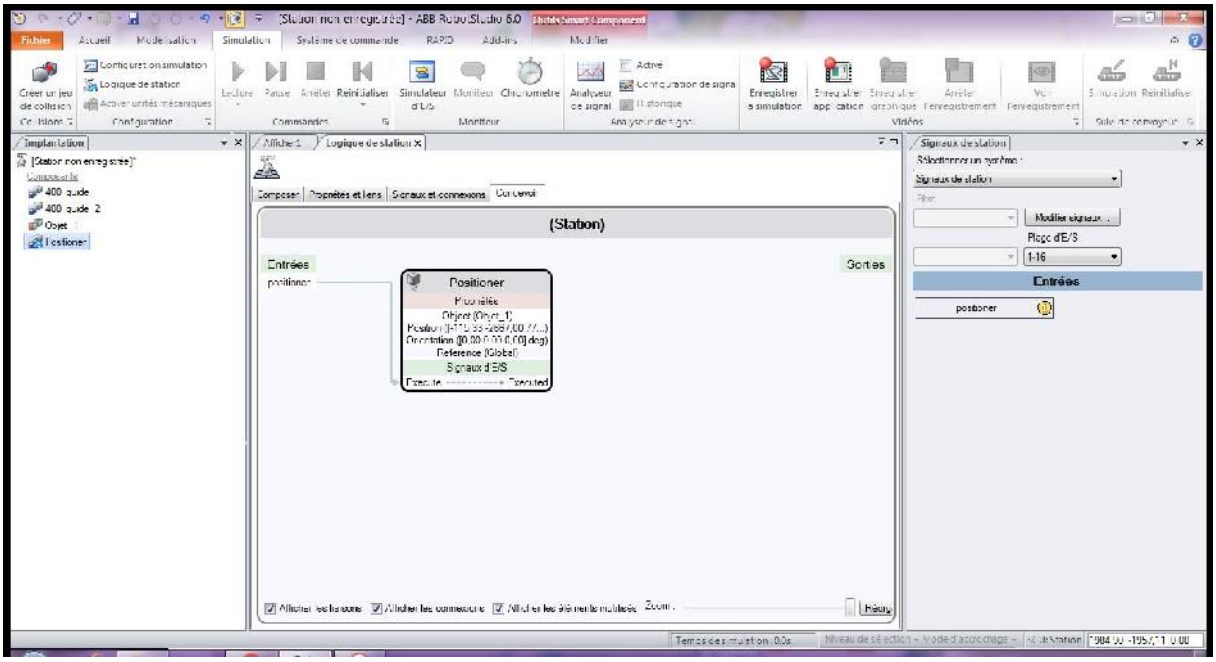


Figure (IV.6): La connexion entre la position de l'objet et composant (positionneur).

- Après on ajoute détecteur de plan 1et 2 pour capter l'objet entre en intersection avec le plan.

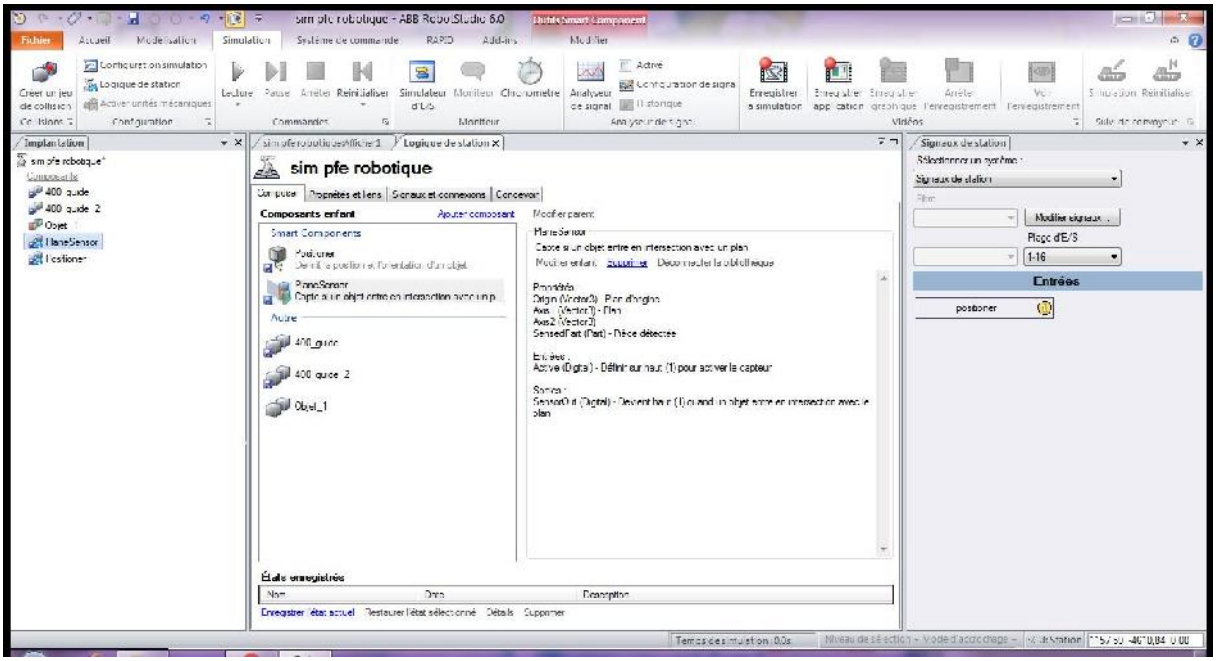


Figure (IV.7) : Capte l'objet.

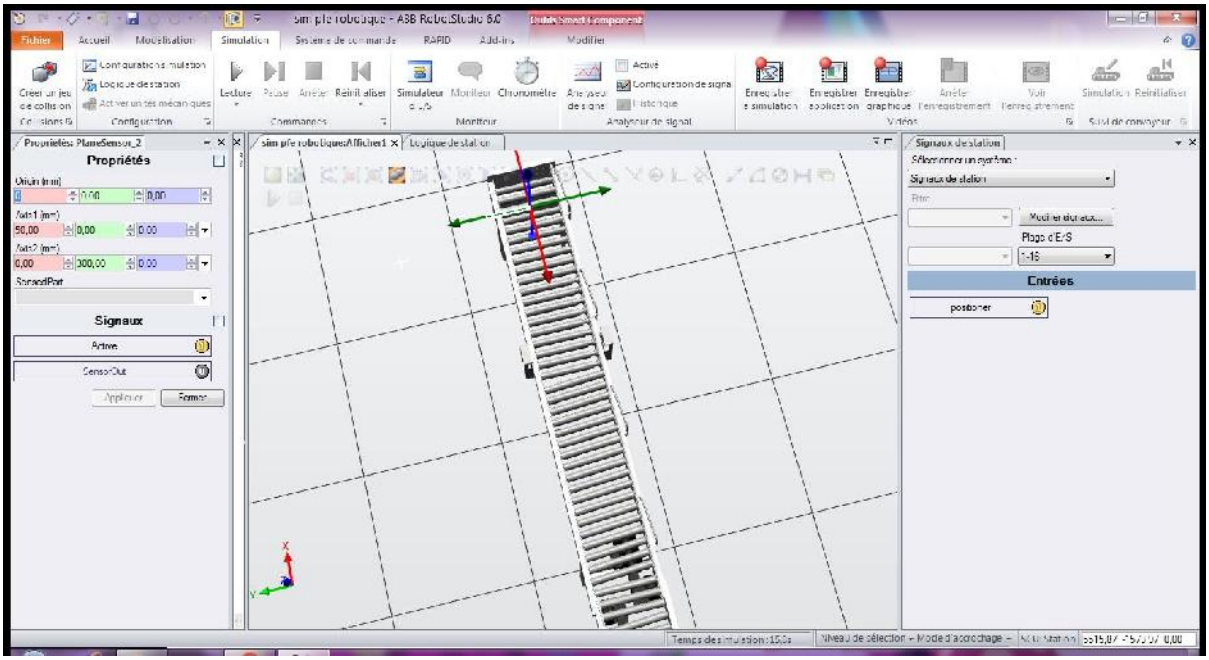


Figure (IV.8) : Déplacer le capteur de plan sur le convoyeur.

Remarque :

- On a déplacé les deux détecteurs de plan sur le convoyeur (détecteur de plan 1 pour le convoyeur 1 et le détecteur de plan 2 pour le convoyeur 2).

- Pour déplacer l'objet linéairement sur les 2 convoyeurs on ajouter un composant déménageur linéaire.

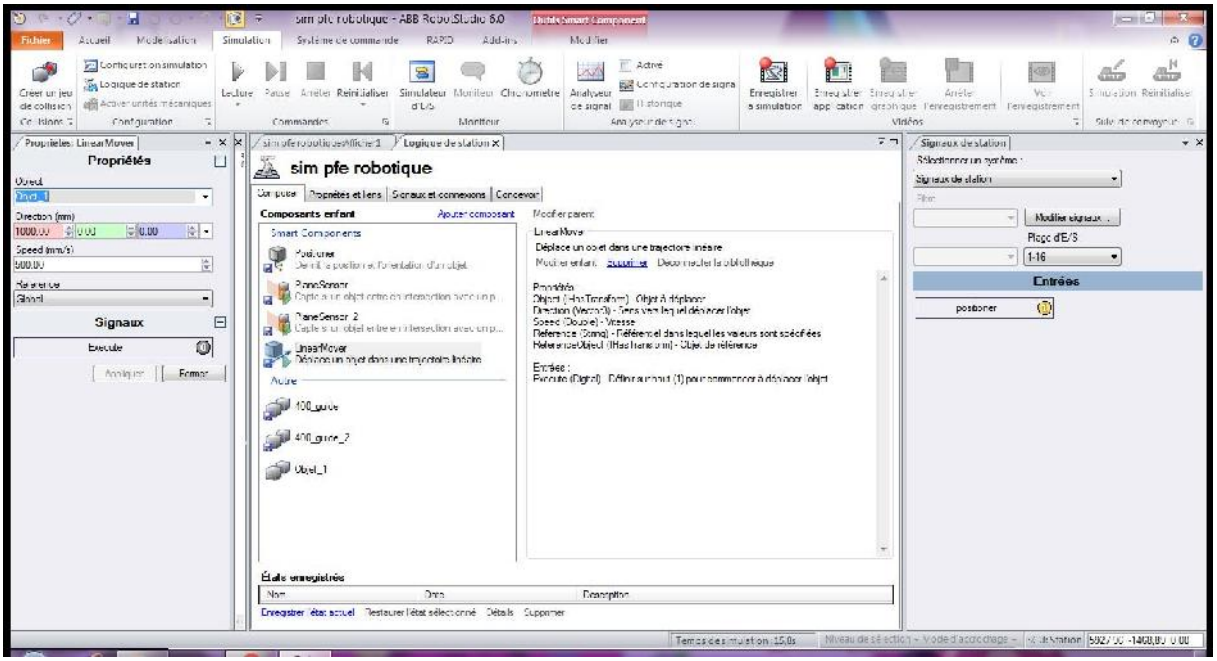


Figure (IV.9) : déplacer l'objet linéairement

- On ajouter les composants Simulateur d'évènements (pour émet des signaux lorsque la simulation commence et s'arrête) et le deuxième composant LogicSRLatch (réinitialiser).

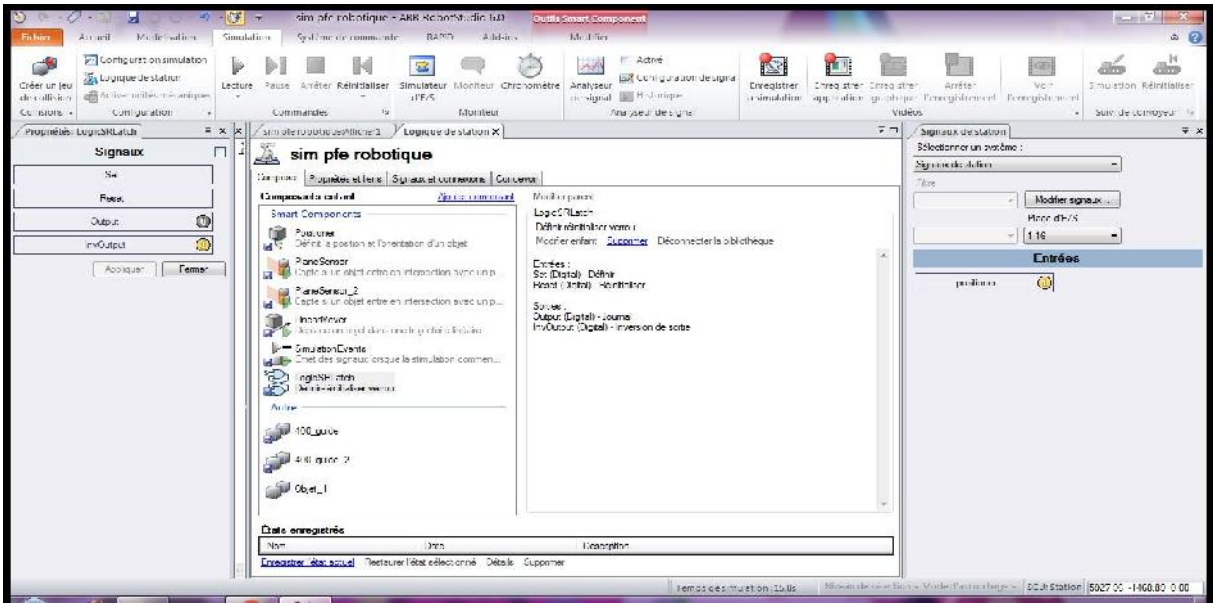


Figure (IV.10) : Ajouter les composants Simulateur d'évènements et LogicSRLatch

- Après on met la connexion entre tous ces composants.

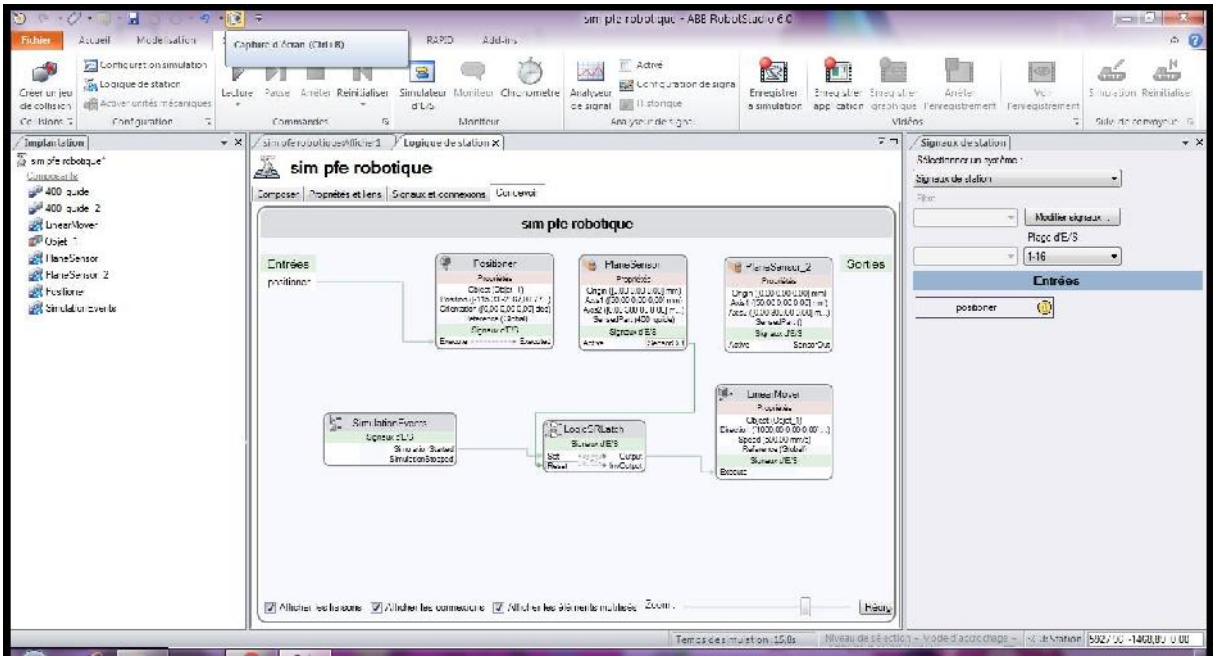


Figure (IV.11) : Connexion entre entrée et les autres composants

- Apres on importe le robot IRB 1600 (Capacité 6kg et rayon d'action de 1.45m) de bibliothèque ABB pour la tache de soudage.

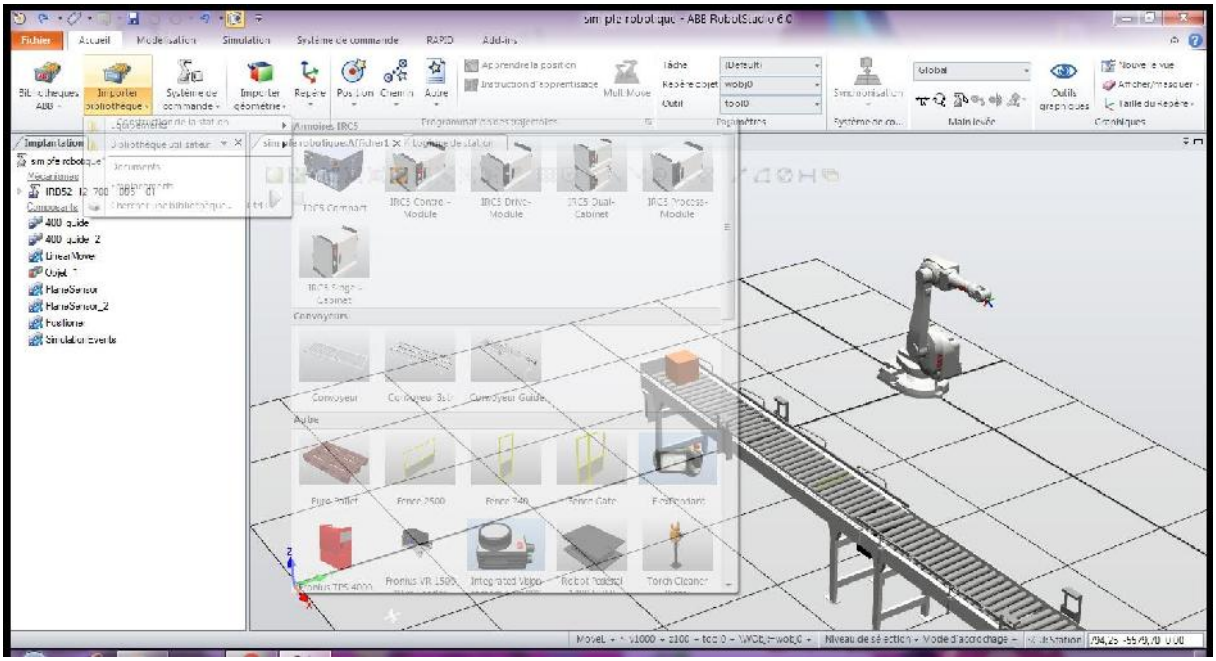


Figure (IV.12) : Sélection du robot IRB1600.

- On importe l'organe terminal en cliquant sur importer bibliothèques, équipement Binzel water 22.

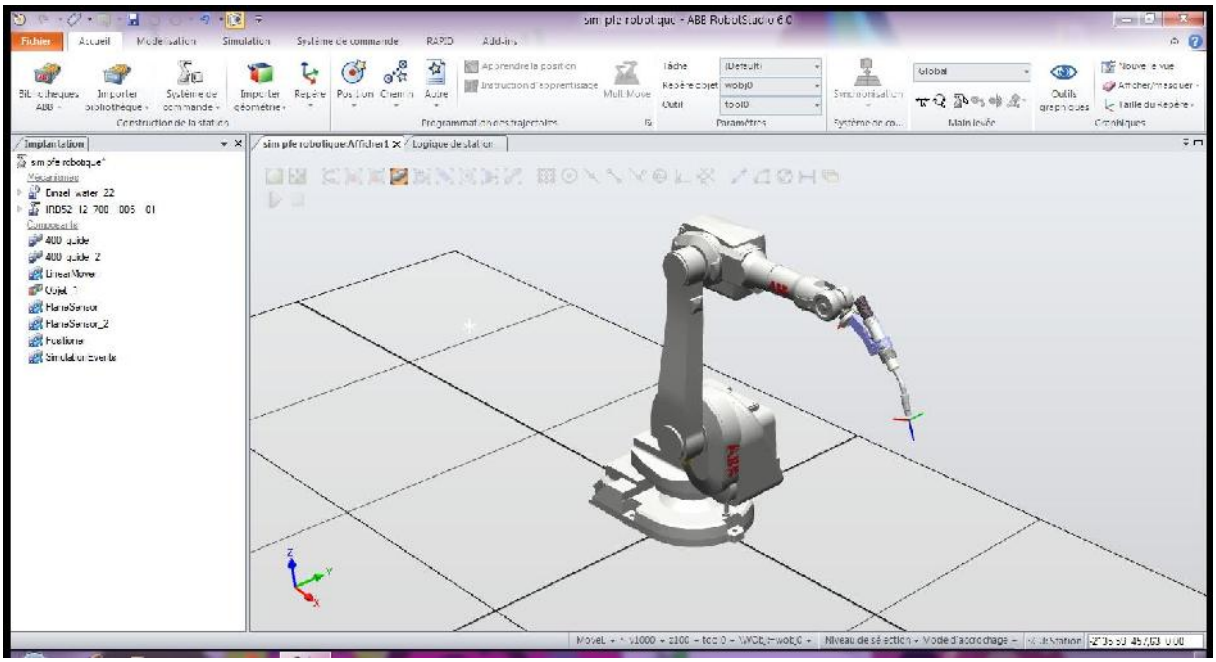


Figure (IV.13) : Attachement de l'outil avec le robot.

- Après de choisir outil on déplace le robot à la position du détecteur de plan 1.

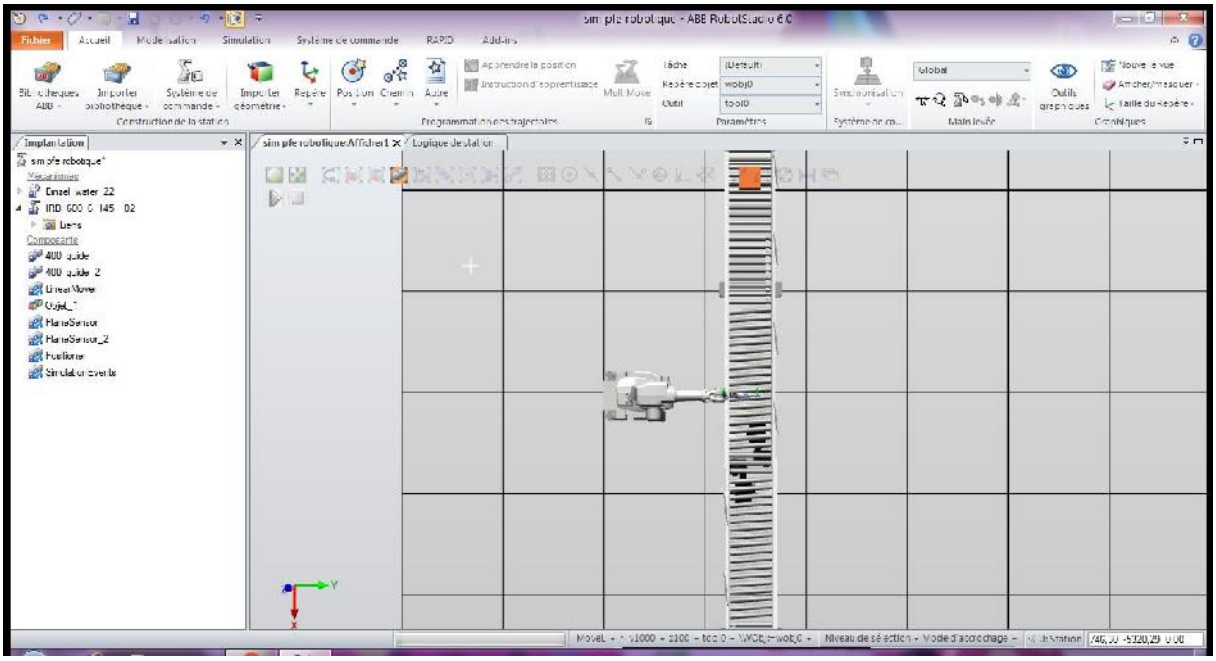


Figure (IV.14) : Déplacer le robot vers détecteur de plan.

- Ensuite, on crée les positions (les points cibles).

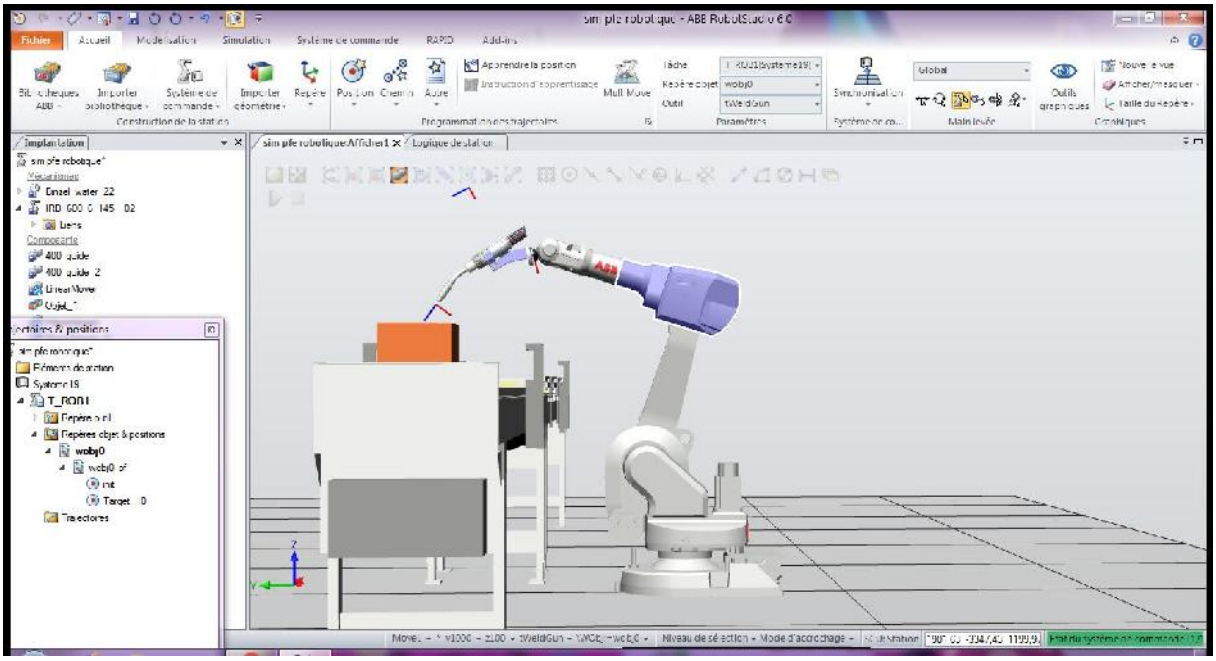


Figure (IV.15) : Création des points cibles.

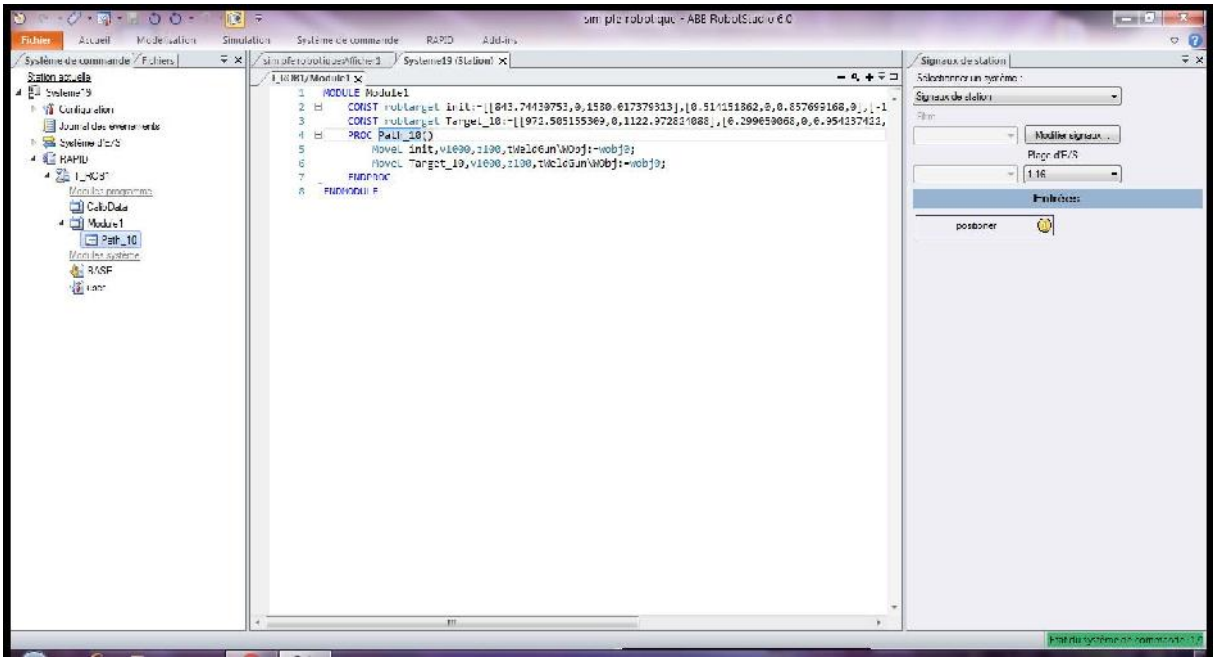


Figure (IV.16) : Insertion des instructions RAPID.

- On click sur éditeur de configuration pour ajouter des signaux numériques (entrée entrée numérique et SOR1 sortie numérique).

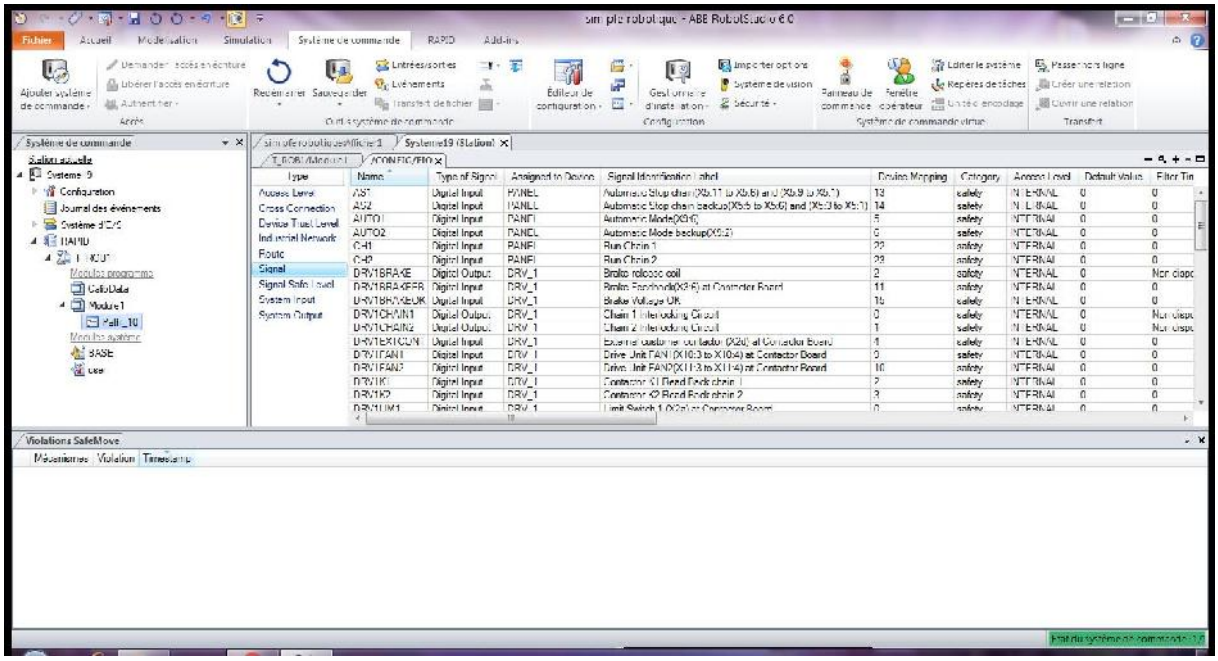


Figure (IV.17) : Ajouter des signaux

- Après on redémarre le système.

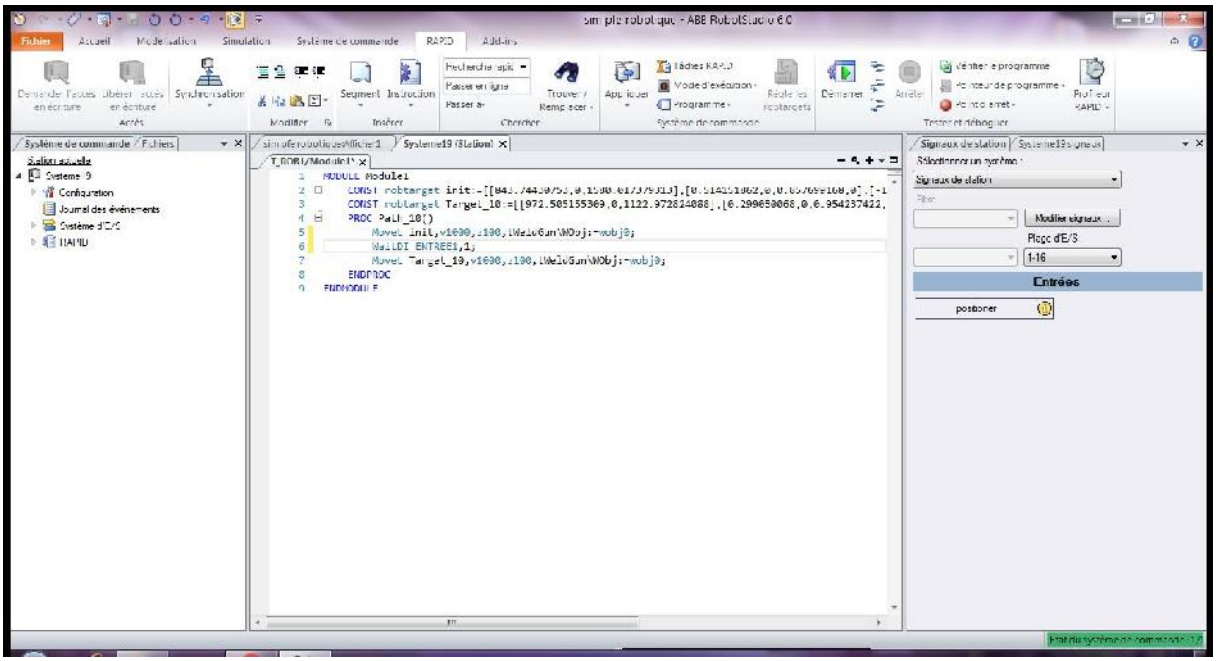


Figure (IV.18) : Programme RAPID de la tache (déplacement de l'objet).

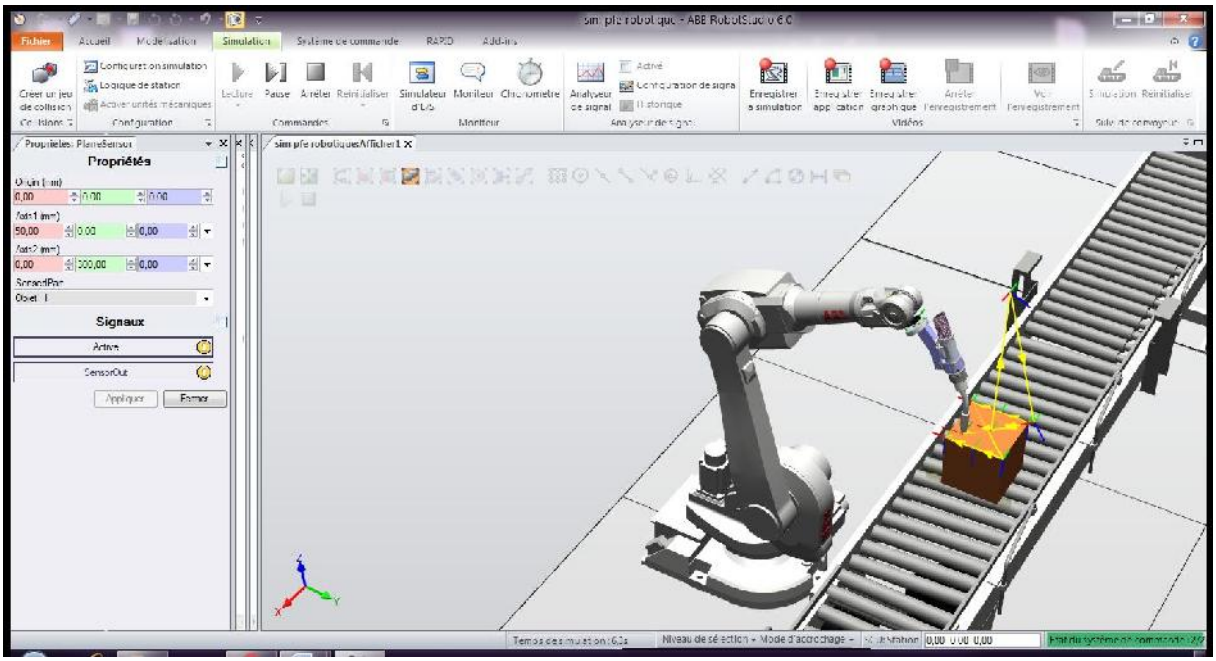


Figure (IV.19) : L'exécution de tache.

- On importe le deuxième robot IRB 1600 (Capacité 6kg et de rayon d'action 1.2m) de bibliothèque ABB pour la tache de palettisation.

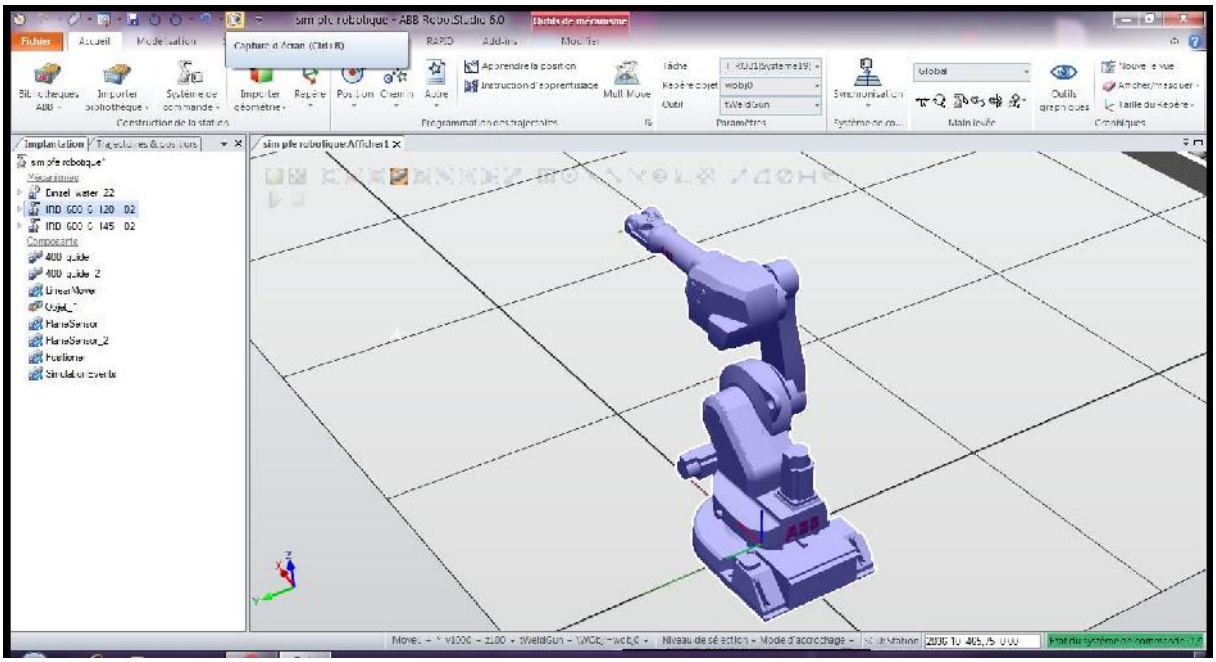


Figure (IV.20) : Sélection le deuxième IRB1600

- On importe l'organe terminal en cliquant sur Importer bibliothèques, équipement mon outil.

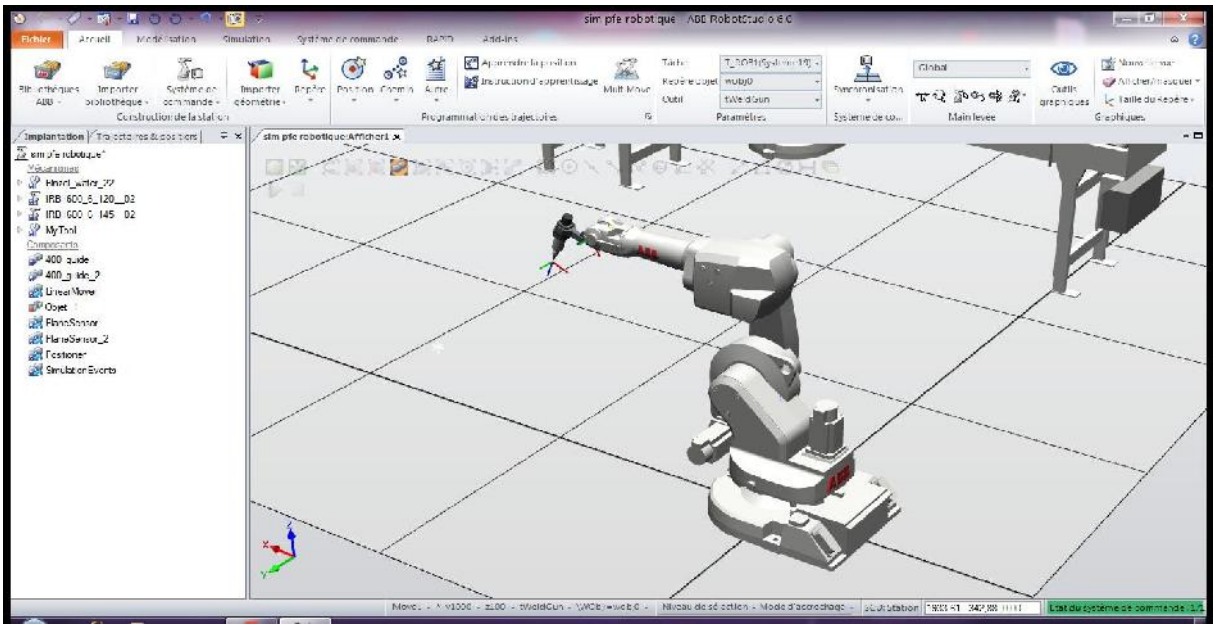


Figure (IV.21) : Attachement de l'outil avec le robot.

- On doit préciser après ça les cibles que le robot doit atteindre pour avoir notre pièce découpée en cliquant sur Target ; création Target ;
- Ensuit on crée les cibles (il suffit de cliquer sur les points désirés en utilisant le bouton droite de la souris et on aura toutes les cibles) ;
- Dernier on crée le chemin entre les différentes cibles pour que le robot puisse les atteindre en cliquant sur (path) chemin.

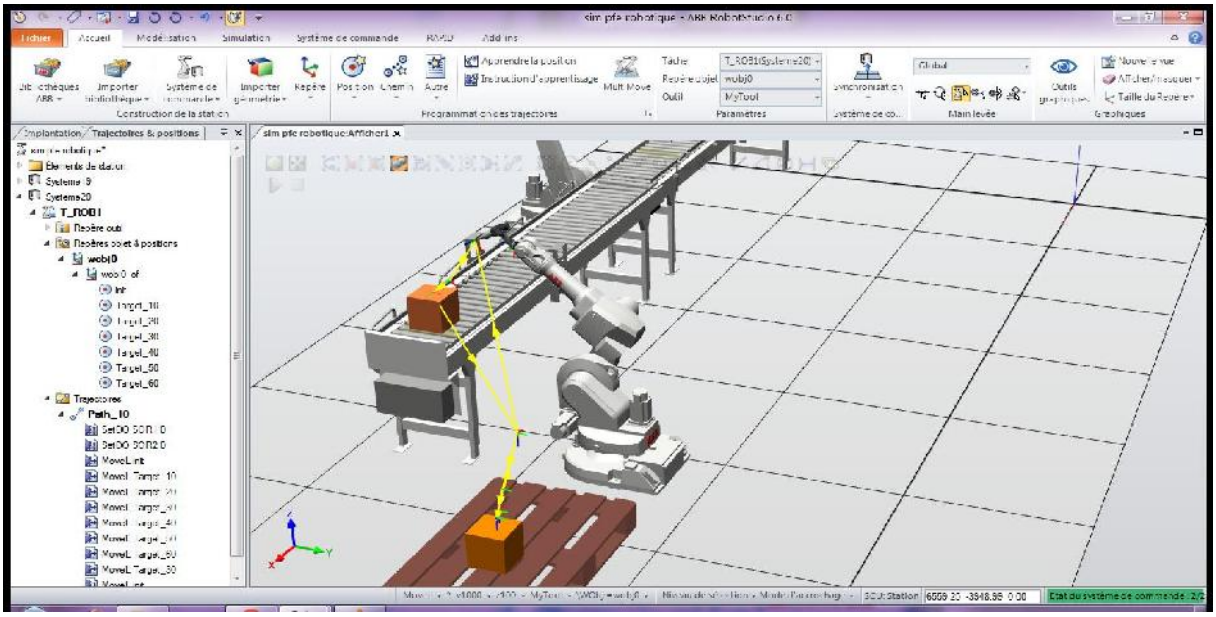


Figure (IV.22) : Création des positions.

- On ajoute de signaux Numériques (d'entré et sortie).

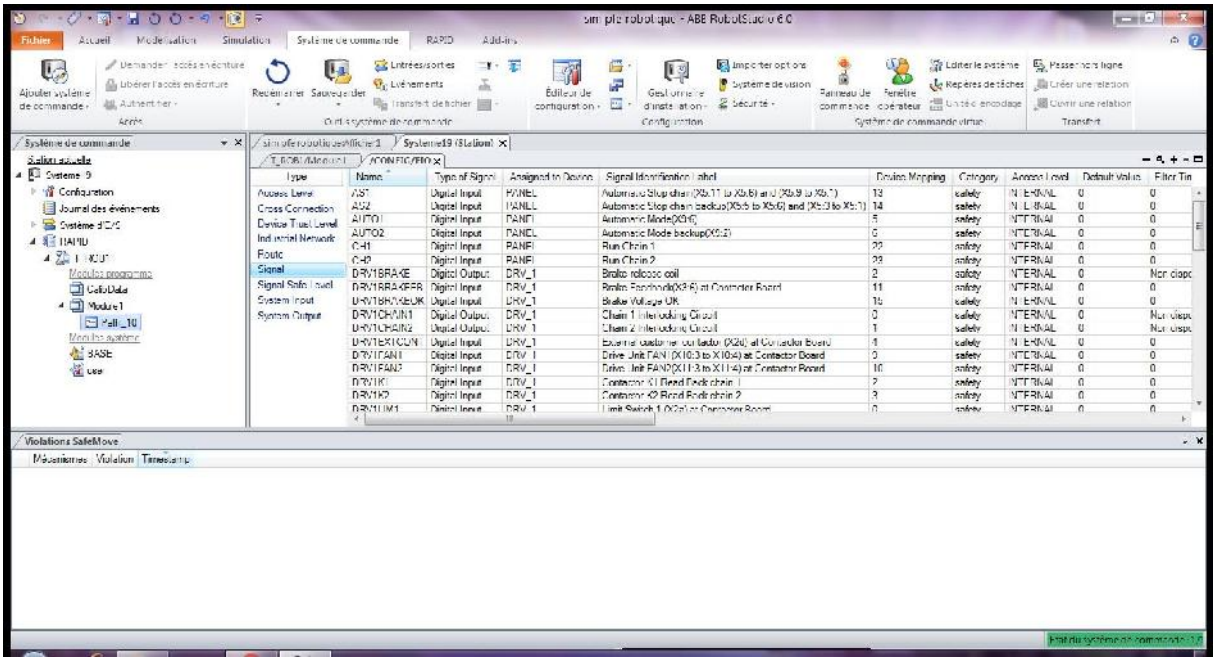


Figure (IV.23) : Ajouter des signaux (SOR1 et SOR2).

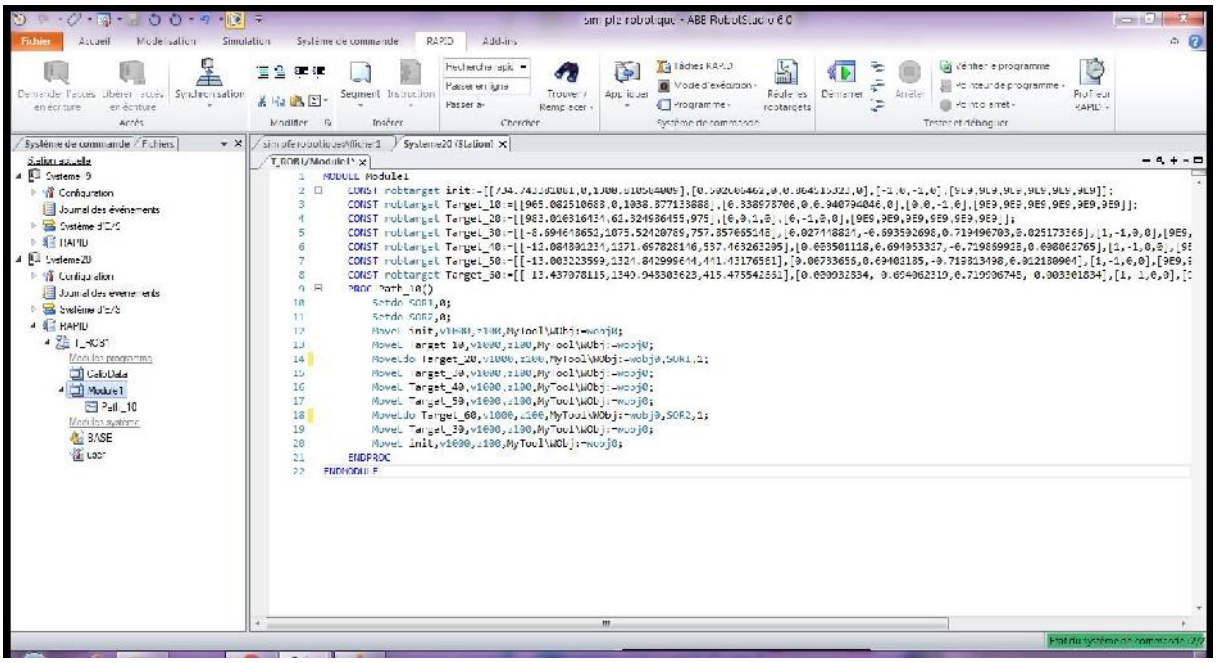


Figure (IV.24) : Programme RAPID de la tache (palettisation).

- On ajoute les composants Attacher (relie un objet) et Détacher (délie un objet lie)

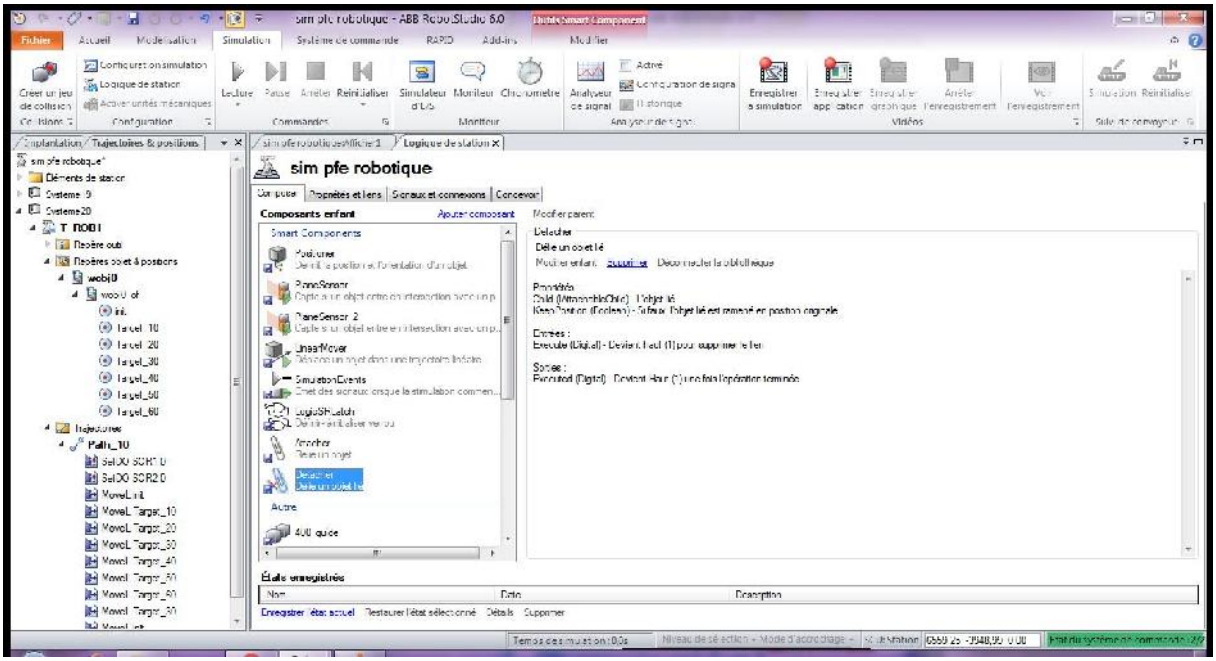


Figure (IV.25) : Ajouter les composants Attacher et détacher.

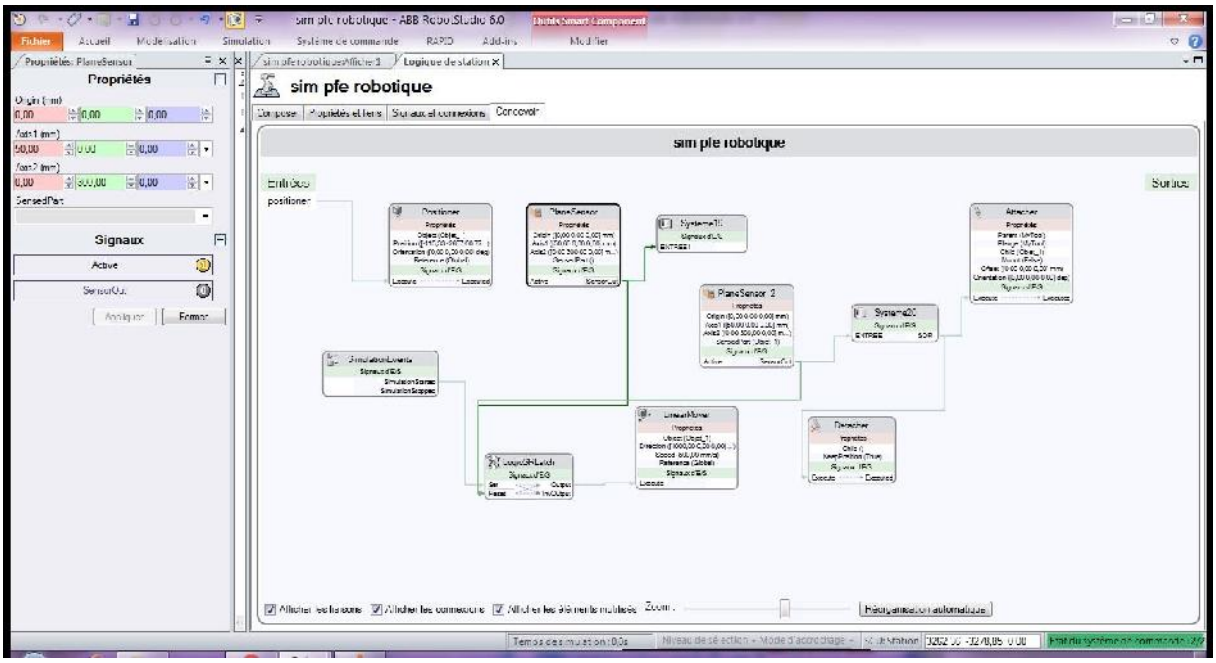


Figure (IV.26) : Connexion entre Entrée et les autres composants.

- Pour des raisons de sécurité, ces robots sont installés dans des cages pour éviter que l'on s'en approche trop.

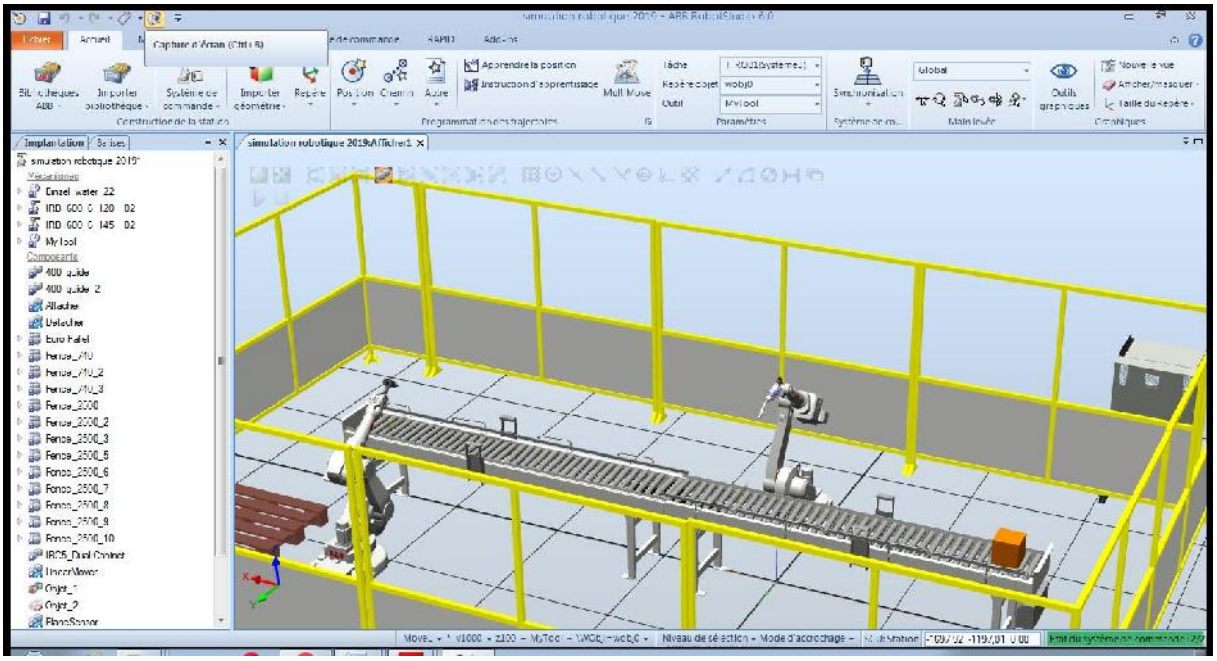


Figure (IV.27) : cellule industriel

- Simulation (les taches finale).

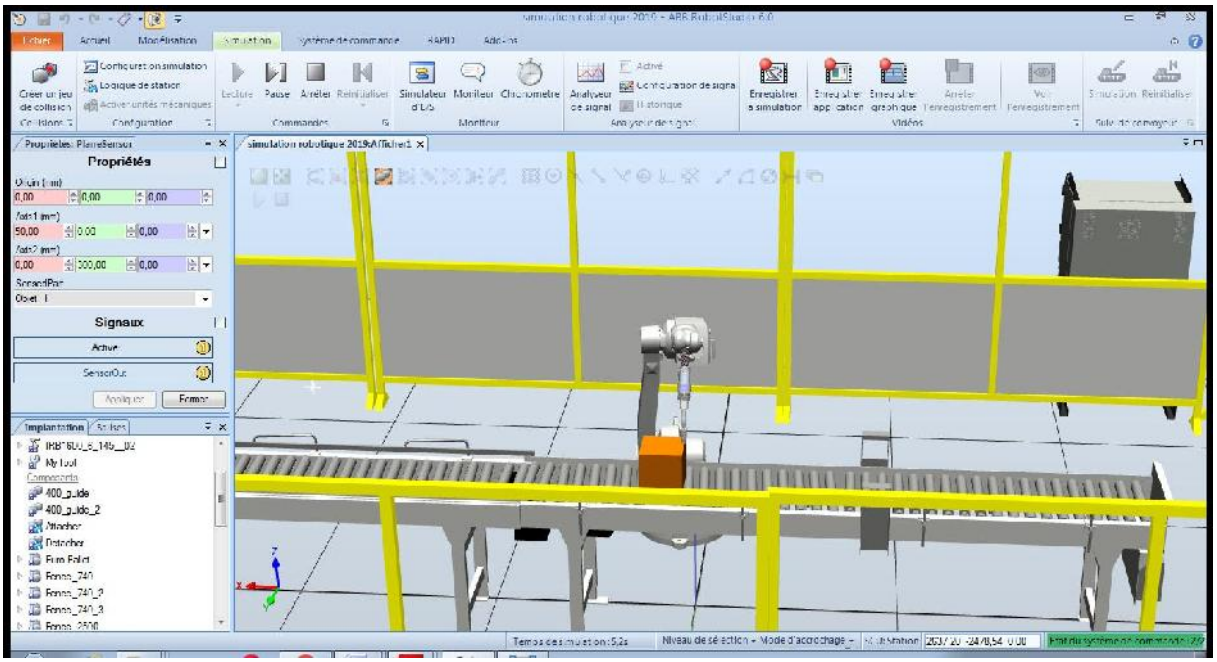


Figure (IV.28) : exécution de la tache de soudage

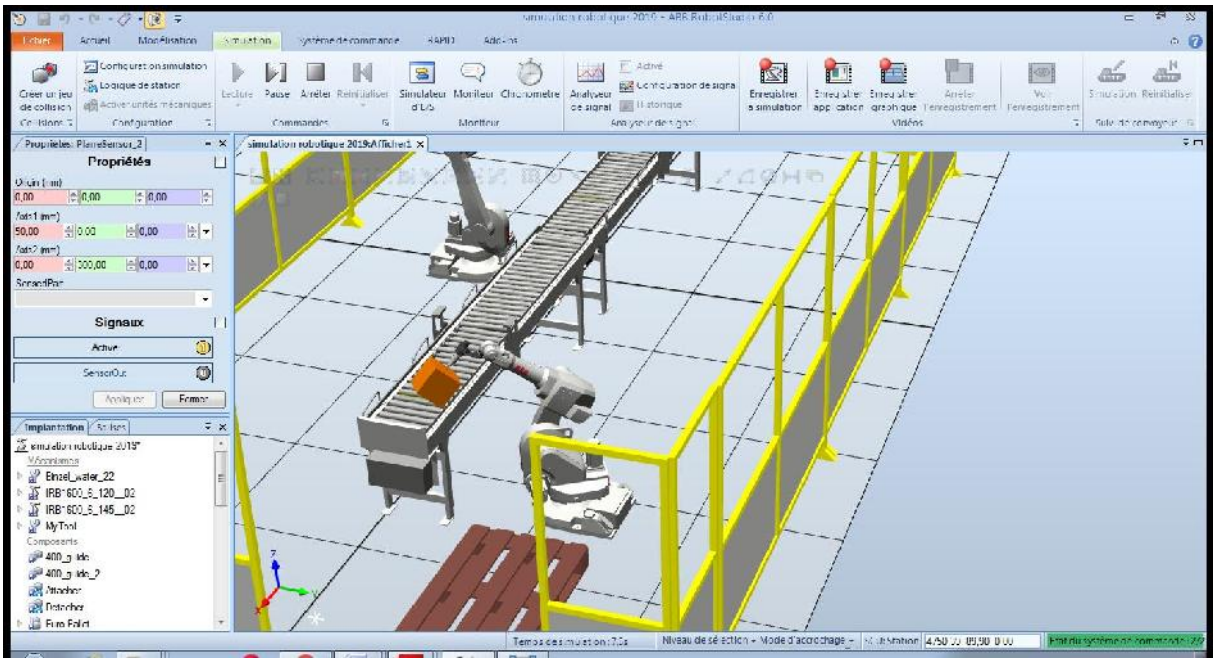


Figure (IV.29) : exécution de la tache palettisation

IV.2.Le programme RAPID

Pour le Robot 1 (Soudage) :

Syt1

MODULE Module1

CONST robtarget init:=[[843.74430753,0,1580.017379313],[0.514151862,0,0.857699168,0],[-1,0,-1,0],[9E9,9E9,9E9,9E9,9E9,9E9]];

CONST robtarget Target_10:=[[972.505155309,0,1122.972824088],[0.299050068,0,0.954237422,0],[-1,0,-1,0],[9E9,9E9,9E9,9E9,9E9,9E9]];

CONST robtarget Target_20:=[[817.777459542,55.56540206,975],[0,0,1,0],[0,-1,0,0],[9E9,9E9,9E9,9E9,9E9,9E9]];

CONST robtarget Target_30:=[[905.604459542,151.64040206,975],[0,0,1,0],[0,-1,0,0],[9E9,9E9,9E9,9E9,9E9,9E9]];

CONST robtarget Target_40:=[[805.604459542,151.64040206,975],[0,0,1,0],[0,-1,0,0],[9E9,9E9,9E9,9E9,9E9,9E9]];

CONST robtarget Target_50:=[[705.604459542,151.64040206,975],[0,0,1,0],[0,-1,0,0],[9E9,9E9,9E9,9E9,9E9,9E9]];

CONST robtarget Target_60:=[[905.604459542,-48.35959794,975],[0,0,1,0],[-1,0,-1,0],[9E9,9E9,9E9,9E9,9E9,9E9]];

```
    CONST robtarget Target_70:=[[805.604459542,-48.35959794,975],[0,0,1,0],[-1,0,-1,0],[9E9,9E9,9E9,9E9,9E9,9E9]];
    CONST robtarget Target_80:=[[705.604459542,-48.35959794,975],[0,0,1,0],[-1,0,-1,0],[9E9,9E9,9E9,9E9,9E9,9E9]];
    PROC Path_10()
      MoveL init,v1000,z100,tWeldGun\WObj:=wobj0;
      WaitDI ENTREE1,1;
      MoveL Target_10,v1000,z100,tWeldGun\WObj:=wobj0;
          MoveL Target_20,v1000,fine,tWeldGun\WObj:=wobj0;
          MoveL Target_30,v1000,fine,tWeldGun\WObj:=wobj0;
          MoveL Target_40,v1000,fine,tWeldGun\WObj:=wobj0;
          MoveL Target_50,v1000,fine,tWeldGun\WObj:=wobj0;
          MoveL Target_60,v1000,z100,tWeldGun\WObj:=wobj0;
          MoveL Target_70,v1000,z100,tWeldGun\WObj:=wobj0;
          MoveL Target_80,v1000,z100,tWeldGun\WObj:=wobj0;
          MoveL Target_20,v1000,z100,tWeldGun\WObj:=wobj0;
          MoveL init,v1000,z100,tWeldGun\WObj:=wobj0;
    ENDPROC
ENDMODULE
```

Pour le Robot2 (palettisation) :

```
Syt2
MODULE Module1
    CONST robtarget init:=[[734.743381081,0,1308.810584009],[0.502606462,0,0.864515323,0],[-1,0,-1,0],[9E9,9E9,9E9,9E9,9E9,9E9]];
    CONST robtarget Target_10:=[[905.082510688,0,1038.877133888],[0.338978706,0,0.940794046,0],[0,0,-1,0],[9E9,9E9,9E9,9E9,9E9,9E9]];
    CONST robtarget Target_20:=[[983.010316434,62.324986455,975],[0,0,1,0],[0,-1,0,0],[9E9,9E9,9E9,9E9,9E9,9E9]];
    CONST robtarget Target_30:=[[648.838898198,669.976017694,1200.366172099],[0.153186446,-0.355809021,0.919024855,0.072987461],[0,-1,0,0],[9E9,9E9,9E9,9E9,9E9,9E9]];
```

```
    CONST robtarget Target_40:=[[4.141199512,932.653071681,1200.366365908],[0.114878697,-0.673089522,0.719831389,0.124884552],[1,-1,0,0],[9E9,9E9,9E9,9E9,9E9,9E9]];
    CONST robtarget Target_50:=[[-0.37455786,1236.466370022,591.510104609],[0.006554775,-0.691152253,0.722674514,0.002672968],[1,-1,0,0],[9E9,9E9,9E9,9E9,9E9,9E9]];
    CONST robtarget Target_60:=[[-0.968765159,1309.260532746,375.364954032],[0.028667442,0.690576985,-0.721854247,0.034757031],[1,-1,0,0],[9E9,9E9,9E9,9E9,9E9,9E9]];
    PROC Path_10()
  Setdo SOR1,0;
  Setdo SOR2,0;
      MoveL init,v1000,z100,MyTool\WObj:=wobj0;
  WaitDI ENTREE,1;
      MoveL Target_10,v1000,z100,MyTool\WObj:=wobj0;
          MoveLdo Target_20,v1000,z100,MyTool\WObj:=wobj0,SOR1,1;
          MoveL Target_20,v1000,z100,MyTool\WObj:=wobj0;
          MoveL Target_30,v1000,z100,MyTool\WObj:=wobj0;
          MoveL Target_40,v1000,z100,MyTool\WObj:=wobj0;
          MoveL Target_50,v1000,z100,MyTool\WObj:=wobj0;
          MoveLdo Target_60,v1000,z100,MyTool\WObj:=wobj0,SOR2,1;
          MoveL Target_60,v1000,z100,MyTool\WObj:=wobj0;
          MoveL Target_30,v1000,z100,MyTool\WObj:=wobj0;
          MoveL init,v1000,z100,MyTool\WObj:=wobj0;
    ENDPROC
ENDMODULE
```

Conclusion

Le logicielle Robot Studio dédiée à la simulation des robots de type ABB permet aux concepteurs et utilisateurs des robots de mettre en œuvre virtuellement des robots dans leur environnement de travail. Ces outils permettent la simulation de robots dans une cellule de production complète en vue de la validation d'un nouveau système ou/et de l'optimisation d'une cellule existante. En plus de la conception et de l'analyse des cellules de production, ce programme est essentiellement utilisé pour la programmation hors-ligne des robots.

Conclusion générale

La robotique peut être définie comme l'ensemble des techniques et études tendant à concevoir des systèmes mécaniques, informatiques ou mixtes, capables de se substituer à l'homme dans ses fonctions motrices, sensorielles et intellectuelles.

L'objectif de ce travail porte sur la planification et la programmation d'une tache susceptible de ramener la précision requise et la facilité d'exécution.

La première partie est consacrée à donner une vision plus large sur le domaine d'application de la robotique et les différents concepts utilisés dans les différents milieux d'utilisation médical, industrielle.

Dans la seconde partie on présente les différentes approches et méthodes de calcul pour définir les positions et les paramètres des positions des éléments du robot. On abordait la modélisation en appliquant la convention de Denavit-Hartenberg pour le calcul du modèle géométrique qu'a nous permis la localisation de l'effecteur à n'importe quel point de l'espace de travail.

Dans le troisième chapitre on a démontré la programmation graphique et les commandes de déplacement qui ont une importance dans la programmation et la planification des taches.

Enfin dans le quatrième chapitre on a utilisé la programmation CFAO RoboStudio qui offre à son utilisateur la facilité et la richesse de conception et de planification des tâches couramment utilisées en industrie, de la plus simple à la plus complexe, profitant ainsi du repère du superviseur et de la haute qualité graphique en 3D.

Perspectives

Dans les conclusions de ce mémoire, plusieurs parties restent à découvrir pour mieux maitriser ce domaine qui est à la pointe de la technologie moderne et de permettre de faire des approches plus optimales et de réduire au maximum les erreurs de conception, les défauts de positions et d'atteindre les résultats souhaités dans un future si possible proche.

Bibliographie

[1] ISO, Robots et composants robotiques {Vocabulaire, ISO 8373, 2012}.

[2] Université ABOU-BEKR BELKAID - TLEMCEN FACULTE DES SCIENCES DE L'INGENIEUR DEPARTEMENT D'AUTOMATIQUE, Modélisation des robots, Master Automatique Mme S.BORSALI.

[3] PDF «Cours de robotique » Ensps 3A Master ISTI, Jacques Gangloff

[4] Modeling, Identification & Control of Robots, W. Khalil, E. Dombre, Hermes Penton Science 2002

[5] PDF Notes de cours GPA546 Ilian Bonev, ing. Yanick Noiseux, ing. 21 septembre 2014

[6] UNIVERSITE KASDI MERBAH OUARGLA FACULTE DES SCIENCES APPLIQUEES DEPARTEMENT DE GENIE MECANIQUE Cours de Robotique

Master Génie Mécanique Dr. Belloufi Abderrahim Mise à jour 2015

[7] Pollard Jr., W.L.G., and <Spray painting machine>, brevet américain No. 2 213 108, déposé le 29 octobre 1934, accepté le 27 aout 1940.

[8] Roselund, H.A., < Means for moving spray guns or other devices through predetermined Paths >, brevet américain No. 2 344 108, déposé le 17 aout 1939, accepté le 14 mars 1944.

[9] Rosheim, M.E., Robot Evolution {the Development of Anthrobotics, John Wiley & Sons , 1994}.

[10] Westerlund, L., the Extended Arm of Man {A History of the Industrial Robot, Information's_ orlaget, 2000.

[11] Makino, H., Kato, A., ET Yamazaki, Y., < Research and commercialization of SCARA robot >, International Journal of the Robotics Society of Japan, Vol. 23, No. 5, pages 61-62, 2007.

[12] IFR, World of robotics 2013.

[13] Université Mohamed Khider Biskra Département de Génie Electrique Mémoire de Fin d'Etudes Proposé et Dirigé Par: Mr. GUESBAYA TAHAR /2010

[14] W.Khalil ; E. Dombre Baszs de modélisation et de la commande des robots manipulateurs de types séries ; 22 janvier 2012 PDF

[15] I.Bonev, Y. Noiseaux Notes de cours GPA546 .27 mars 2013 PDF

[16] Robotique-Jean-Louis Boimond –Université angers

[17] Université HADJ LAKHDAR Batna- Département génie industriel- thème : optimisation d'un processus soudage effectue par bras manipulateur –MEDJGHOU Aicha.

[18] Dandan Fang. Diagnostic et adaptation des trajectoires robotiques en projection thermique. Mécanique physique, Université de Technologie de Belfort-Montbeliard, 2010.

[19] thèse de doctorat école doctorale sciences pour l'ingénieur et microtechniques : Zhenhua CIA. Programmation robotique en utilisant la méthode de maillage et la simulation thermique

Du procédé de la projection thermique. Autre. Université de Technologie de Belfort-Montbeliard, 2014.

[20] Thèse de doctorat école doctorale Sciences physique pour l'ingénieur et microtechniques ; Programmation robotique hors ligne et contrôle en temps réel des trajectoires ; université de Belfort – Montbéliard .

[21] PDF robotisation mode d'emploi

[22] Fabio MORBIDI Laboratoire MIS Équipe Perception Robotique Université de Picardie Jules Verne ; Licence Professionnelle Automatisme et Robotique session 2017 ; Initiation a la robotique.

[23] Khaled Arrouk. Techniques de conception assistée par ordinateur (CAO) pour la caractérisation de l'espace de travail de robots manipulateurs parallèles. Autre. Université Blaise Pascal - Clermont- Ferrand II, 2012.

[24] UNIVERSITE ABOU BEKR BELKAID-TLEMCENFACULTE DE TECHNOLOGIE

DEPARTEMENT DE GENIE MECANIQUE ; Projet de fin d'étude master En Génie Mécanique ; THÈME Rétro-conception du bras horizontal de robot manipulateur de la cellule flexible (Tlemcen) 2013

[25] Université de Biskra ; Faculté des Sciences et de la Technologie , Département de Génie Mécanique ; Mémoire de Fin d'Etudes ;Étude de la Robotique pour le calcul de la position d'un bras manipulateur juin 2014.