الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne Démocratique et Populaire
وزارة التعليم العالي والبحث العلمي
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 coopération de robots manipulateurs avec le logiciel RobotStudio

Présenté et soutenu par :

Nom : ALOUANI **Prénom :** Karim

Nom : DJERIOU **Prénom :** Zakaria

Devant le jury composé de :

Nom et prénom	Grade	Etablissement	Qualité
OTSMANI Zineb	MCA	IMSI – Univ Oran 2	Président
KACIMI Abderrahmane	MCB	IMSI – Univ Oran 2	Encadreur
	MAA	IMSI – Univ Oran 2	Examinateur

Année : 2020/2021

Remerciement

C’est avec une profonde reconnaissance et considération particulière que je remercie :

- Mes parents qui m'ont toujours entouré et motivé sans cesse à devenir meilleur et à me surpasser.
- Mon encadreur Mr **KACIMI Abderrahmane** d’avoir accepté de superviser ma thèse, pour votre disponibilité et votre soutien.
- Je souhaite adresser mes remerciements les plus sincères à Mme **OTSMANI Zineb** qui a contribué à ma formation, merci aussi pour vos conseils, votre temps et votre soutien.
- Tous mes amis que je ne peux nominativement citer, qu’ils trouvent ici l’expression de ma profonde reconnaissance …

Karim

Dédicace

- Je dédie ce travail A **ma chère maman**, la personne qui a beaucoup sacrifié pour moi sans elle Je n’aurais eu la volonté d’atteindre ce niveau .Que Dieu puisse la garder éternellement heureuse.
- Merci du fond du cœur. A **mon cher père** qui m’a toujours aidé et soutenu durant toutes ces années de sacrifices et des moments difficiles, que dieu le garde.
- A **mes chers frères** qui m’ont toujours soutenu, à **ma chère sœur**. A **mes chers Amis** Et à tous ceux qui m’ont aidé de prêt ou de loin à finaliser cette thèse et qui m’ont contribué dans ma formation.

Zacky

Sommaire

CHAPITRE 1 : Introduction à la robotique

CHAPITRE 2 : Modélisation mathématique pour la commande des robots

CHAPITRE 3 : Programmation des taches avec RobotStudio

CHAPITRE 4 : La simulation avec RobotStudio

Glossaire des acronymes

DDL	Degré 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 figures

Résumé

Notre travail consiste à simuler une coopération de robots manipulateurs, commandés par le logiciel RobotStudio d'ABB.
Dans un tout premier temps, nous nous intéresserons aux définitions et généralités sur les robots ainsi qu'à leurs caractéristiques, paramètres et domaines d'application.
Dans un second temps, les outils mathématiques pour la planification des tâches industrielles sont abordés,
De cette façon, nous avons terminé avec la planification des tâches industrielles de nos cellules robotiques et ensuite l'application sous le logiciel et la conception des tâches.

Mots-clés : RobotStudio, cellule, ABB, synchronisation, programmation, industrie, articulation, apprentissage, commande.

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.
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 coopération de robots manipulateurs avec le logiciel RobotStudio.

CHAPITRE 1 : Introduction à la robotique

1. Introduction :

Le premier chapitre, illustre l'historique de la robotique, et les inventions développées par l'homme depuis 2000 ans. L'homme a depuis toujours été charmé par la possibilité de pouvoir recréer la vie. Les divers hommes d'airain, androïdes ou robots, crées dans la littérature, au cinéma ou dans la réalité, ont depuis la haute antiquité répondu à nos fictions. Sans remonter aux premiers concepts de machine remplaçant l'homme dès le 17ème siècle, la robotique est née en 1950, du croissement des besoins et des disponibilités de nouvelles technologies développées durant la 2ème guerre mondiale.
La robotique industrielle a répondu dans un premier temps au besoin de traiter de manière automatique des objets entre les machines de fabrications. Le robot devait avoir un organe au minimum de saisie de pièce pouvant suivre des trajectoires programmable. Actuellement Le robot industriel est un produit major et répandu à plus d'un million d'exemplaires, utilisé dans nombreux domaines. La robotique est dévisagée comme un élément clé de la compétitivité des entreprises industrielles, au point d'éveiller des soutiens publics massifs.

1.1. Historique :

Le concept de robot date de plusieurs siècles, mais le terme robot fut engendré par let chèque Karel Capek dans une pièce de théâtre écrite en 1920: "RUR ou les robots universels de Rossum". Il est certain que depuis fort longtemps, les réalisateurs d'automates ont cherché à pouvoir créer à leurs machines des comportements adaptés aux circonstances. Malheureusement, jusqu'au 20ème siècle, les techniques étaient trop archaïques pour permettre de telles réalisations. Il fallut attendre 1959 pour que Georges Devol invente une machine originale, polyvalente et reprogrammable, ce qui a permis au robot de procurer une réalité industrielle. Mais ce ne fut que vers la fin des années 1970 que les robots industriels de première génération ont vu le jour.

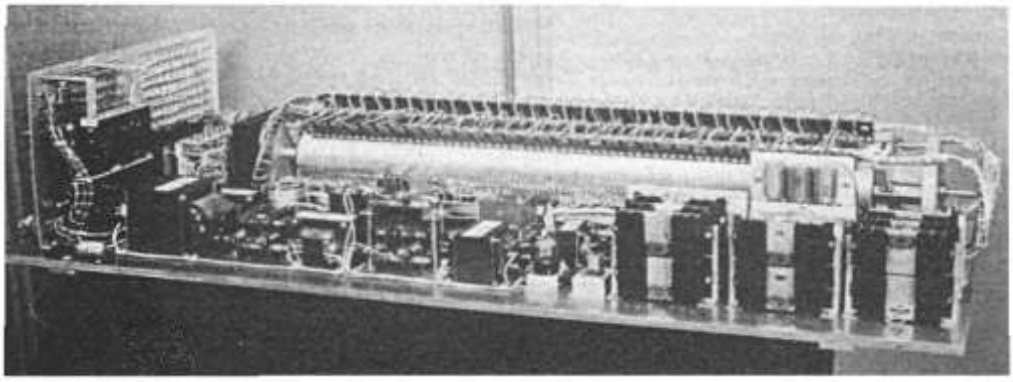


Figure 1-1 : Premier robot industriel de G.Devol

Depuis que les premiers robots industriels sont apparus, leur progression a été considérable et, chaque année, de nouveaux secteurs industriels s'ouvrent à la robotisation, les équipements devenant plus diversifiés, plus adaptables et surtout moins chers. Les robots sont devenus tellement indispensables dans certains secteurs industriels que leur utilisation est essentielle à la survie économique des entreprises. Il est donc important de bien maîtriser leur technologie.
Le terme robot peut désigner une grande variété de réalisations technologiques, allant de simples dispositifs mécaniques exécutant des mouvements répétés, à des machines morphologiquement similaires aux bras humains et dotées d'une certaine intelligence.
Pour l'anecdote, la firme unimation qui a réalisé le premier robot vers 1960, était pratiquement la seule sur le marché américain quinze ans plus tard, et n'avait commercialisé que 1000 robots. Alors que la JIRA (association japonaise de robotique industrielle) annonçait en 1975 65000 robots ? Il s'est avéré plus tard que ce que certains appelaient simple automatisation, les autres l'appelaient robot !
De nos jours, on compte plus de 15 000 robots industriels au Japon, soit près d'un tiers du total mondial. Figure 1-2.

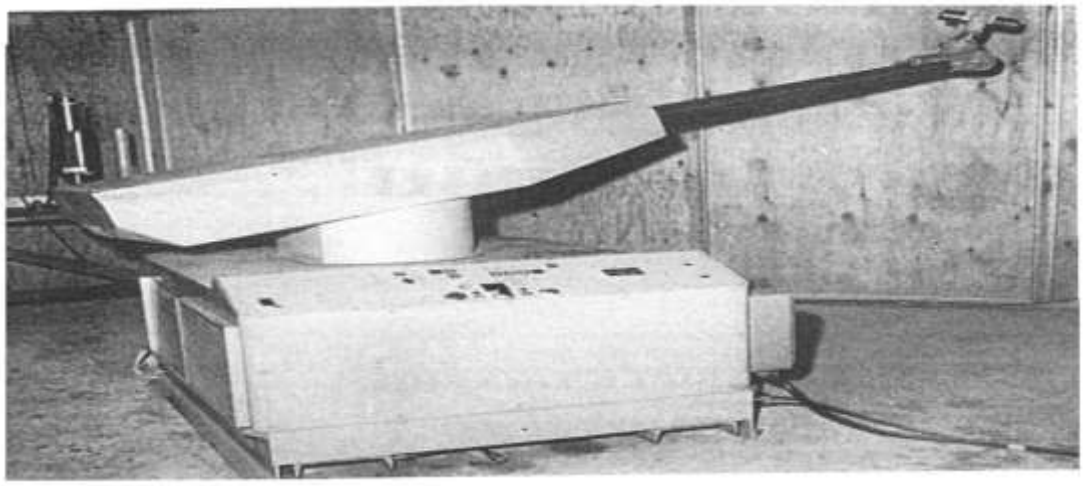


Figure 1-2 : Prototype UNIMATE (1959)

1.2. Définition

1.2.1. Définition d'un automate :

Un automate est une machine programmée pour exécuter une tâche précise dans un environnement donné [3].

1.2.2. Définition d'un robot :

Un robot est un automate doté de capteurs et d'effecteur lui donnant une capacité d'adaptation et de déplacement proche de l'autonomie. Un robot est un agent physique réalisant des tâches dans l'environnement dans lequel il évolue [3].

1.2.3. Définition d'un robot industriel :

Un robot industriel est un dispositif multi-axes semblable à un bras humain, composé souvent de six degrés de liberté, de trois axes de positionnement et de trois axes d'orientation, permettant le déplacement et l'orientation d'un outil (effecteur) dans un espace de travail donné.
Les robots industriels sont largement utilisés dans le secteur automobile. Leur conception nécessite une bonne maîtrise technique et un très haut niveau d'ingénierie

1.2.4. Définition de la robotique industrielle :

La robotique industrielle est définie par l'Organisation internationale de normalisation (ISO) comme un dispositif de manipulation à commande automatique, polyvalent, reprogrammable et pouvant être programmé selon trois axes ou plus.

1.2.5. Définition d'un bras robotisé :

C'est le principal organe mécanique du déplacement. Il est généralement défini par un système d'articulation semblable à celui d'un bras humain. Il existe plusieurs formes de bras, dont certaines ressemblent davantage à un bras humain mais remplissent les mêmes tâches.

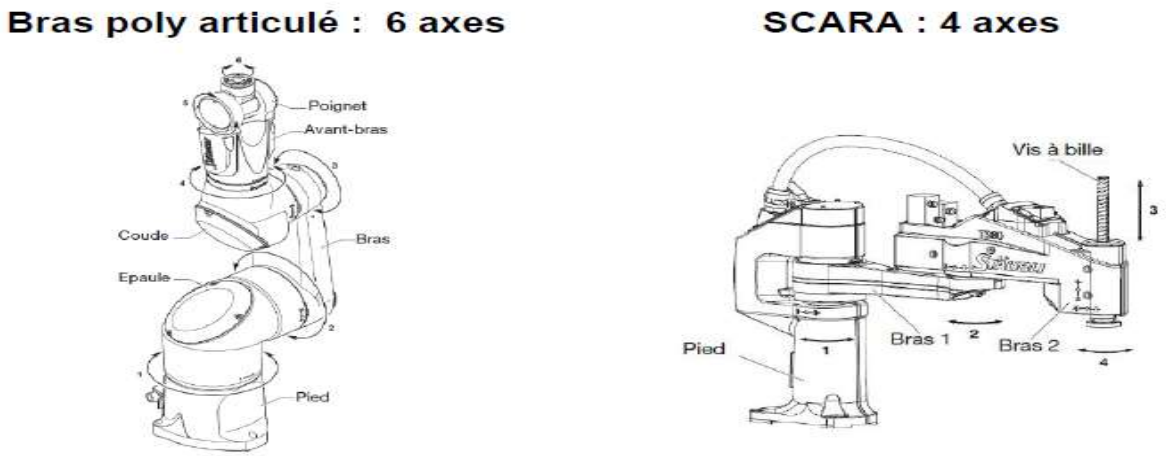


Figure 1-3 **:** Bras robotisé

1.3. Les types des robots industriels :

A - Robots articulés

Les robots articulés sont les plus courants des robots industriels. Ils ont l'apparence d'un bras humain, c'est pour cela qu'ils sont aussi nommés bras robotique ou bras manipulateur. Leurs articulations à multiples degrés de liberté permettent au bras articulé une grande variété de déplacements.

B - Les robots cartésiens

On appelle robots cartésiens les articulations de type prismatique pour le déplacement de l'outil, mais obligatoirement 3 rotoïdes pour l'orientation de ce dernier. Pour pouvoir déplacer et orienter l'effecteur dans toutes les directions en 3D, un tel robot a besoin de 6 axes : 3 prismatiques pour le déplacement, 3 rotoïdes pour l'orientation. Dans un environnement à 2 dimensions, 3 axes sont suffisants : 2 pour le déplacement, 1 pour l'orientation.

C - Les robots cylindriques

Les robots cylindriques se différencient par leur articulation rotative à la base et au moins une articulation pragmatique reliant les éléments. Ils peuvent se déplacer verticalement et aussi horizontalement par coulissement. La conception compacte de l'effecteur final permet au robot d'atteindre des endroits de travail étroits sans perte de vitesse.

D - Robots Delta

Les robots Delta sont aussi appelés robots à liaisons parallèles. Ils sont composés de liaisons parallèles reliées à une base commune. Les robots Delta sont très utiles pour les tâches de contrôle direct et les opérations à haute manœuvrabilité (comme les tâches rapides de prise et de mise en place). Les robots Delta tirent parti des systèmes de liaison à quatre barres ou à parallélogrammes.

E - Robots polaires

Les robots polaires sont des robots dotés d'articulations exclusivement rotoïdes. Afin de déplacer et d'orienter l'effecteur dans toutes les directions en 3D, un tel robot a besoin de 6 axes : 3 pour le mouvement, 3 pour l'orientation. Dans un environnement à 2 dimensions, seuls 3 axes sont nécessaires : 2 pour le mouvement, 1 pour l'orientation.

F - Les robots SCARA

SCARA est une abréviation de (Selective Compliance Assembly Robot Arm). Les robots SCARA se distinguent par leurs deux articulations parallèles qui assurent le mouvement dans le plan X-Y. Les arbres rotatifs sont placés à la verticale de l'axe de rotation. Les arbres rotatifs sont positionnés verticalement et l'effecteur final se déplace à l'horizontale. Les robots SCARA sont utilisés pour les tâches qui nécessitent un déplacement latéral précis et sont idéaux pour les applications d'assemblage.

1.4. Classification des robots :

A - Manipulateurs à distance :

Ou manipulateurs à commande manuelle. Ils sont contrôlés à distance et "en temps réel" par un opérateur humain. Cette télécommande se fait à plus ou moins grande distance par des signaux mécaniques, hydrauliques, ou le plus fréquemment électriques. Ces manipulateurs sont employés en forge, fonderie, rectification, ébavurage, milieux " hostiles "..., mais requièrent la présence et l'intervention constante d'un opérateur [1].

B - Manipulateurs automatiques :

À cycles préétablis Leurs déplacements sont limités par des arrêts et des cames réglables à la main. Ils sont contrôlés par des logiques à relais ou pneumatiques (séquences fixes), ou par des automates programmables et des cartes à microprocesseur (séquences variables). En général modulaires, ces appareils sont destinés à une application spécifique [1].

C - Les robots programmables :

Ils sont pilotés par des ordinateurs ou des armoires à commande numérique. Leurs mouvements continus dans l'espace sont alors programmés par un apprentissage ou en langage de synthèse via un clavier, ou sur l'écran d'une station de CAO. Ils permettent d'assurer des manipulations complexes, des opérations de soudure, d'usinage, de découpe, de peinture, de projection, etc... [1].

D - Les robots dits "intelligents :

Équipés de capteurs (par exemple un système de vision artificielle ou un système de suivi de cordon de soudure), ils peuvent ainsi analyser les modifications de leur environnement ou de leur trajectoire et réagir en fonction. Ces machines dites robots de "deuxième génération" ont commencé à être répandues dans l'industrie. La "troisième génération", dotée de capacités de rationnement grâce à l'intelligence artificielle, désormais l'objet de recherches approfondies [1].

1.5. Constituants d'un robot :

Figure 1-4 : Vocabulaire des bras manipulateur

A - Le système mécanique articulé (SMA) :

Il s'agit d'une construction mécanique articulée à multiples degrés de liberté (6 en général pour un robot destiné à manipuler des objets), et dotée de moteurs que l'on appelle aussi actionneurs. Ces derniers peuvent être électriques ou pneumatiques pour les petites charges (jusqu'à quelques déca newtons), ou hydrauliques pour les grandes charges (quelques centaines de newtons). Ils entraînent les articulations du manipulateur par l'intermédiaire de transmissions, elles aussi de divers types (câbles, courroies crantées, engrenages, bielles, etc.).
Pour savoir à chaque instant la position des articulations, on fait appel à des capteurs (potentiomètres, codeurs optiques ou incrémentaux, "résolveurs", etc...) appelés "proprioceptifs" par analogie avec le système humain. Le système mécanique articulé peut être considéré à la fois comme un générateur de déplacements de l'outil dans l'espace géométrique, et comme un producteur de forces.
Cette 1ère entité correspond à ce qui est habituellement montré sur une photographie de robot, et à ce à quoi nous pensons dans la conversation courante lorsque nous parlons d'un robot. Pour notre propos, nous utiliserons indifféremment le terme de manipulateur ou de système mécanique articulé (SMA) [2].

B - L'environnement :

C'est un univers dans lequel la machine est immergée. Pour les robots à poste fixe, il se réduit à ce que l'on rencontre dans l'espace accessible du manipulateur, défini comme l'ensemble des points atteints par l'effecteur final quand le manipulateur passe par toutes les configurations géométriques possibles.
Dans un tel environnement, le manipulateur va croiser des obstacles qu'il doit éviter et des objets d'intérêt, c'est-à-dire sur lesquels il doit agir. Il y a donc une interaction entre le SMA et l'environnement [2].
Des informations sur l'état de l'environnement peuvent être acquises grâce à des capteurs dits " extéroceptifs ", c'est-à-dire permettant de repérer ce qui se trouve à l'extérieur du manipulateur. On utilise des caméras, des détecteurs et capteurs de proximité, des capteurs tactiles, etc... [2].

C - Les tâches à réaliser :

Nous pouvons les définir de manière générale comme la différence entre 2 états de l'environnement : l'état initial d'exécution de la tâche, et l'état final lorsque la tâche est achevée. La principale difficulté est de définir et de modéliser l'environnement. Les tâches sont définies au robot dans un langage approprié. Cette description peut se présenter sous différentes formes, et certains paramètres nécessaires à l'exécution peuvent être acquis pendant l'exécution.
Le support des langages peut être "gestuel" (on montre au robot ce qu'il doit faire par guidage), oral (encore dans le domaine de la recherche et les seules applications connues sont à destination des handicapés moteurs), ou scriptural (on "écrit" dans un langage formalisé) [2].

D - Le contrôleur ou calculateur :

C'est l'organe qui génère les commandes (signaux de puissance des actionneurs), qui vont provoquer les mouvements désirés des différentes articulations du manipulateur, ou les forces appliquées sur les objets ; et ce à partir des informations a priori (connaissance de la tâche à exécuter) et a posteriori (connaissance actuelle et passée du manipulateur et de l'environnement).La commande du robot inclut les fonctions qui lui permettent d'"apprendre" et d'être programmé pour une tâche spécifique, puis d'exécuter cette tâche. La séquence des mouvements, le type de déplacement entre 2 points et l'interaction avec des équipements externes font tous partie de la fonction de contrôle. Pour assurer ses différentes fonctions, le contrôleur doit avoir en mémoire

- un modèle du manipulateur : c'est-à-dire les relations entre les signaux d'excitation des actionneurs et les déplacements du manipulateur qui en découlent ;
- un modèle de l'environnement : qui est une description de ce qui se trouve dans l'espace accessible ;
- des programmes : comprenant les données relatives à la tâche à exécuter ainsi que plusieurs algorithmes de contrôle [2]. Figure 1.5.

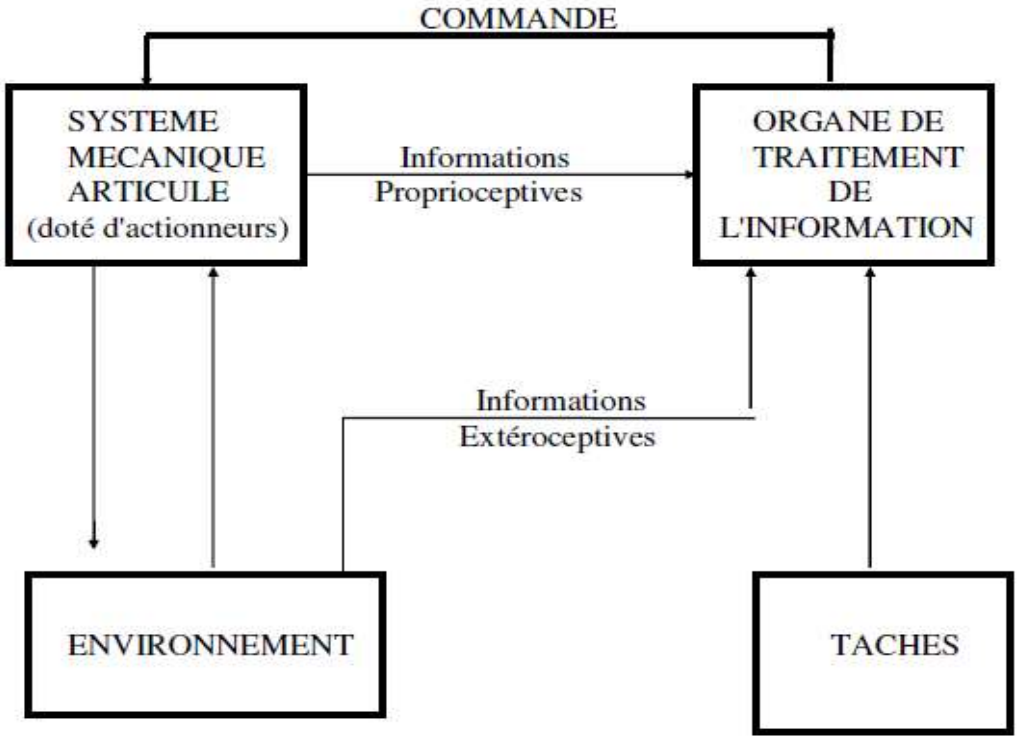


Figure 1-5:Ensembles intervenant dans un robot en fonctionnement

1.6. Caractéristiques d'un robot :

Un robot doit être sélectionné en fonction de son application. Voici donc quelques paramètres à tenir compte, le cas échéant :

- La charge maximale transportable (de quelques kilos à plusieurs tonnes), à déterminer dans les conditions les plus désavantageuses (en élongation maximale).
- L'architecture du S.M.A., le choix est guidé par la tâche à effectuer.
- Le volume actif, défini comme l'ensemble des points qui peuvent être atteints par le dispositif terminal. Tous les déplacements ne sont pas possibles en tous points du volume de travail.
- L'espace de travail (Reachable workspace), également appelé espace de travail maximal, est le volume d'espace que le robot peut atteindre via au moins une orientation.
- Le positionnement absolu, qui correspond à l'erreur entre un point désiré (réel) - défini par une position et une orientation dans l'espace cartésien - et le point atteint et calculé via le MGI du robot .En général, l'erreur absolue de placement, aussi nommée précision, est d'environ 1 mm.
- La répétabilité, ce paramètre caractérisent la capacité qu'a le robot de revenir à un point donné (position, orientation). La répétabilité correspond à l'erreur de positionnement maximale sur un point précis dans le cas de trajectoires répétitives. En général, la répétabilité est de l'ordre de 0,1 mm.

- La vitesse de mouvement (vitesse maximale à l'allongement maximal), l'accélération.
- La masse du robot.
- Le coût du robot.
- La maintenance, ... [2].

1.7. Morphologie d'un robot : [6]

Un robot manipulateur est un ensemble constitué de :

- Une structure mécanique qui soutient l'organe terminal à se repérer.
- Des actionneurs qui servent à agir sur la structure mécanique pour modifier la situation de l'organe terminal.
- Divers capteurs indispensables à la commande.
- Un système de commande qui actionne les actionneurs du robot manipulateur.
- Un système de décision qui garantit la fonction de raisonnement et élabore le mouvement du robot manipulateur.
- Un système de communication qui gère le message transféré entre le système décisionnel et l'opérateur via une console d'affichage.

Pour faire la description de la topologie du mécanisme constituant le robot manipulateur, on associe un graphe dont les sommets sont les corps constitutifs et les arcs les assemblages entre ces corps. Deux corps extrémaux ont des rôles particuliers :

- La base, qui est XXe sur la terre ou sur un véhicule.
- Le corps terminal qui transporte l'outil (ou effecteur).Figure 1.6.

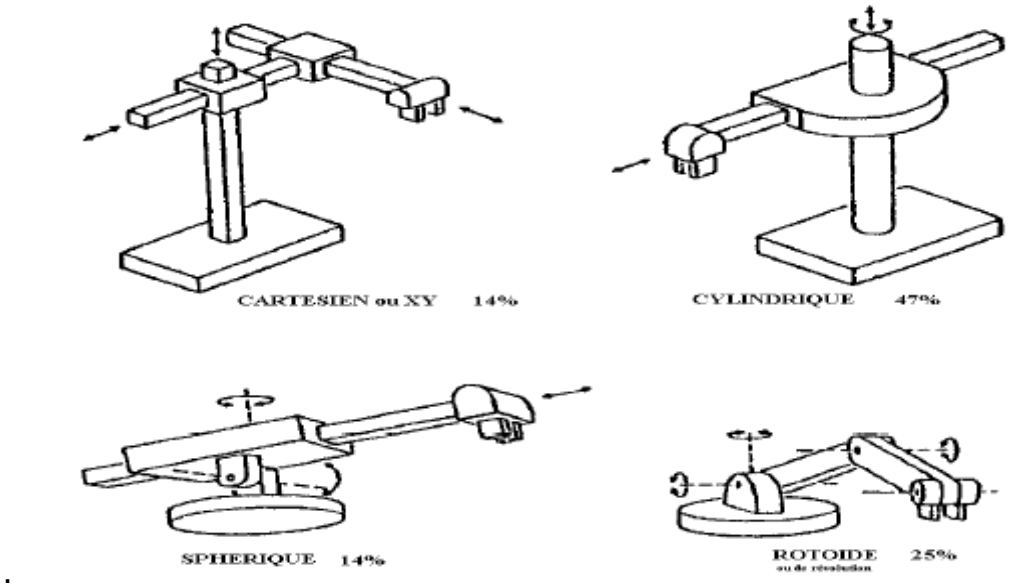


Figure 1-6 : Structure des porteurs les plus utilisés

1.7.1. Les compliments d'un robot :

Il existe différents types d'outils allant du plus simple :

- Vision 2D ou 3D
- Reconnaissance des formes: tracking, contrôle de qualité
- Tracking convoyeur (droit ou circulaire)
- Capteurs d'efforts
- Compliance
- Modification / adaptation de trajectoire
- Gestion d'un 7ème axe
- Augmenter l'enveloppe de travail du robot
- Pince
- Ventouse
- Optimiser les process

1.8. Les générations de robot : [2]

Des progressions ont lieu dans tous les domaines :

- Mécanique,
- Micro-informatique,
- Energétique,
- Capteurs - actionneurs.

On distingue aujourd'hui 3 générations de robots :

1. Le robot est passif : Il est capable de réaliser une tâche compliquée, mais de manière très répétitive, il ne doit pas y avoir de modifications inopportunes de l'environnement. L'auto-adaptabilité est très basse. De nombreux robots sont encore de cette génération.

2. Le robot devient actif : Il est capable d'avoir une image de son environnement, et donc de choisir le meilleur comportement (sachant que les différentes configurations ont été prévues). Le robot peut se calibrer lui-même.

3. Le robot devient "intelligent" : Le robot est capable de définir des stratégies, ce qui fait appel à des capteurs sophistiqués, et bien souvent à l'intelligence artificielle.

1.9. Programmation des robots :[2]

De manière classique, deux étapes sont utilisées pour assurer qu'un robot connaît la tâche à effectuer.

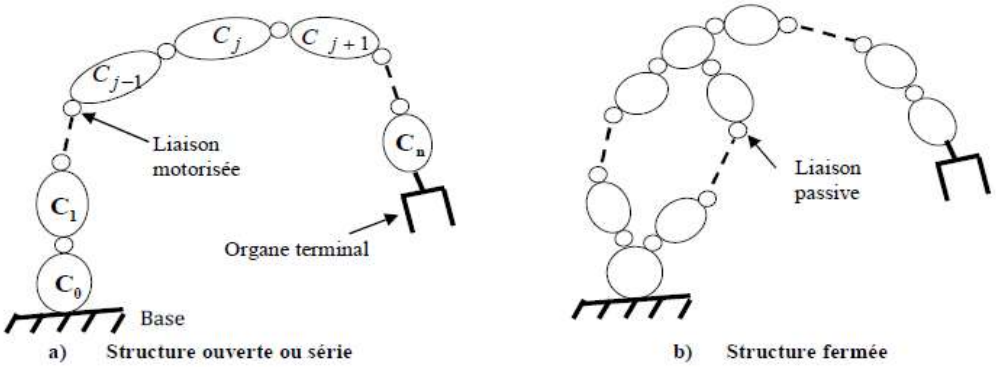
A - Apprentissage :

- Enregistrement dans une mémoire de la trajectoire à effectuer, sous le contrôle d'un opérateur humain,
- Pantalon : Une structure mécanique parfaitement identique à celle du robot, que l'on déplace et qui garde en mémoire les points "pertinents",
- Syntaxeur : Une commande (joystick) permet de contrôler les déplacements de l'organe terminal,
- Boîte à boutons : Un commutateur par actionneur.

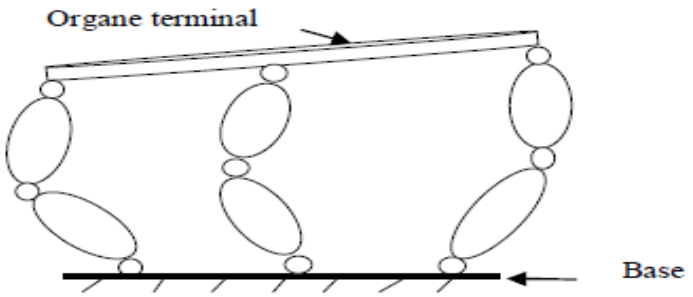
B - La génération des trajectoires :
Et des opérations à exécuter le long de ces trajectoires, qui permettent de définir la tâche à réaliser : On fait appel à un logiciel qui, à partir du modèle du robot, et des trajectoires à effectuer, met au point la succession des commandes des actionneurs.

1.10. Types de chaines : [4].Figure 1-7.

- **Chaine ouverte simple :** aucun retour mécanique des segments.
- **Chaine arborescente :** il existe plusieurs organes terminaux qui agissent en parallèle.
- **Chaine fermées :** il existe un retour mécanique d'un ou plusieurs segments dans la chaine.
  Meilleur équilibrage statique.
  Une grande précision.
  Rigidité élevée.
- **Chaine parallèles :** l'organe terminal est relié a une base mécanique fixé par plusieurs chaines parallèles
  Plus grande précision.
  Plus grande rigidité.
  Capacité de charge élevée.



a) Structure ouverte ou série b) Structure fermée



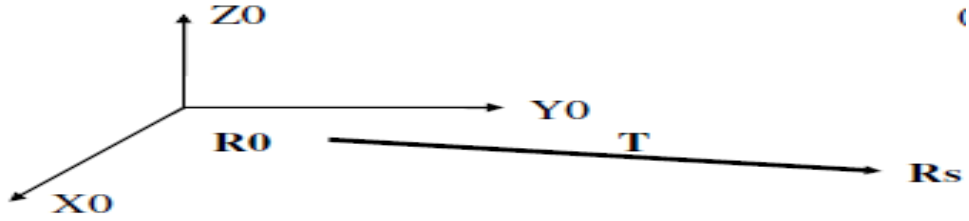
c- Architecture parallèle

Figure 1-7 : Les différents types de chaines des manipulateurs

1.11. Degré de liberté : [1]

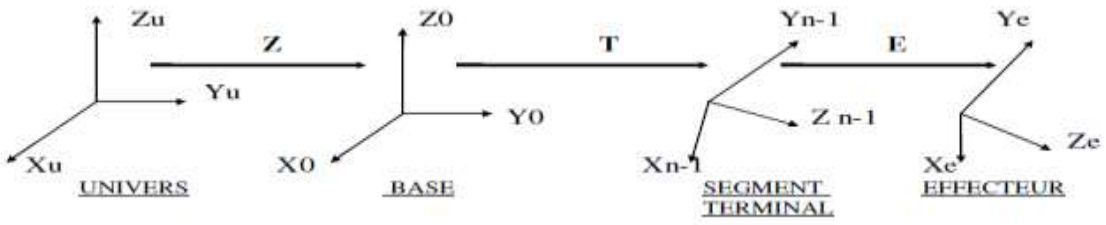
A - Degrés de liberté d'un corps dans l'espace :

Soit Rs un repère fixé à un corps rigide (S). Le nombre de degrés de liberté maximum de ce corps est le nombre de paramètres indépendamment nécessaires pour définir complètement, à tout moment, la transformation géométrique T qui permet le passage du référentiel Ro au référentiel Rs attaché à l'objet.



B - Degrés de liberté du porteur :

Le nombre de d.o.f. d'un mécanisme est le nombre de grandeurs indépendantes qui servent à définir la position du mécanisme à un moment donné du déplacement.
Considérons un manipulateur avec n segments rigides connectés par des articulations. Le nombre de degrés de liberté du porteur est le nombre de paramètres indépendants requis pour exprimer la transformation T du cadre de base (Ro) du manipulateur au cadre du tout dernier segment (Rn-1) sur lequel est fixé l'outil : T : Ro > Rn-1



Ce nombre de degrés de liberté est égal (ou inférieur) au nombre d'articulations autonomes. Le critère (angulaire ou de translation) lié à chaque articulation est appelé variable articulaire.
Pour préciser complètement la situation d'un corps dans l'espace, six paramètres sont nécessaires : 3 coordonnées pour repérer son centre de gravité, et trois angles pour préciser ses rotations autour d'un axe passant par son centre de gravité. Une manipulation totalement générale de ce corps nécessite donc que le manipulateur dispose de six degrés de liberté.

1.12. Mécanismes :[2]

- Un mécanisme est un ensemble de solides reliés 2 par 2 par des liens. Il existe 2 types de mécanismes :
- - Les mécanismes en chaîne ouverte simple (ou en série). Lorsque l'on parcourt le mécanisme, on ne passe jamais deux fois sur le même maillon, ou sur le même solide. Ce type de système est le plus courant.
- - Les mécanismes en chaîne complexe, c'est-à-dire tout ce qui n'est pas en série (au moins un solide avec plus de 2 maillons). Ces systèmes peuvent être divisés en deux groupes : les chaînes arborescentes, et les chaînes fermées (dont l'avantage est qu'elles sont a priori plus rigides, plus précises, capables de supporter de lourdes charges). A titre d'exemple.
- Pour représenter un mécanisme, il existe 2 méthodes :
- Le schéma cinématique8 : On utilise la représentation normalisée des liaisons pour représenter le mécanisme, soit en perspective, soit en projection.
- Le graphe, non normalisé. A titre d'exemple, considérons quelques mécanismes.

1.13. Liaison :

Une liaison entre deux solides indéformables limite la d.d.l. d'un solide par rapport à un autre. Le nombre de paramètres indépendants qui permettent de définir la localisation (position et orientation) d'un solide par rapport à l'autre dans tout mouvement (compatible avec la liaison) est appelé d.d.l. de la liaison.

Dans l'assemblage des robots manipulateurs, les liaisons les plus fréquentes sont :

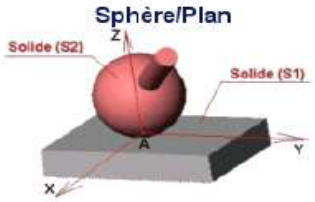
- La liaison rotoïde (ou pivot).
- La liaison prismatique (ou glissière).
- La liaison rotule S (sphérique).
- La liaison cardan U (joint universel).

Certaines liaisons sont motorisées, on parlera de liaisons actives. Ce sera généralement le casdes liaisons rotoïde ou prismatique. Les liaisons non motorisées seront appelées liaisons passives.

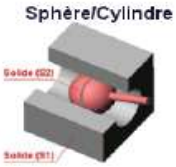
Exemples **:**

- Un objet cubique sur un plan a 3 d.d.l. 2 pour déterminer les coordonnées du point dans le plan, 1 pour définir son orientation dans le plan.
- Une sphère sur un plan a 5 d.o.b. 2 pour fixer les coordonnées d'un point du plan, 3 pour en déterminer l'orientation dans le plan.
- Une porte par rapport à la paroi a 1 d.o.b. [2].

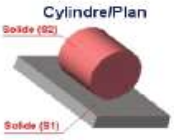
1.13. 1 Les différents contacts :



contact ponctuel

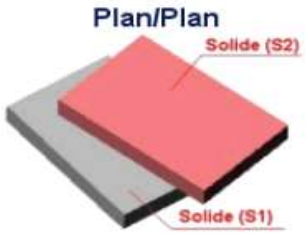


contact linéique



contact linéique

contact surfacique



contact surfacique

1.13.2. Indice de mobilité :[5]

L'indice de mobilité M est égal au nombre de paramètres variables qui déterminent la configuration du manipulateur.

M = n

Si :

- La chaîne cinématique est simple (chaque joint) a, au maximum, un suiveur et un prédécesseur).
- chaque joint est de classe 5.

En général, le degré de liberté du robot (DLr) est égal à M à moins que le robot soit redondant. Dans tous les cas :
DLr <= M

1.13.3. Redondance : [5]

Le nombre de degrés de liberté de l'organe terminal est inférieur au nombre de variables articulaires actives (des articulations motorisées).

- plus de six articulations.
- plus de 3 articulations rotoïdes à axes concourants.
- plus de 3 articulations rotoïdes à axes parallèles.
- plus de 3 articulations prismatiques.
- deux articulations prismatiques à axes parallèles.
- deux joints rotoïdes avec des axes coïncidents.

1.13.4. Configuration singulière : [5]

Quel que soit le type de robot (redondant ou non), il est possible qu'il y ait des configurations dites singulières comme le nombre de degrés de liberté de l'embout est plus petit que la dimension de l'espace opérationnel (espace dans lequel on représente le ddl de l'OT).

- deux axes d'articulations prismatiques sont parallèles.
- deux axes d'articulations rotoïdes sont confondus.

1.13.5. Répétabilité : [8]

Selon la norme ISO 9328, la répétabilité est la reproductibilité de l'orientation et de la position du dispositif terminal du robot à l'aide de la même commande (Figure 1.8). La répétabilité des robots industriels diffère selon leur taille, leur charge maximale, leur structure et la qualité de fabrication de leurs composants. Elle est de l'ordre de 0,03 mm à 0,1 mm pour les robots de petite et moyenne taille. Pour les grands manipulateurs, elle peut être supérieure à 0,2 mm. Plusieurs facteurs peuvent affecter la répétabilité par des phénomènes aléatoires tels que le frottement, le jeu, la non-linéarité du modèle géométrique et les réglages des servomoteurs. Il est presque impossible de modéliser et d'anticiper ces facteurs. Figure 1-8.

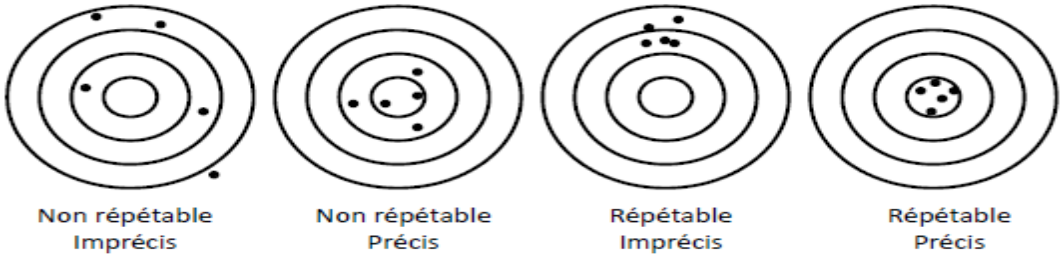


Figure 1-8 : Précision et répétabilité du robot

1.13.6. Rigidité du robot : [9]

L'utilisation des robots anthropomorphes dans l'usinage du bois par enlèvement de matière se heurte au manque de rigidité du système. Celui-ci provient de plusieurs sources telles que : les systèmes de transmissions, les jeux mécaniques, les corps du robot et la commande des actionneurs. Abele et al ont montré dans que la flexibilité de l'ensemble bras, épaule et poignet est infinitésimal par rapport à celle des articulations. Face à ces contraintes, le choix de la configuration du notre robot pour les campagnes expérimentales est basé sur les travaux menés par Doukas et al et Dumas et al. Claire Dumas a établi une cartographie de dextérité d'un robot ®Kuka série Kr (Figure 2.10) en utilisant le nombre de conditions de la Jacobienne. Elle a démontré les endroits de l'espace de travail où le comportement du robot est dextre (couleur claire) et les zones où la manipulabilité du robot est faible (couleur foncée)figure 1-9.

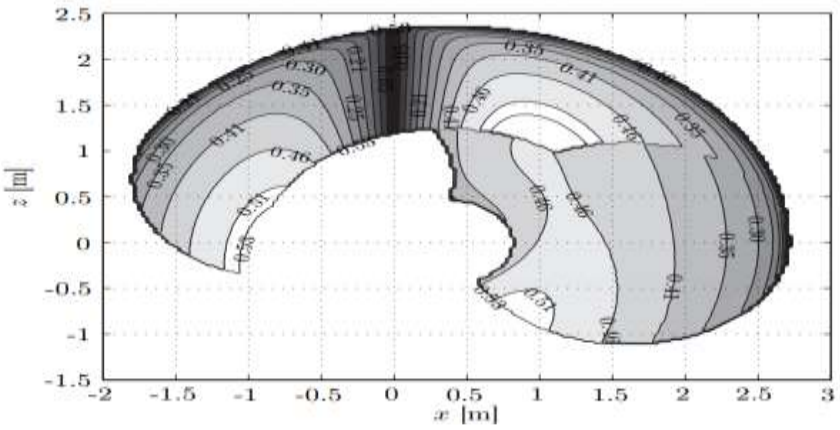


Figure 1-9 **:** Cartographie de dextérité du Kuka série Kr

1.14. Domaines d'étude :

Un grand nombre de disciplines scientifiques sont touchées par la mise en œuvre de robots :

- L'automatique : calibration des capteurs, effecteurs ...
- Electronique : mise en place de constituants pour les robots.
- Informatique : réalisation de programmes pour les robots.
- Mathématiques : modèles pour la prise de décision et/ou l'apprentissage, calcul de trajectoire, localisation…
- Sciences cognitives : interactions homme-machine, machine-machine, prise de décision…
- Physique : cinématique des robots, navigation, ...

1.15. Intelligence Artificielle :

L'intelligence Artificielle d'un robot se résume a un ensemble de programmes préalablement écrits avec un ordinateur :

- les programmes s'exécutant sur les robots sont écrits avec un langage de programmation (exemples : C++, Java, ...),
- ils s'exécutent grâce au contrôleur et a la mémoire du robot,
- ils prennent en entrée les informations obtenues par les capteurs et en sortie envoient des ordres aux effecteurs.

1.15.1. Les enjeux de l'IA dans la robotique : [14]

"L'intelligence artificielle pourrait être le plus important événement de l'Histoire de notre civilisation "Stephen Hawking", cette phrase interpelle et soulève son partie d'espoirs et d'inquiétudes. L'IA est au cœur de polémiques technologiques, économiques et sociales. Encore récemment médiatisée au travers des rapports Villani et de l'académie des technologies, l'IA est sans aucun doute la thématique numérique la plus en vue car elle s'apprête à modifier en profondeur de nombreux domaines d'activités.
Pour le grand public, IA et Robotique sont intimement liées. Les robots sont ainsi devenus l'allégorie idéale de l'IA. Ils donnent l'impression de devenir "intelligents" dès que leur concepteur intègre des algorithmes de haut niveau. S'intéresser à l'IA et aux robots c'est comprendre les enjeux et les fantasmes qu'ils véhiculent, du travail à la santé en passant par l'économie et la religion.

1.16. La robotique dans l'industrie mondiale : [7]

Entre 2005 et 2008, le nombre annuel de robots vendus a évolué pour atteindre environ 115 000 unités. L'année 2009 est exclue en raison de la crise économique et financière mondiale. Celle-ci a provoqué une chute exceptionnelle des ventes, seuls 60 000 robots sont vendus dans le monde, et de nombreux projets d'investissements industriels ont été reportés.

Depuis 2010, la demande de robots industriels s'accélère en raison de la tendance à l'automatisation des lignes de production ainsi que des multiples bénéfices qu'offre la robotisation des tâches industrielles. En 2019, le nombre total de robots industriels commercialisés a atteint 420 870 unités (le même nombre a été enregistré en 2018). Les pronostics prévoient qu'entre 2020 et 2022, ce nombre progressera de 12 % chaque année figure 1.10.

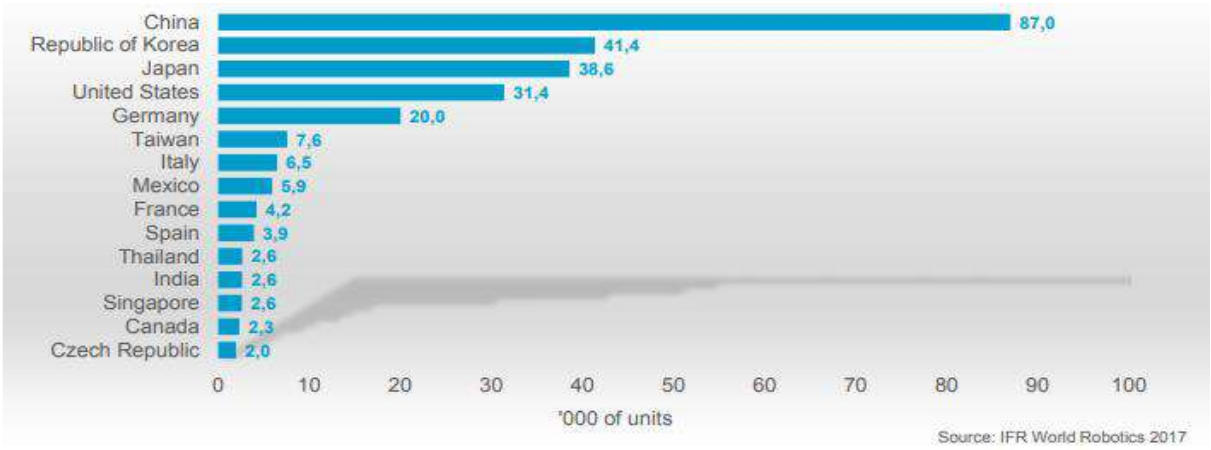


Figure 1-10 **:** Estimation du stock annuel de robots industriels par industries en 2017 (Source : IFR World Robotics)

1.17. Pourquoi robotisé :

L'intégration des robots collaboratifs dans des chaînes de production possède de multiples avantages :

- C'est un outil informatique qui permet de gérer l'ensemble des processus opérationnels d'une entreprise.
- C'est un progiciel capable de structurer, organiser et gérer l'ensemble des services au sein d'une entreprise (gestion des ressources humaines, financière, vente, approvisionnement...).
- C'est un système de supervision industrielle capable de mettre en place une traçabilité complète des informations de production.
- 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
- Les robots industriels ont la capacité de reproduire une même tâche répétitive sans dégradation des performances.
- Augmentation de la sécurité sur le poste de travail.
- Réduction des coûts énergétiques.

1. Conclusion :

Le robot industriel est l'équipement de production flexible par excellence, il est désormais performant, relativement peu coûteux et fiable. Grâce à son efficacité et aux performances techniques dont elle a fait preuve, la robotique conquiert de nouveaux secteurs et occupe un rôle central dans leurs progressions et développement tel que la métallurgie, plasturgie, agro-alimentaire, pharmaceutique et bien d'autres.

CHAPITRE 2 : Modélisation mathématique pour la commande des robots

2. Introduction :

La conception et la commande des robots sont des systèmes qui nécessitent le calcul des modèles mathématiques, afin de le représenter, tels que :

- Les modèles géométriques directs et inverses.
- Les modèles cinématiques directs et inverses.
- Les modèles dynamiques.

On va présenter dans ce deuxième chapitre quelques méthodes mathématiques qui nous permettront d'établir ces modèles.

2.1. Modèle géométrique du robot MG :

La conception et le pilotage des robots exigent le calcul de certains modèles mathématiques, tels que le MGD, qui expriment la situation de l'organe terminal en fonction des variables articulaires du mécanisme et vice versa.
La modélisation des robots de façon systématique et automatique nécessite une méthode appropriée pour la description de leur morphologie. Différentes méthodes et notations ont été proposées. Pour garantir la qualité de la tâche, la précision de la pose de l'effecteur final doit être contrôlée. Un modèle géométrique conforme au comportement géométrique de la structure du robot doit être développé. Le modèle le plus utilisé est la méthode de Denavit-Hartenberg. Mais cette dernière, développée pour des structures ouvertes simples, présente des incertitudes lorsqu'elle est appliquée à des robots dont la structure est fermée ou arborescente. Pour cette raison, nous conseillons la notation de Khalil et Kleinfinger qui permet de décrire de manière homogène, avec un nombre minimal de paramètres, des architectures ouvertes simples et complexes de systèmes mécaniques articulés [10].

2.2. *Modèle de Denavit-Hartenberg DH :*

Denavit et Hartenterg ont proposé une méthodologie pour décrire la transformation homogène entre 2 solides contigus avec une position et une orientation particulière des repères.

2.2.1. Principe :

- Fixer des repères à chaque corps du robot.
- Calculer les matrices homogènes entre chaque corps.
- Calculer cette matrice entre base et organe terminal.

2.2.2. Hypothèse et les paramètres de DH :[10]

Une simple structure ouverte est constituée de n+1 corps notés C0, ..., Cn et de n articulations. Le corps C0 désigne la base du robot et le corps Cn le corps qui porte l'organe terminal. L'articulation j relie le corps Cj au corps Cj-1.
La méthode de la description est basée sur les règles et conventions suivantes :
- les corps sont censés être parfaitement rigides. Ils sont reliés par des articulations jugées idéales (aucun jeu mécanique, aucune élasticité), soit rotoïdes, soit prismatiques ;

- le repère Rj est lié au corps Cj ;
- l'axe zj est mené 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 xj n'est pas unique.

Le passage du cadre Rj-1 au cadre Rj s'exprime en fonction des 4 paramètres géométriques suivants :

- aj : angle entre les axes zj-1 et zj qui correspond à une rotation autour de xj-1.
- dj : distance entre les axes zj-1 et zj le long de xj-1.
- qj : angle entre des axes xj-1 et xj correspondant à une rotation autour de zj.
- rj : distance entre les axes xj-1 et xj le long de zj.Figure 2-1

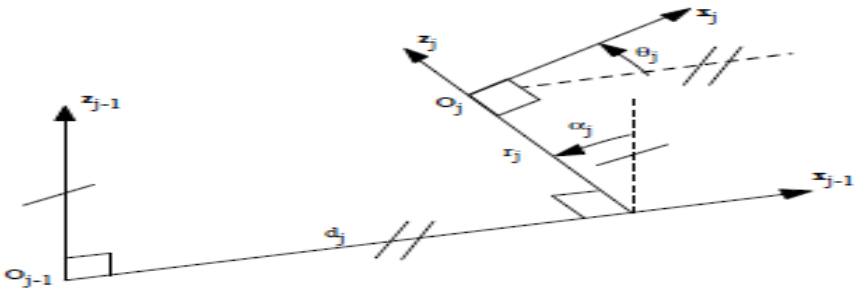


Figure 2-1 **:** *Paramètres géométriques dans le cas d'une structure ouverte simple*

La variable articulaire qj associée à la jème articulation est soit qj, soit rj, selon que cette articulation est detype rotoïde ou prismatique, ce qui se traduit par la relation :

Avec :$q_j = s - jq_j + s_j r_j$ équation 2.1

• sj = 0 si l'articulation j est rotoïde ;
• sj = 1 si l'articulation j est prismatique ;
• s -j = 1 – sj équation 2.2

La matrice de transformation définissant le repère Rj dans le repère Rj-1 est :

$j-1Tj = Rot(x,aj)\ Trans(x,dj)\ Rot(z,qj)\ Trans(z,rj)$ équation 2.3

$$= \begin{matrix} C\theta j & -S\theta j & 0 & dj \\ C\alpha jS\theta j & C\alpha jC\theta j & -S\alpha j & -rjS\alpha j \\ S\alpha jS\theta j & S\alpha jC\theta j & C\alpha j & rjC\alpha j \\ 0 & 0 & 0 & 1 \end{matrix}$$

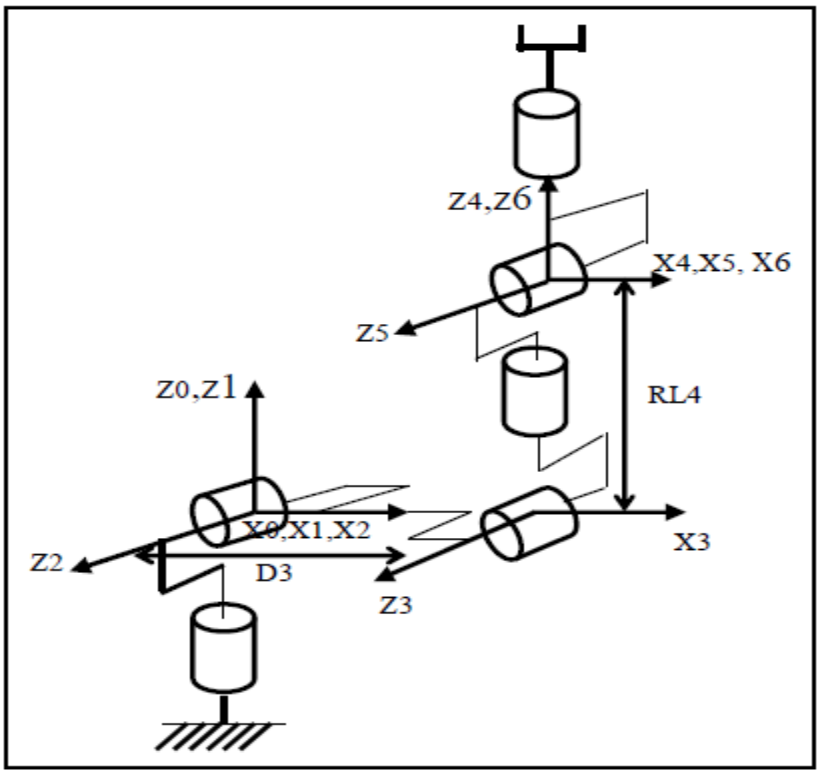


Figure 2-2 **:** *Placement des repères et notations pour le robot Stäubli RX-90*

j	σ j	α j	dj	θ j	rj
1	0	0	0	θ1	0
2	0	π / 2	0	θ2	0
3	0	0	D3	θ3	0
4	0	- π / 2	0	θ4	Rl4
5	0	π / 2	0	θ5	0
6	0	- π / 2	0	θ6	0

Tableau 2.1 : paramètres géométriques du robot staubil RX-90

2.3. Modèle géométrique direct MGD :

Le MGD est un ensemble de relations qui permettent d'exprimer la situation de l'organe terminal, c'est-à-dire les coordonnées opérationnelles du robot, en fonction de ses coordonnées articulaires. Dans le contexte d'une chaîne ouverte simple, il peut être exprimé par la matrice de transformation 0Tn :

$0\mathbf{T}n = 0\mathbf{T}1\ (q1)\ 1\mathbf{T}2\ (q2) \ldots n\text{-}1\mathbf{T}n\ (qn)$ équation 2.4

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

$X = f(q)$ équation 2.5

q étant le vecteur des variables articulaires tel que :

$q = [q1\ q2 \ldots qn]T$ équation 2.6

Les coordonnées opérationnelles sont définies par :

$X = [x1\ x2 \ldots xm]T$ équation 2.7

Plusieurs possibilités existent pour la définition du vecteur **X**. Par exemple, avec les éléments de la matrice0**T**n :

$X = [px\ \ py\ \ pz\ sx\ sy\ sz\ nx\ ny\ nz\ \ ax\ ay\ \ az]T$ équation 2.8

Ou bien, sachant que $\mathbf{s} = \mathbf{n} \times \mathbf{a}$:

$\boldsymbol{X = [px\ \ py\ pz\ nx\ ny\ nz\ ax\ \ ay\ az]T}$ équation 2.9

2.3.1. Représentation d'un point dans l'espace :[15]

Soit P un point de coordonnées cartésiennes Px,Py,Pz (fig.2.3). On appelle coordonnées homogènes du point P, les termes W × Px, W × Py, W× Pz, ou 'W' est un facteur d'échelle égale à 1 en robotique. On présente alors les coordonnées homogènes d'un point parle vecteur : Figure 2-3

$$, \quad P= \begin{matrix} px \\ py \\ pz \\ 1 \end{matrix}$$

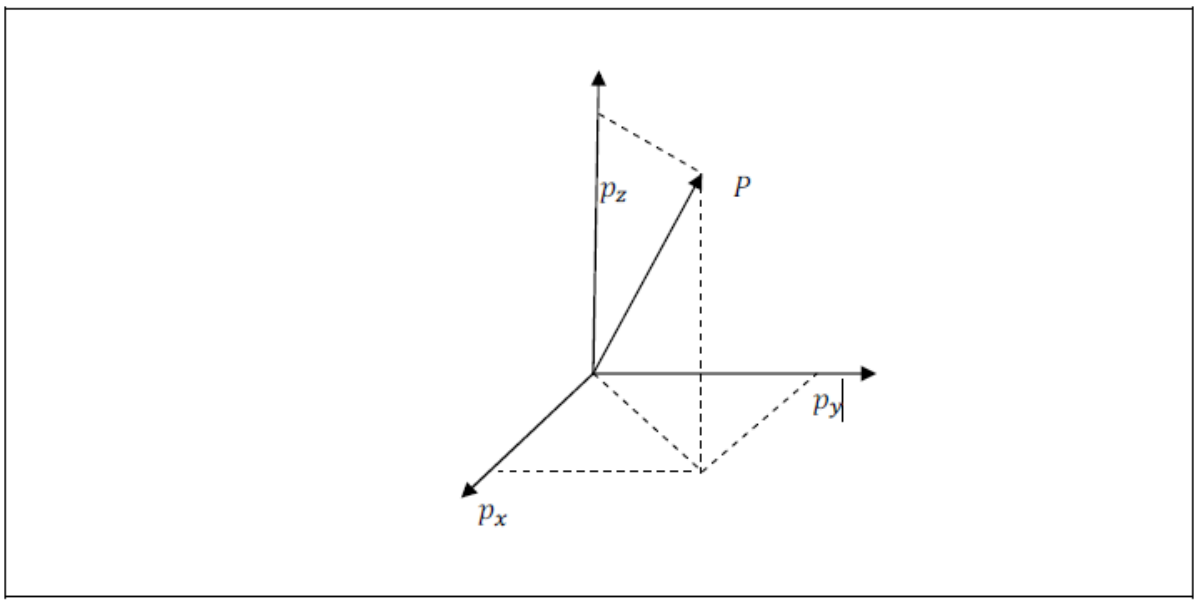


Figure 2-3 **:** Représentation d'un point dans l'espace.

2.3.2. Représentation d'une direction : [15]

La représentation d'une direction (vecteur libre) se fait aussi par quatre composantes, la quatrième étant nulle, indique un point à l'infini. Si l'on note Ux, Uy, Uz, les coordonnées cartésiennes d'un vecteur unitaire u, en coordonnées homogènes on écrit :

$$U = \begin{matrix} Ux \\ Uy \\ Uz \\ 0 \end{matrix}$$

2.3.3. Matrice de translation :[15]

Soit de repère Ro(Oo,xo,yo,zo) ; Ri(Oi,xi,yi,zi) et soient Px, Py, Pz les coordonnées de l'origine Oi du repère Ri dans le repère Ro. Lorsque les axes des deux repère Ro et Ri sont en parallèle, alors la transformation de A vers B est appelée ''matrice de translation'' et donnée par :

$$A = Trans(px, py, pz) = \begin{matrix} 1 & 0 & 0 & px \\ 0 & 1 & 0 & py \\ 0 & 0 & 1 & pz \\ 0 & 0 & 0 & 1 \end{matrix}$$

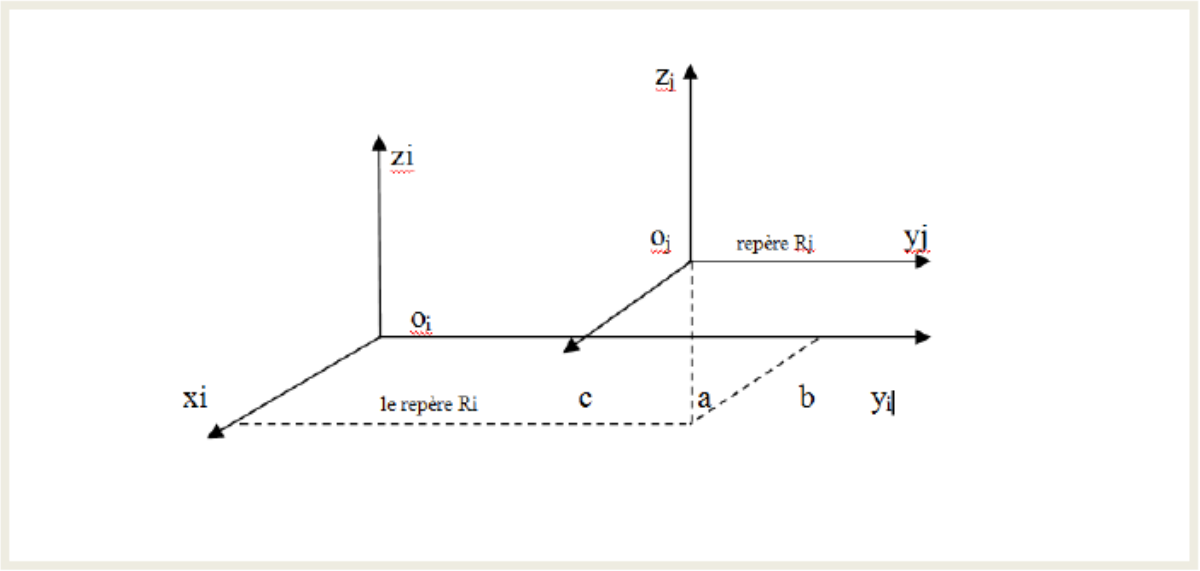


Figure 2-4 : translation pure

2.3.4. Matrice de rotation :

Soient deux repères Ro et Ri précédemment définis, lorsque leurs origines coïncident Oo = Oi et que le repère Ri effectue une rotation d'un angle O autour d'un ou plusieurs axes de R0 alors cette rotation peut être décomposée en rotations élémentaires de différents angles autour des axes de Ro, la matrice de transformation est obtenue par le produit de ces trois matrices de rotation :

- Rotation d'un angle $\theta 1$ autour de $x0$:

$$Rot(xo,\theta 1) = \begin{matrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta\, 1 & \cos\theta 1 & 0 \\ 0 & 0 & 0 & 1 \end{matrix}$$

- Rotation d'un angle $\theta 2$ autour de yo :

$$Rot(yo,\theta 2) = \begin{matrix} \cos\theta 2 & 0 & \sin\theta 2 & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta 2 & 0\cos & \theta 2 & 0 \\ 0 & 0 & 0 & 1 \end{matrix}$$

- Rotation d'un angle $\theta 3$ autour de zo :

$$Rot(zo, \theta 3) = \begin{matrix} \cos\theta 3 & -\sin\theta 3 & 0 & 0 \\ \sin\theta 3 & \cos\theta 3 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{matrix}$$

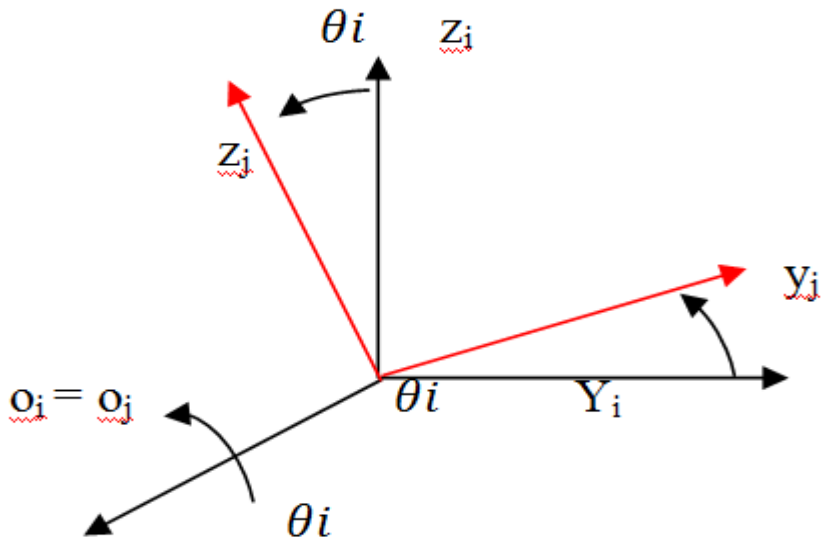


Figure 2-5 : rotation de 1 autour de l'axe **x**

2.4. Modèle géométrique inverse MGI :[11]

MGI : la ou les configurations correspondant à une situation OT donnée :

f-1 : M - ! N x 7- ! q = f-1(x) équation 2.10

Existence d'un nombre fini de solutions :

- Si n < m : pas du tout de solution.
- Si n = m : nombre fini de solutions.
- Si n > m : nombre illimité de solutions.

Résolution de la MGI :

Pas de méthode théorique systématique pour calculer la MGI.

Le mieux est de prendre les équations de la MGD et d'effectuer le calcul en sens inverse étape par étape. Dans le cas où n = 6, l'existence d'un poignet sphérique permet de commencer la résolution par :

$$px = x1 - anxx - rn + 1zx \quad \text{équation 2.11}$$

$$py = x2 - anxy - rn + 1zy \quad \text{équation 2.12}$$

$$pz = x3 - anxz - rn + 1zz \quad \text{équation 2.13}$$

Puis résolution au cas par cas pour exprimer les qi, pour i = 1, 2, ..., n en fonction de px, py, pz et des cosinus directeurs.

2.5. Modèle cinématique direct MCD : [12].

Le modèle cinématique direct d'un robot -manipulateur décrit les vitesses des coordonnées opérationnelles en fonction des vitesses des articulations. On note :

$$\boldsymbol{X = J(q)q} \quad \text{équation 2.14}$$

Où J(q) désigne la matrice jacobienne de dimension (m×n) du mécanisme, égale à

$$\frac{\boldsymbol{\sigma x}}{\boldsymbol{\sigma q}}$$

et fonction de la configuration articulaire q. La même matrice jacobienne est utilisée 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, c'est-à-dire :

$$\boldsymbol{dx = J(q)dq} \quad \text{équation 2.15}$$

2.5.1. Jacobien cinématique : [12]

Le mouvement de l'organe terminal peut également être défini par sa vitesse instantanée sous la forme d'un torseur cinématique. Cette partie propose d'établir la relation entre ce torseur et les positions et vitesses articulaires q et q°.

2.5.2 Intérêt de la matrice jacobienne : [12]

- Elle est la base du modèle inverse différentiel. Elle permet de calculer une solution locale des variables articulaires q connaissant les coordonnées opérationnelles X.
- En statique, le jacobien permet d'établir la relation entre les forces exercées par l'embout sur l'environnement et les forces et couples des actionneurs. Il permet de calculer plus facilement les singularités et la dimension de l'espace opérationnel accessible du robot.

2.6. Modèle cinématique inverse : [12]

L'objectif du modèle cinématique inverse est de calculer les vitesses articulaires q qui assurent au référentiel terminal une vitesse optimale X imposée, à partir d'une configuration q donnée.

Cette définition est analogue à celle du modèle différentiel inverse : ce dernier permet de déterminer la différentielle articulaire dq correspondant à une différentielle spécifiée des coordonnées opérationnelles dX. Pour obtenir le modèle cinématique inverse, le modèle cinématique direct est inversé par la résolution d'un système d'équations linéaires ; la mise en œuvre peut se faire analytiquement ou numériquement ;

- la solution analytique a l'avantage de réduire considérablement le nombre d'opérations, mais on doit traiter tous les cas singuliers
- Les méthodes numériques sont plus générales, la plus répandue étant basée sur la notion de pseudo-inverse : les algorithmes traitent de manière unifiée les cas réguliers, singuliers et redondants ; ils demandent un temps de calcul relativement important.

Dans le cas ordinaire, la matrice jacobienne est carrée d'ordre n et son déterminant est non nul. Deux méthodes de calcul peuvent être mises en œuvre :

J-1 est calculée comme la matrice inverse de J, ce qui permet de déterminer les vitesses articulaires q grâce à la relation :

$$\boldsymbol{q = J - 1X} \qquad \text{équation 2.16}$$

$$\mathbf{J} = \begin{bmatrix} \boldsymbol{A} & \mathbf{0} \\ \boldsymbol{B} & \boldsymbol{C} \end{bmatrix}$$

Lorsque la matrice J est de la forme :

Les matrices A et C étant carré et inversibles, il est facile de monter que l'inverse de cette matrice s'écrit :

$$J^{-1}= \begin{matrix} A^{-1} & 0 \\ -C^{-1}BA^{-1} & C^{-1} \end{matrix}$$

La résolution de ce problème se ramène donc à l'inversion,

2.7. Modèle dynamique MD : [12]

Le modèle dynamique est la relation entre les couples (et/ou forces) appliqués aux actionneurs et les positions, vitesses et accélérations articulaires. On représente le modèle dynamique par une relation de la forme :

$t = f(qx, qy, qz, fe)$

Avec :

- Γ : Vecteur des couples/forces des actionneurs, selon que l'articulation est rotoïde ou prismatique ;
- qx : vecteur des positions articulaires ;
- qy : vecteur des vitesses articulaires ;
- qz : vecteurs des accélérations articulaires
- fe : vecteur représentant l'effort extérieur (force et moments) qu'exerce le robot sur l'environnement

Parmi les applications du modèle dynamique, on peut citer :

- la simulation, qui utilise le modèle dynamique direct
- le dimensionnement des actionneurs
- l'identification des paramètres inertiels et des paramètres de frottements
- la commande, qui utilise le modèle dynamique inverse.

2.7.1. Modèle dynamique direct MDD : [12]

Le modèle dynamique direct est celui qui exprime les accélérations articulaires en fonction des positions, vitesses et couples des articulations.il est alors représentés par les relations :

$$q = g(q, q, t, fe)$$

2.7.2. Modèle dynamique inverse MDI : [12]

On appelle modèle dynamique inverse, ou tout simplement modèle dynamique, la relation de la forme :

$$t = f(qx, qy, qz, fe)$$

2.7.3Formalisme de Lagrange : [13]

Nous considérons un système idéal sans frottement ni aucune élasticité, qui n'exerce ni forces ni moments sur l'environnement. La formulation de Lagrange décrit le fonctionnement d'un système dynamique en termes d'énergie.

Les équations de Lagrange sont généralement écrites sous la forme :

$\tau = \frac{d}{dt}\frac{\theta L}{\theta \dot{q}} - \frac{\theta L}{\theta q}$ équation 2.17

Ou 'L est le Lagrangien du système, se définit comme la différence entre l'énergie cinétique K et l'énergie potentiel P :
$L = K - P$ équation 2.18

2.7.4. Formalisme de Newton-Euler : [12]

Les équations de Newton-Euler expriment le torseur dynamique en Gj des efforts extérieurs sur un corps j par les équations

$F = MVGj$ équation 2.19

$MGj = IGj + \omega \times (IGj\ \omega j$ équation 2.20

2. Conclusion :

Ce chapitre englobe une rétrospective des notions de base dans le domaine de robotique. Il est consacré à la modélisation des robots manipulateurs, _à partir de la modélisation géométrique, cinématique, et dynamique où le formalisme de Lagrange et Newton-Euler est représenté.
Nous disposons dés-lors d'un ensemble de modèles mathématiques pouvant être mis en œuvre après avoir procédé à une identification des paramètres géométriques et inertiels.

CHAPITRE 3 : Programmation des taches avec RobotStudio

3. Introduction :

Dans le processus continu, le robot est piloté par une série de commandes et de fonctions stockées dans un fichier appelé "programme du robot". La trajectoire du robot, et par suite de la torche, doit être programmée avant son exécution. Deux méthodes de planification des trajectoires des robots sont généralement utilisées : la PEL et la PHL. La principale méthode de programmation en ligne est la programmation par apprentissage. Il existe deux types de programmation hors ligne : la programmation par langage et la programmation graphique à partir d'un système de CAO/FAO (conception et fabrication assistées par ordinateur).[18]

3.1. Plate-forme de programmation du robot :

Notre cellule de robotique possède deux moyens de programmer le robot :

- La programmation en ligne
- Programmation hors ligne

3.1.1 Programmation en ligne : [16]

Elle consiste à générer un programme " au pied de la machine " par le biais d'une interface homme-machine appelée " Teach Pendant " (Figure 3-1). Il s'agit de l'interface homme-machine privilégiée d'un robot. Il s'agit de l'unité de commande d'un robot. Elle permet aux programmeurs de créer des trajectoires de mouvement en utilisant différentes méthodes (linéaire, circulaire, point par point). Le Teach Pendant permet également aux programmeurs de gérer les entrées/sorties pour la manipulation des dispositifs qui travaillent en collaboration avec le robot (caméra intelligente, broche d'usinage, pince de manipulation, ventouse, etc.) (Figure 3-2).

Figure 3-1 **:** Teach pendant du robot kuka

Figure 3-2 : Pince + ventouse montées sur un robot Fanuc

La programmation en ligne requiert que le robot soit totalement disponible. Il est nécessaire de bloquer la production pour permettre une manipulation en toute sécurité. Avec cette approche, il est nécessaire de définir manuellement tous les points caractéristiques de la trajectoire à suivre (points de départ et d'arrivée, zones de retrait et de sécurité, évitement des singularités et des obstacles, etc.)
Cette méthode a l'avantage d'être exacte au niveau de la trajectoire de déplacement et de permettre l'anticipation de tous les cas problématiques : collision, singularité, lissage des trajectoires, apprentissage de l'environnement et estimation du temps de cycle. Son principal inconvénient est son coût de mise en œuvre en termes de temps de programmation et donc d'arrêt de production.

3.1.2. L'apprentissage direct : Figure3-3

Le principe de la méthode de programmation réside dans la mémorisation, au cours du processus d'apprentissage, de toutes les configurations articulaires du robot lors de l'exécution de la tâche ou lors d'une simulation de celle-ci. L'opérateur manipule le robot directement à l'aide d'une console d'apprentissage pour lui "apprendre" la tâche à réaliser. L'acquisition de la trajectoire est continue. Ce type de programmation est simple et nécessite peu de formation pour le programmeur. Cette méthode est plutôt utilisée pour des applications qui nécessitent des mouvements de précision limitée, comme la pulvérisation sur des pièces simples ou la peinture. [17]

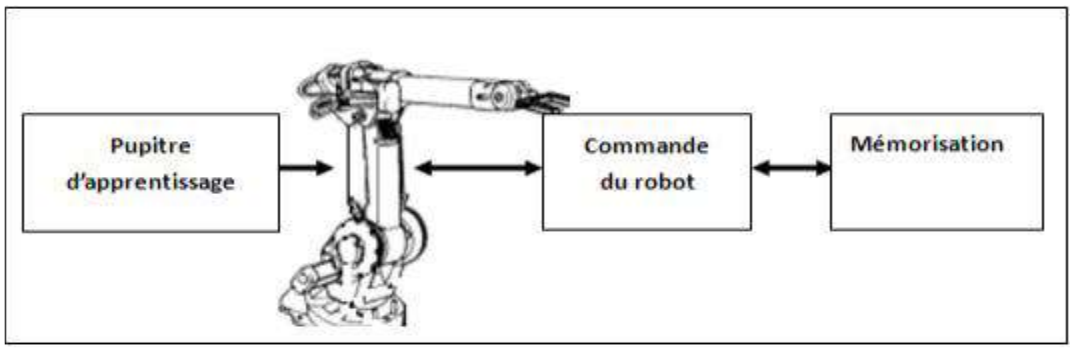


Figure 3-3 **:** Programmation par apprentissage

3.1.3 L'apprentissage indirect point par point :

Le principe de cette méthode est également basé sur une démonstration physique de la tâche à effectuer. L'opérateur déplace le robot en mode manuel, à l'aide du panneau de commande à écran tactile. Cependant, la configuration du robot n'est enregistrée qu'en quelques points caractéristiques de la trajectoire. Le lien entre ces points est spécifié par les caractéristiques de la trajectoire entre ces points (linéaire, articulaire ou encore circulaire) et par la définition d'une vitesse de déplacement spécifiée par l'opérateur. [17]

3.1.4. Programmation hors ligne :

Cette démarche permet de générer des programmes sur un robot à partir d'un environnement de simulation. La plateforme de simulation est dans ce cas hébergée sur un PC distant et nécessite une modélisation fidèle de la cellule du robot. Chaque fabricant de robots propose sa propre plate-forme de simulation. Cette méthode a pour avantage de permettre aux programmeurs d'anticiper les difficultés liées à la génération des trajectoires (obstacles, singularités, régularité, vitesse et accélération, torsion des câbles, optimisation du temps de cycle...) sans avoir à arrêter la production.
La PHL correspond à la génération d'un programme de robot à exécuter sur site à partir de données CAO générées avec un logiciel informatique, notamment le logiciel virtuel RobotStudio équipé d'une bibliothèque de robots.
Ce type de programmation devient une tendance pour de nombreuses applications robotiques, dont l'interface est illustrée à la figure 3-3.

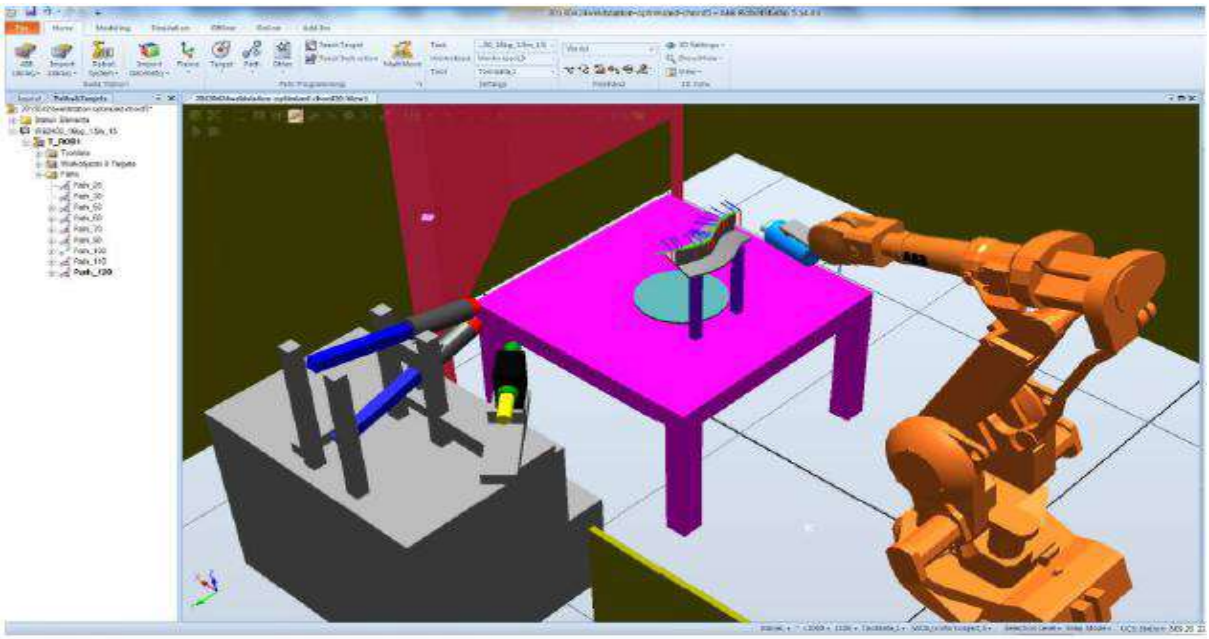


Figure 3-4 : Interface du logiciel RobotStudio

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]

3.1.5. 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. [17]
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 exemple : staubli (langage *VAL3*), KUKA (langage *KRL*), ABB (langage *RAPID*). Dans notre étude nous allons utiliser le langage RAPID.
Donc la programmation par apprentissage qui fait référence à un langage «gestuel», la programmation par langage est basée sur un langage «textuel». [18]

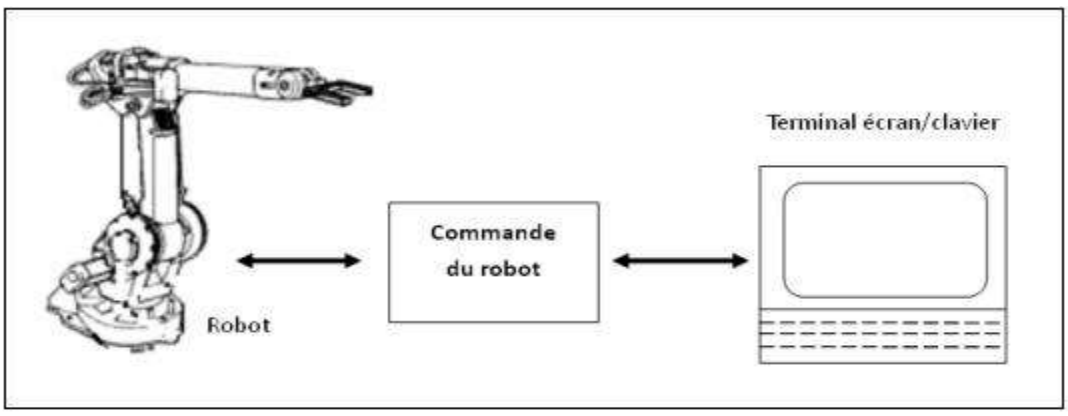


Figure 3-5 **:** Programmation par langage

3.1.6. La programmation graphique :

Cette méthode de programmation est une méthode plus pratique pour programmer des trajectoires robot sur les pièces de forme complexe. Les logiciels de CFAO actuels permettent de représenter et de modéliser les robots, les pièces à revêtir et l'environnement [20]
La figure 3-6 présente la programmation hors-ligne à l'aide d'un logiciel CFAO robotique.

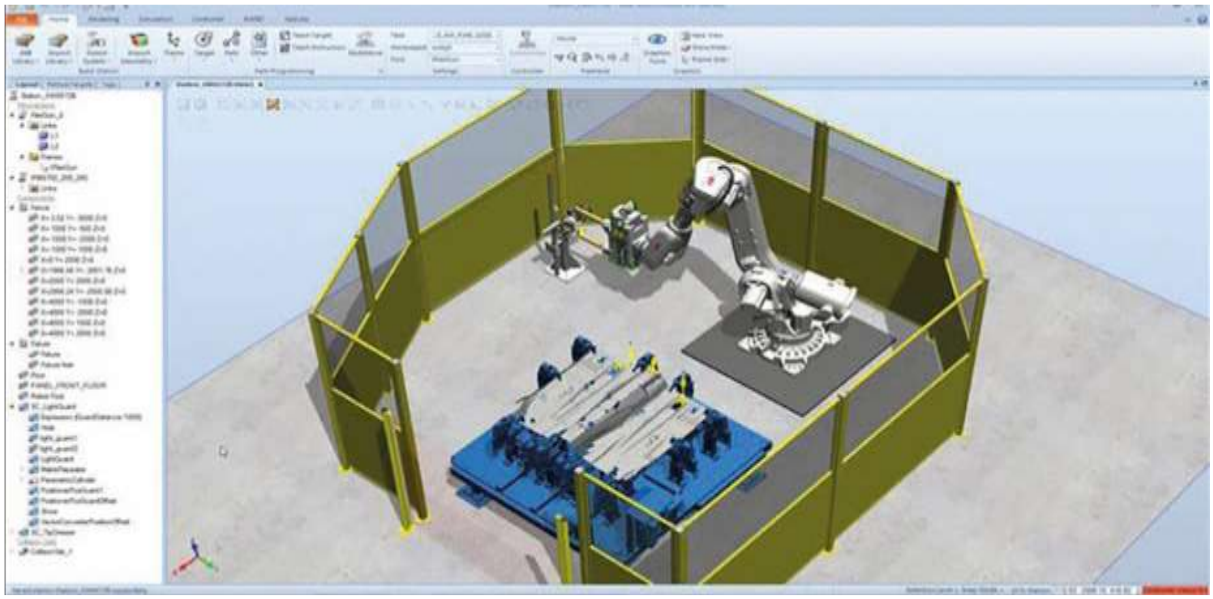


Figure 3-6 **:** Programmation graphique

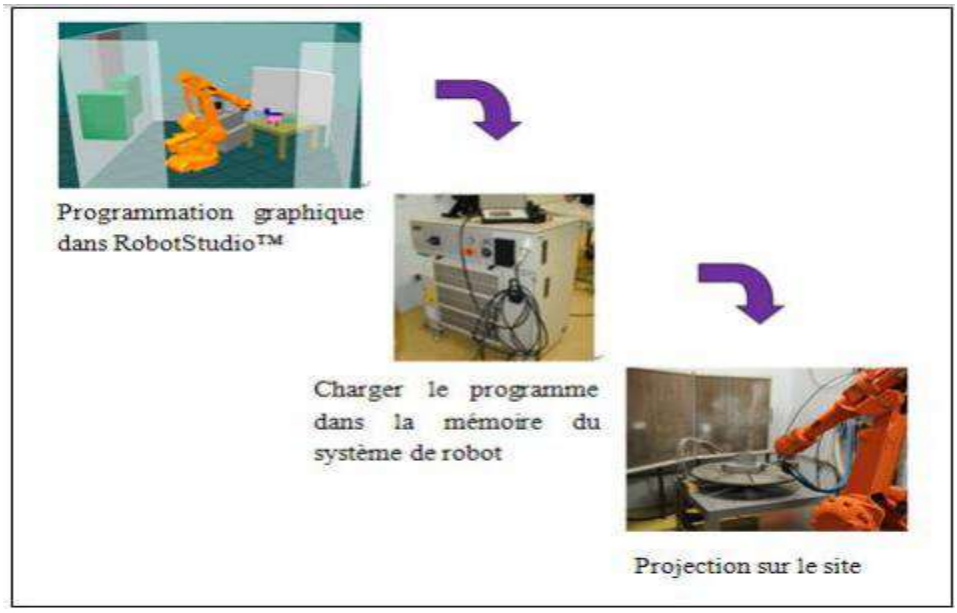


Figure 3-7 **:** Programmation graphique

3.2. Logiciel de programmation hors-ligne : RobotStudio™ :

3.2.1. RobotStudio™ :[19]

RobotStudio™ est un logiciel de PHL fourni par l'entreprise
ABB. Celui-ci est spécialement conçu pour la réalisation des tâches robotiques telles que la PHL, 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.
RobotStudio™ fournit la possibilité de concevoir le programme du robot à partir d'un ordinateur, avec une méthode de programmation d'apprentissage.

3.2.2. Les avantage 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.
- Environnement de simulation 3D- représentation 3D complète des systèmes robotiques.
- Pupitre mobile virtuel d'apprentissage- parfait pour la formation hors -ligne des opérateurs.
- Bibliothèque de système du robot : tous les types de modèles robots des produits ABB sont disponibles. [19]

3.2.3. Site robotisé virtuel et génération de la trajectoire du robot :

La modélisation du site du robot permet de générer une station virtuelle qui comprend tous les outils essentiels à la génération de la trajectoire du robot. Une station de robot peut contenir un ou plusieurs robots, des outils, des positionneurs et d'autres équipements nécessaires. Le processus de modélisation de l'équipement du robot doit tenir compte de la précision de la trajectoire, car celle-ci peut directement introduire des erreurs importantes lors de la synchronisation du programme sur site. La création d'une station à partir d'un système existant est

assurée par RobotStudio™. Cela génère une nouvelle station virtuelle qui inclut les modèles de robot et les contrôleurs de robot correspondants.
Les positions (cibles) et les trajectoires (séquences d'instructions de mouvement vers les positions) sont utilisées lors de la programmation des mouvements du robot dans RobotStudio™ [19].
Les positions représentent les coordonnées que l'ODC (centre de l'outil) du robot doit atteindre et l'orientation que l'ODC du robot doit suivre dans l'espace tridimensionnel. La trajectoire est une série d'instructions qui permet de déterminer les commandes pour exécuter les positions et la façon dont le robot arrive à ces positions (orientation du mouvement et configuration du robot).
RobotStudio™ fournit deux méthodes pour générer des positions de trajectoire :

- Les positions des points de trajectoire peuvent être générées manuellement : un module est fourni pour saisir les positions des points de trajectoire en pilotant manuellement au niveau du CDO. L'emplacement et l'orientation de trajectoire sont choisis par l'utilisateur à partir du CDO de la pièce dans RobotStudio™.
- Génération de la trajectoire à partir d'une courbe : si la pièce présente des courbes ou une série de courbes importées ou créées par l'utilisateur dans la station, la trajectoire complète peut être générée automatiquement à partir de ces courbes. Figure 1.15 présente une trajectoire générée par cette méthode.

3.2.4. Station de RobotStudio :

D'une certaine façon, une station de RobotStudio correspond à une cellule de robot dans une usine. La station contient le robot, l'outil, les pièces de travail, les programmes et tout ce qui peut être utile lors de la simulation. Une seule station peut être ouverte lors de l'utilisation du logiciel.
Cette station peut être enregistrée dans un fichier. Les fichiers de station portent l'extension <<.stn>>. Un fichier de station contient :

- Un système de coordonnées dans lequel sont positionnés les objets de la station.
- Des références aux objets de la station.
- Des informations sur le système de commande virtuel utilisé dans la station.

3.3. La CAO : [20]

Nous pouvons définir la **C**onception **A**ssistée par **O**rdinateur (**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. Un logiciel de **CAO** se compose généralement de quatre parties majeures qui peuvent être organisées comme suit :

3.3.1.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,

3.3.2. L'outil de visualisation.

3.3.3. 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.

3.3.4. Contrôleur :

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

La technologie **CAO** a pris naissance au sein des grands programmes militaires américains dans les années 1950. Ensuite, elle a pénétré le domaine de l'aéronautique civile, l'automobile, l'industrie informatique, l'architecture, le génie civil. Nous pouvons distinguer plusieurs générations de systèmes de CAO (***Figure 3-8***) qui peuvent être classifiées, d'un point de vue historique, selon [Di Monaco et al., 2010], de la manière suivante:

- le **D**essin **A**ssisté par **O**rdinateur (**DAO**) : les objets sont représentés par la projection de leurs arêtes sur un plan bidimensionnel 2D.
- la représentation dite fil de fer (Wireframe) : les objets sont représentés par ses arêtes mais dans l'espace tridimensionnel 3D.
- la représentation des objets par leurs frontières surfaciques B-REP (***B****oundary* ***Rep****resentaion*).
- la représentation par l'espace tridimensionnel occupé par l'objet, cette génération est appelée la technique.

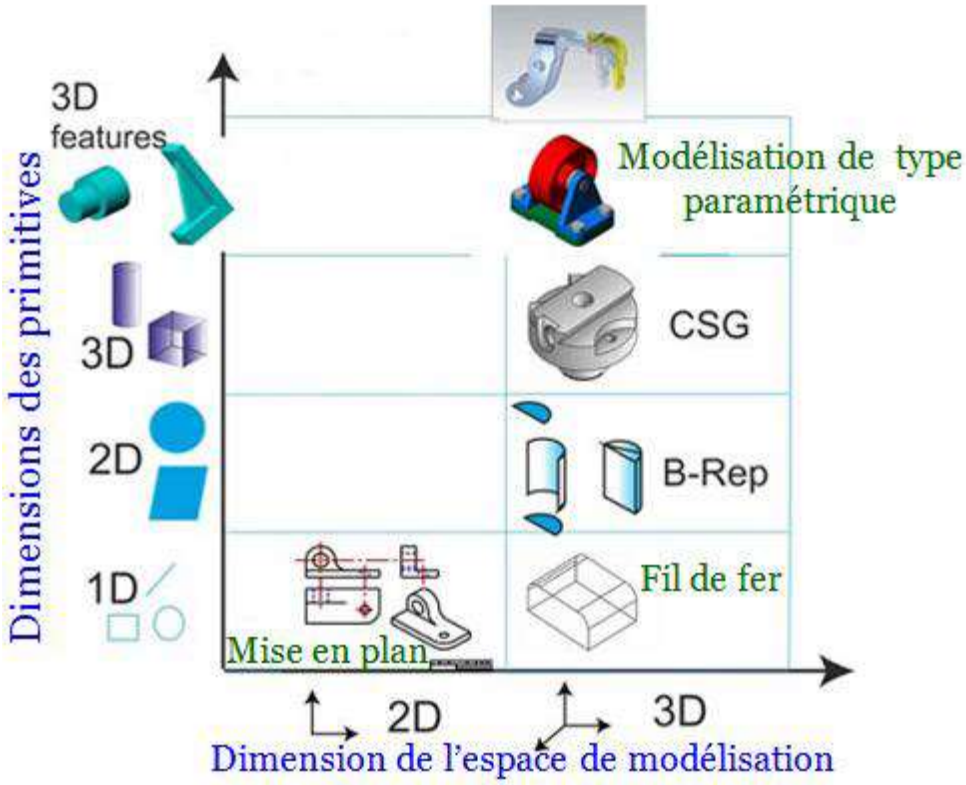


Figure 3-8 **:***Le diagramme représentant les cinq générations de systèmes de CAO*

3.3.5. Logiciels CAO en robotique :

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.

3.4. La programmation :

On peut programmer le robot à partir du RobotStudio. Ou les programmes sont développés en langage texte.

3.4.1.Le langage RAPID :[21]

Le langage de programmation des robots ABB s'appelle RAPID. Un programme en RAPID est un ou plusieurs fichiers texte non compilés (ASCII). Le contrôleur du robot interprète le code, tout en validant la syntaxe de l'ensemble du programme. Un programme qui contient des erreurs de syntaxe peut être chargé dans le système de commande, 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. Pour le système de commande du robot, le fait que le programme soit écrit dans un ou plusieurs modules ne fait pas de différence. L'utilisation de plusieurs modules ne se justifie que par la facilité avec laquelle le programmeur peut saisir et réutiliser les modules. D'autre part, il ne peut y avoir qu'un seul programme actif dans le système de commande du robot, c'est-à-dire qu'un seul des modules peut contenir une procédure appelée "main". Lors du chargement des modules individuels dans le contrôleur, le contrôleur considère automatiquement l'ensemble des modules comme un seul programme. Les modules peuvent être chargés séparément, un par un, ou en utilisant le second type de fichier : le programme (.pgf), qui est simplement un fichier de type XML qui spécifie la liste des modules ˆ charger. Le contrôleur exécute le programme avec la procédure "main".
Celle-ci doit se trouver au début d'un module, juste après la déclaration des variables globales de la tâche.

3.4.2. 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. [21]

3.4.3. Type de variables de base : [21]

Le langage RAPID offre la possibilité au programmeur de se créer des registres mémoires a fin 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.

3.4.4. 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). [21]

3.4.5. La pose :[21]

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]].

3.4.6. La wobjdata :[21]

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 3-9). 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_à l'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).

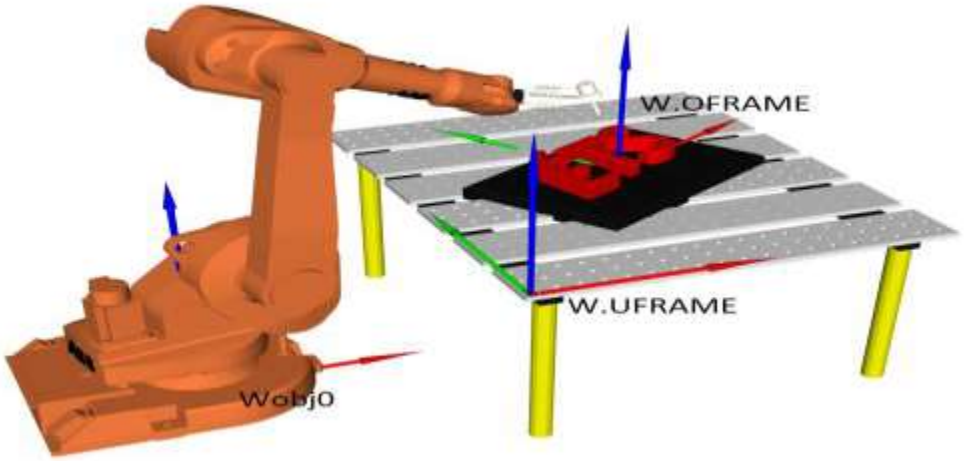


Figure 3-9 : Référentiels faisant partie d'un enregistrement de type workoject,

3.4.7. Le robtarget :[21]

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 et6, ainsi que le numéro de configuration (de 0 à 7, dans le cas des robots sériels à 6ddl).

- extax (extjoint) : Permets d'interagir avec les articulations externes du robot.

3.4.8. La tooldata :[21]

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.

3.5. Les type de déplacements :

3.5.1. Déplacement linéaire :

Dans un déplacement linéaire, l'origine du référentiel de l'outil spécifié suit une trajectoire linéaire (figure 3-10). La réorientation de l'effecteur, si nécessaire, se fait de façon automatique (l'opérateur ne peut pas influencer cette réorientation). Ce type de déplacement est très contraignant, car il oblige souvent le robot à réaliser des mouvements complexes et parfois brusques.

Figure 3-10 **:** Déplacement linéaire de l'outil

Un déplacement linéaire peut également être limité parce 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. Les singularités ne sont pas liées à une marque de robot, mais bien au design mécanique ce qui signifie que certains robots en possèdent plus que d'autres et que toutes les marques de robot ont ce problème avec certains de leurs modèles. Par contre, chaque fabricant du robot a des limitations spécifiques additio-nnelles lors d'un déplacement linéaire.
Les paramètres de base d'une commande de linéaire sont comme suit :
MoveL robtarget, speeddata, zonedata, tooldata, \wobj:=wobjdata;
\wobj:=wobjdata;

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

3.5.2. Déplacement articulaire :

Dans un déplacement articulaire, la trajectoire suivie par l'effecteur du robot est difficilement prévisible (figure 3-11). 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 ou 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 Figure 3-11 **:**

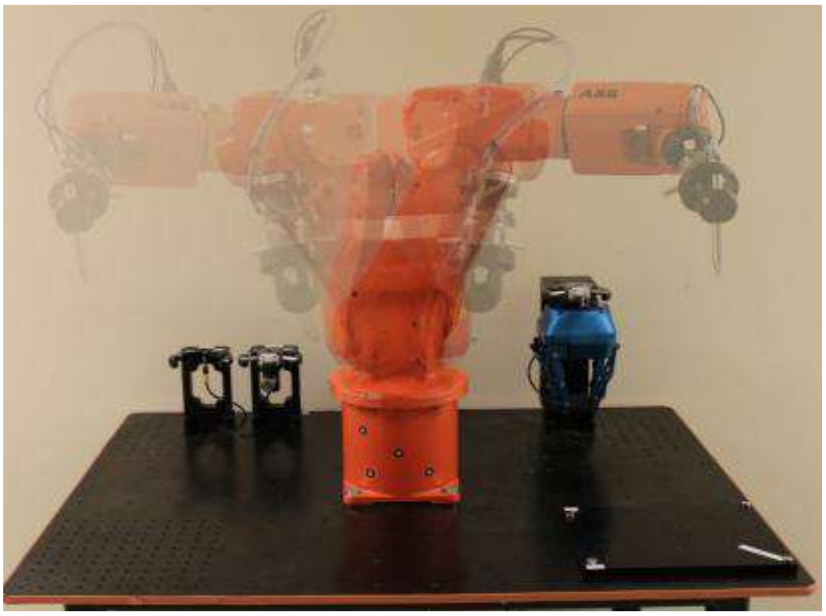


Figure 3-11: Déplacement articulaire de l'outil (commande MoveJ).

Déplacement articulaire de temps, à différentes vitesses, afin de toutes les (commande MoveJ). L'outil arrêté en même temps, à la destination finale (définie par un robtaget).
Les paramètres de base d'une commande de déplacement articulaire sont comme suit :
MoveJ robtarget, speeddata, zonedata, tooldata, \wobj:=wobjdata;

- MoveJ : Commande indiquant un déplacement articulaire de l'endroit ou il se trouve vers le robtarget spécifié.
- robtarget : Valeurs venant d'une variable, fonction ou autre qui indique la destination de l'outil.
- speeddata: Vitesse de croisière maximale que l'outil peut atteindre durant le déplacement.
- zonedata : Précision d'arrêt ou d'approche de l'origine du référentiel de l'outil sur un robtarget à atteindre.
- tooldata : Le référentiel de l'outil et les caractéristiques physiques de l'outil.
- wobjdata : Paramètre optionnel qui définit par rapport à quel référentiel de travail on veut positionner le référentiel de l'outil à la pose définie dans le robtarget. Si ce paramètre est omis, le référentiel par défaut est celui de l'atelier (appelée wobj0).

3.5.3. Déplacement circulaire :

Dans un déplacement circulaire, l'origine du référentiel de l'outil spécifié suit une trajectoire circulaire passante par trois robtargets (le robtarget actuel, un intermédiaire et un final). Un déplacement circulaire a les mêmes contraintes

qu'un déplacement linéaire puisque l'on force le trajet. Il est donc à risque pour ce qui concerne les singularités.
Les paramètres de base d'une commande de déplacement articulaire sont comme suit :
MoveC robtarget, robtarget, speeddata, zonedata, tooldata, \wobj:=wobjdata;

- MoveC : Commande indiquant un déplacement circulaire par calcul de trois points (position courante, une valeur intermédiaire et une valeur finale).
- robtarget (1er) : Valeur venant d'une variable, fonction ou autre qui indique le robtarget intermédiaire par lequel l'outil doit passer.
- robtarget (2e) : Valeur venant d'une variable, fonction ou autre qui indique la destination finale de l'outil.
- speeddata : Vitesse de croisière maximale que l'outil peut d'atteindre durant le déplacement.
- zonedata : Précision d'arrêt ou d'approche de l'origine du référentiel de l'outil sur un robtarget à atteindre.
- tooldata : Le référentiel de l'outil et les caractéristiques physiques de l'outil.
- wobjdata : Paramètre optionnel qui définit par rapport à quel référentiel de travail on veut positionner le référentiel de l'outil à la pose définie dans le robtarget. Si ce paramètre est omis, le référentiel par défaut est celui de l'atelier (appelée wobj0).

3.6. Les commandes de déplacement :

Le but premier d'une commande de déplacement est de déplacer l'outil du robot d'une manière spécifique. Dans les commandes de déplacements les plus utilisées, on doit toujours spécifier un robtarget comme cible. Il existe trois types de déplacements dont la cible est un robtarget.

3.7. Vitesse et précision du déplacement :

3.7.1. 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.).[21]

3.7.2. L'interpolation : Figure 3-12

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. [21]

- **finep :** Variable booléenne définissant si le robot doit s'arrêter ;
- **pzone-tcp :** rayon en mm de la sphère ou l'origine du référentiel de l'outil peut commencer l'interpolation vers le prochain trajet ;
- **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.

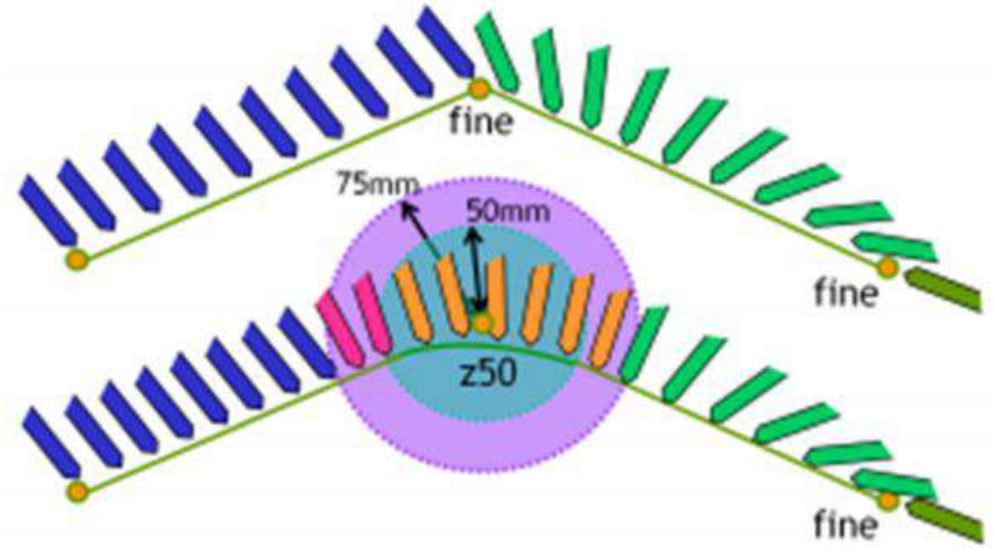


Figure 3-12 **:** Interpolation entre deux déplacements linéaires.

3. 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, une attention particulière a été donnée à ce sujet.

CHAPITRE 4 : La simulation avec RobotStudio

4- Introduction :

RobotStudio utilise des systèmes de commande virtuels pour piloter les robots.
Les systèmes de commande virtuels peuvent exécuter des systèmes de véritables Robots et des systèmes virtuels particuliers pour le test et l'évaluation. Un système de commande virtuel utilise le même logiciel que le système de commande pour exécuter le programme RAPID, pour calculer les mouvements du robot et pour gérer les signaux d'E/S.
Parmi Les systèmes de commande virtuel les plus connus on trouve les systèmes à convoyeurs qui sont nécessaires et primordiaux dans l'industrie et surtout dans les unités de production, ainsi que le soudage qui est une tache indispensable dans tous les secteurs de l'industrie.
Donc dans ce projet on a réalisé une cellule robotisée pour faire le soudage avec deux Robots de types IRB 2600 sur des pièces métalliques avant de leurs déplacer dans un convoyeur avec un Robot de type IRB 6640.
Tout d'abord on va présenter les deux robots et le système de contrôle IRC5 avant d'entamer la programmation des taches.

4.1- Robot type IRB 2600 :

4.1.1 Description :

L'IRB 2600 est le dernier robot ABB Robotics de nouvelle génération doté de fonctions enrichies et innovantes. Sa conception a été optimisée afin de s'adapter parfaitement aux diverses applications ciblées. L'IRB 2600 permettra d'enrichir les fonctions de distribution, d'usinage, de mesure, de montage et de soudage à l'arc. Le robot est équipé du système de commande IRC5 et du logiciel de commande du robot, RobotWare. RobotWare prend en charge tous les aspects du système de robot, notamment le contrôle des mouvements, le développement et l'exécution des programmes applicatifs, la communication, etc.

4.1.2- Caractéristiques :

Type : articulé
Nombre d'Axes : 6 axes
Application : la manutention et le service de machines et le soudage à l'arc.
Capacité de charge : 12 kg
Rayon d'action : 1.65 m
Répétabilité : 0.05 mm

4.1.3- Les avantages :

- Haute précision
- Design compact
- Des temps de cycle courts
- Une enveloppe de travail remarquable

4.2- Robot type IRB 6640 :

4.2.1- Description :

IRB 6640 est le nouveau robot ABB doit soulever des charges lourdes. Construit pour remplacer son prédécesseur, IRB 6600. Avec six axes de mouvement et une charge utile maximale de 130 kg, prêt à faire face à des défis difficiles. Sa conception robuste est idéal pour la manutention des matériaux, des machines de chargement ou des applications de soudage. La version standard de l'IRB 6640 a une portée horizontale 3,20 m. Il est monté sur un modèle de terre avec la capacité de se courber derrière elle-même pour fournir un accès supplémentaire. Le 6640 peut utiliser à la fois le contrôle ABB S4C + ou le contrôleur IRC5.

1.3.2- Caractéristiques :

Type : articulé
Nombre d'Axes : 6 axes
Capacité: 0/130 Kg
Répétabilité: 0,07 mm
Poids du Robot: 1700 kg
Montage: Au sol

4.2.2- Les avantages :

- Une protection optimale
- Charge utile plus élevée
- Performances de chemin accrues
- Facilité de la maintenance

4.3- IRC5 Controller :

4.3.1- Description :

l'IRC5 est la référence de l'industrie robotique en matière de technologie de contrôleur de robot. En plus du contrôle de mouvement unique d'ABB, il apporte flexibilité, sécurité, modularité, interfaces d'application, contrôle multi-robot et prise en charge d'outils PC. L'IRC5 est disponible en différentes variantes pour fournir des solutions rentables et optimisées pour chaque besoin.

4.3.2- Caractéristiques et avantages :

• Conçu pour une protection IP élevée et une protection complète L'extensibilité.
• Fournit un environnement protégé pour les axillaires Équipement dans le système de robot.
• Capable de contrôler jusqu'à quatre robots dans un Configuration MultiMove.
• Rapide, précis et facile a la programmation. [22]

4.4- Programmation des Tâches :

Dans cette partie nous allons effectuer la simulation avec le logiciel ABB Robotstudio d'une cellule robotisée composée de trois robots et un convoyeur ainsi que les étapes et les procédures suivies pour arriver à notre objectif.

- En premier temps on va créer une station vide.

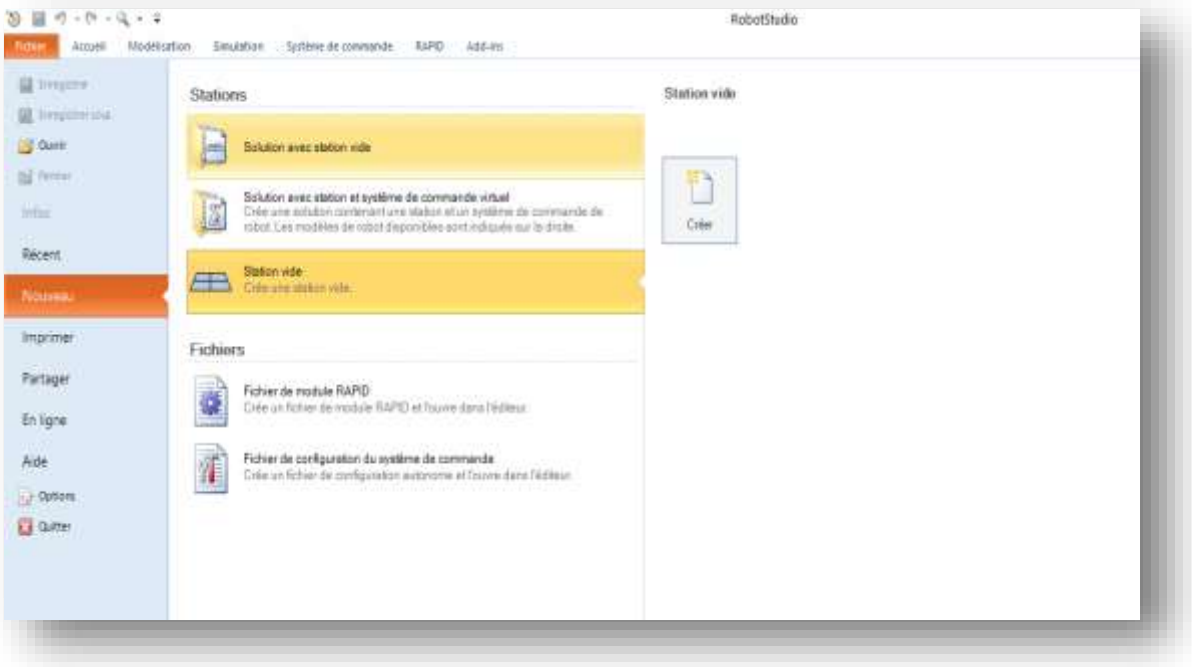


Figure (2.1) : création d'une station vide

Apres la création d'une station vide on va ajouter les éléments matériels nécessaires pour débuter notre travail c.-à-d les Robots IRB 2600 et les pièces métalliques.

On va tout d'abord aller sur Accueil ► construction de la station ► Bibliothèque ABB ► Importer le robot IRB 2600.

Le IRC5 singel-cabinet se trouve dans construction de la station ► importer bibliothèque ► Equipements.

Pour importer les pièces métalliques on va sur Modélisation ► solide ► boite.

Après l'importation des pièces on va les modifier dans apparence du graphique.

Figure (2.2) : importation des robots IRB2600

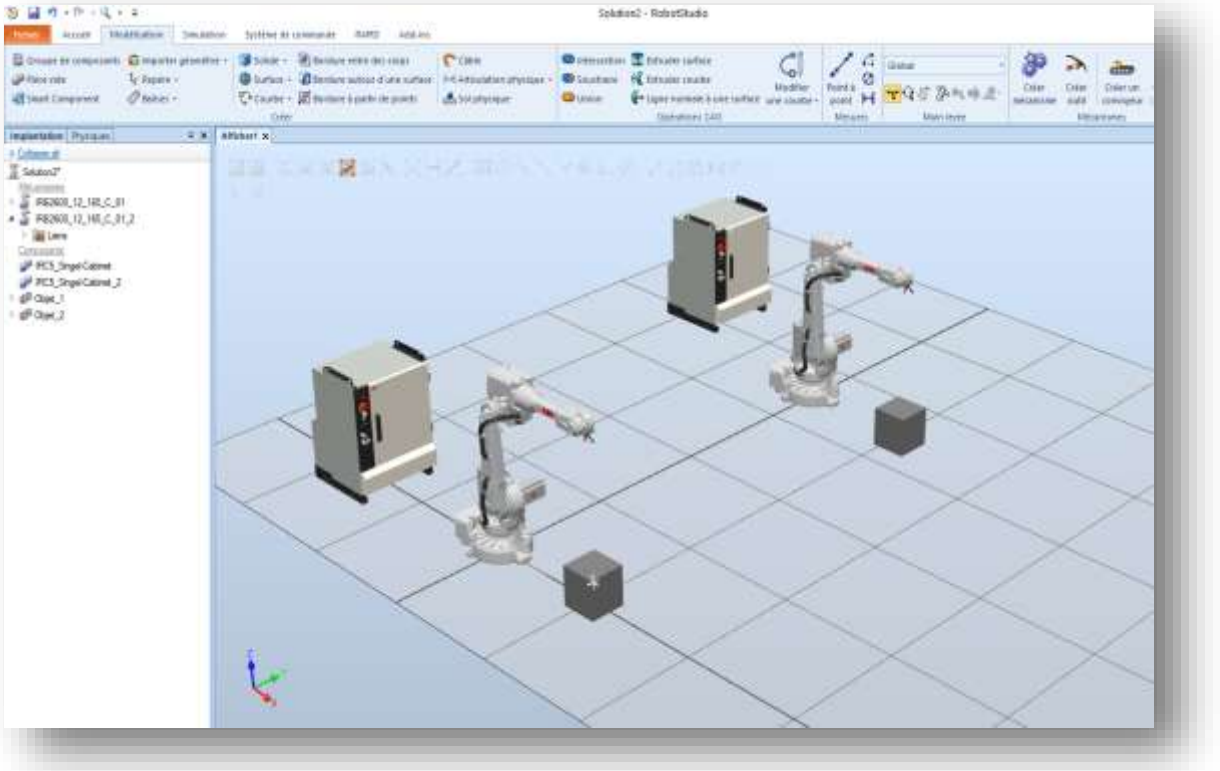


Figure (2.3) : ajouts d'IRC5 single cabinet et la création des pièces métalliques

Après avoir Implanté les pièces on va ajouter les outils de soudage de type « AW Gun PSF 25 » d'après Equipements bibliothèque ABB et les attacher avec le robot.

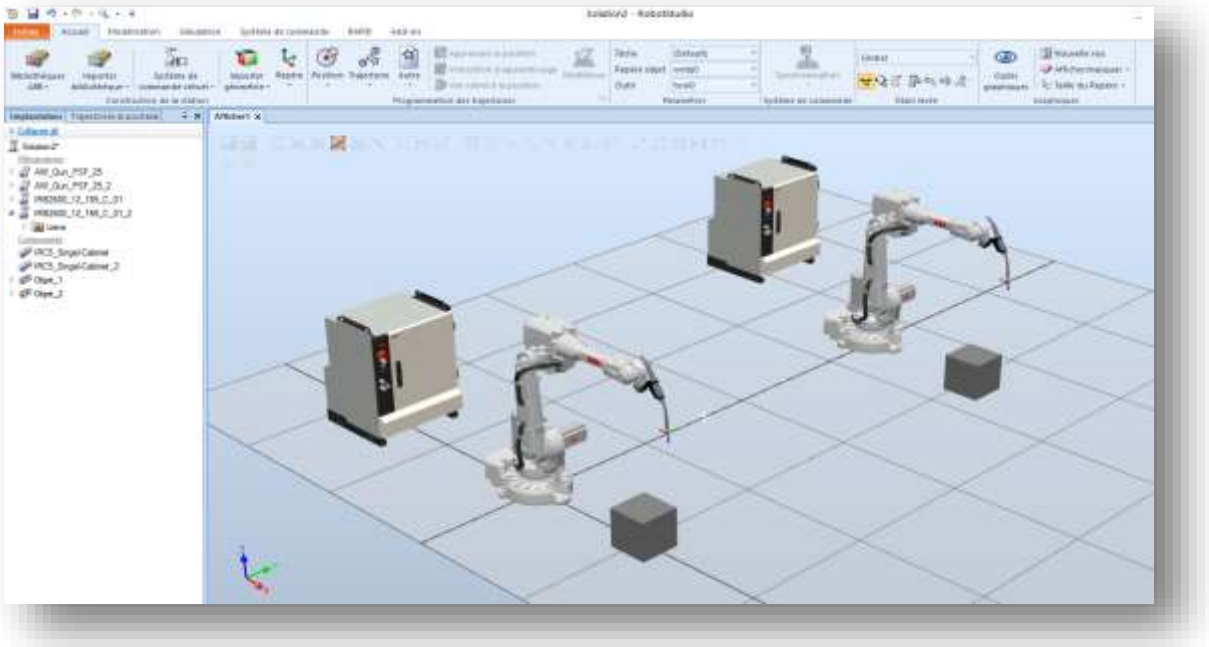


Figure (2.4) : Attachement de l'outil Gun PSF 25 avec le robot IRB 2600

- Maintenant on va déterminer les cibles que le robot doit atteindre pour faire le soudage sur la pièce.
- Ensuite on va créer une trajectoire qui enchaine les cibles.

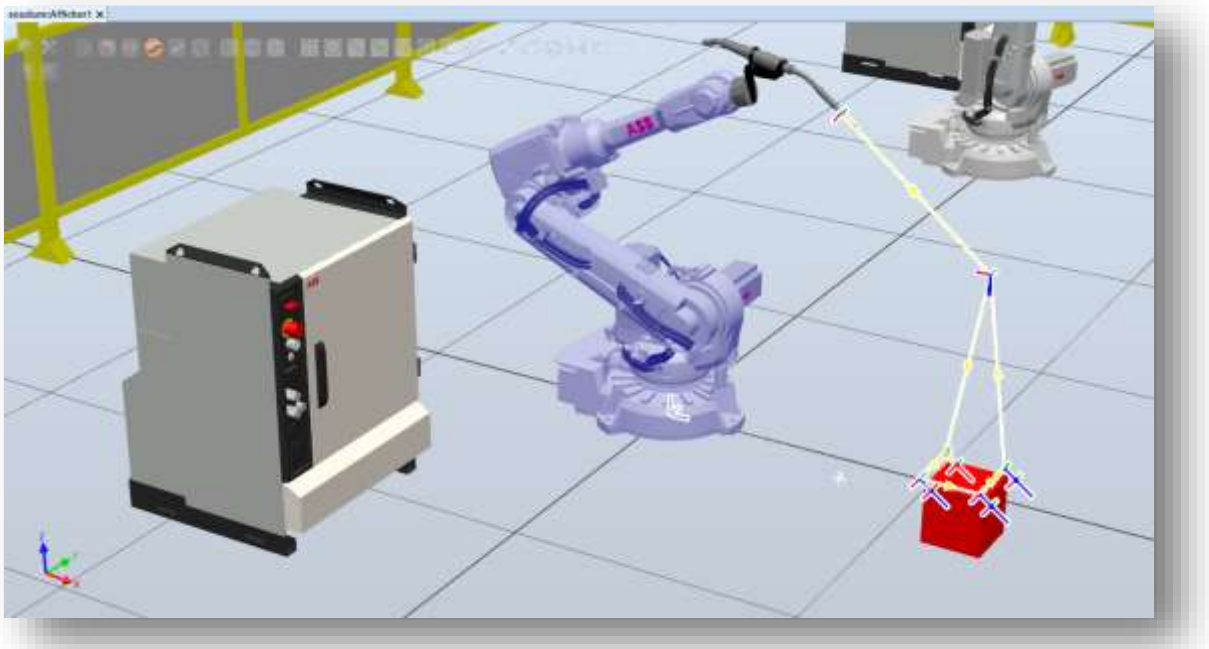


Figure (2.5) : création des points cibles et d'une trajectoire du robot

L'étape suivante est de créer un système de commande pour les deux robots dans construction de la station ► système de commande virtuel ► nouveau système de commande, 'Controller2' pour le premier robot et 'Controller6 ' pour le deuxième.

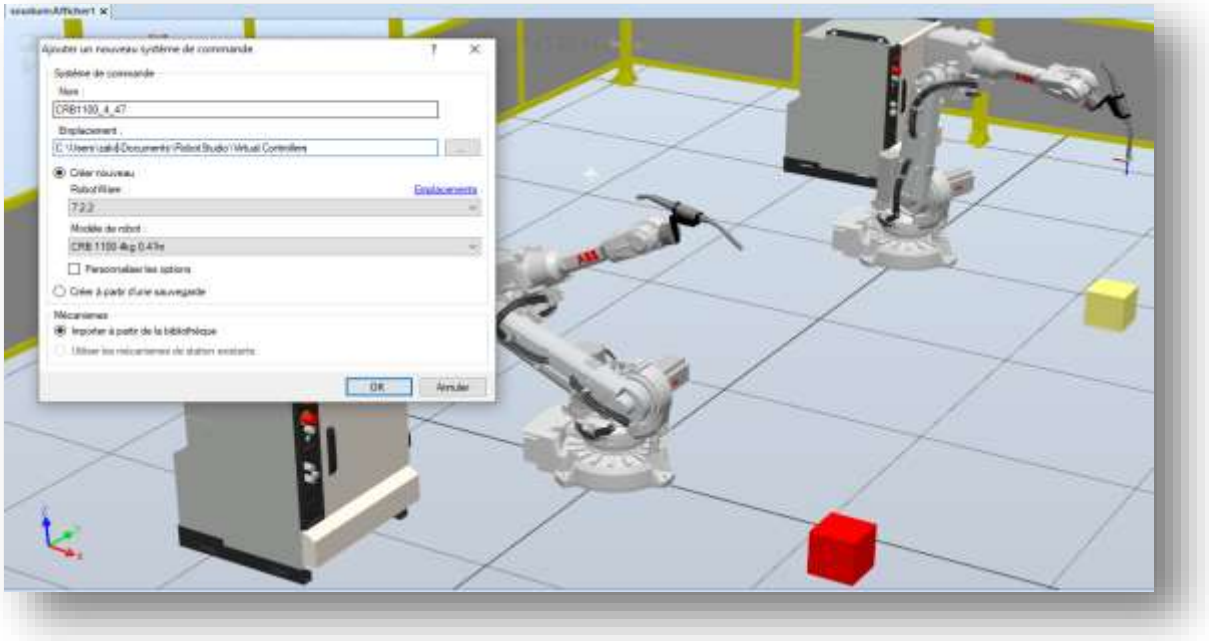


Figure (2.6) : création du système de commande « controller2 »

Maintenant on va créer une relation entre les deux robots pour faire la tâche de soudage en parallèle dans : système de commande ► transfert ► créer une relation.

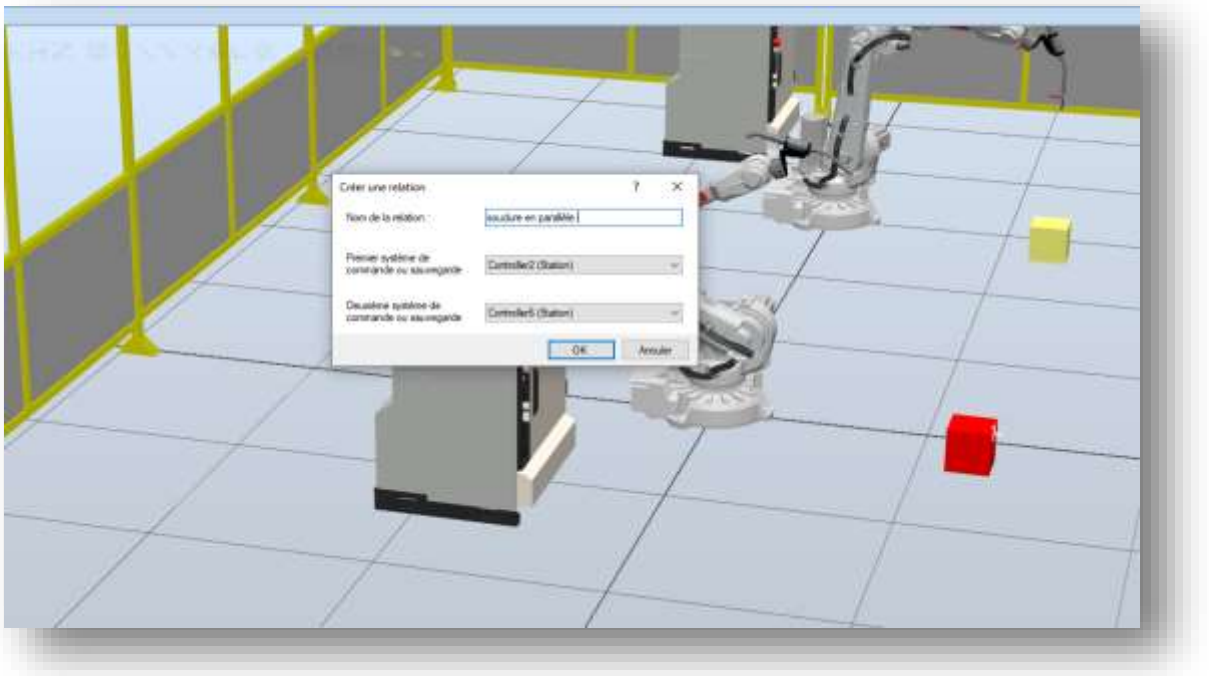


Figure (2.7) : faire la relation entre les deux robots

Après on va sur la synchronisation avec RAPID par les étapes suivantes : RAPID ► Accès ► Synchronisation avec RAPID

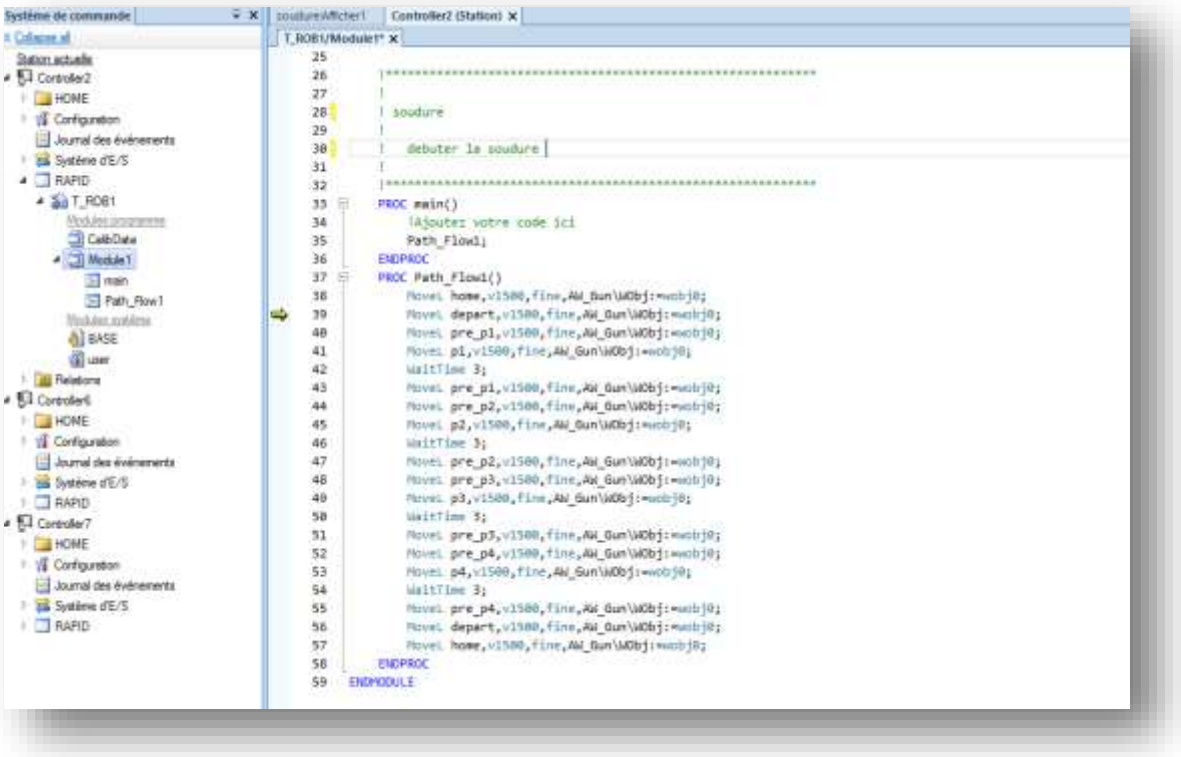


Figure (2.8) : programme RAPID du premier Robot

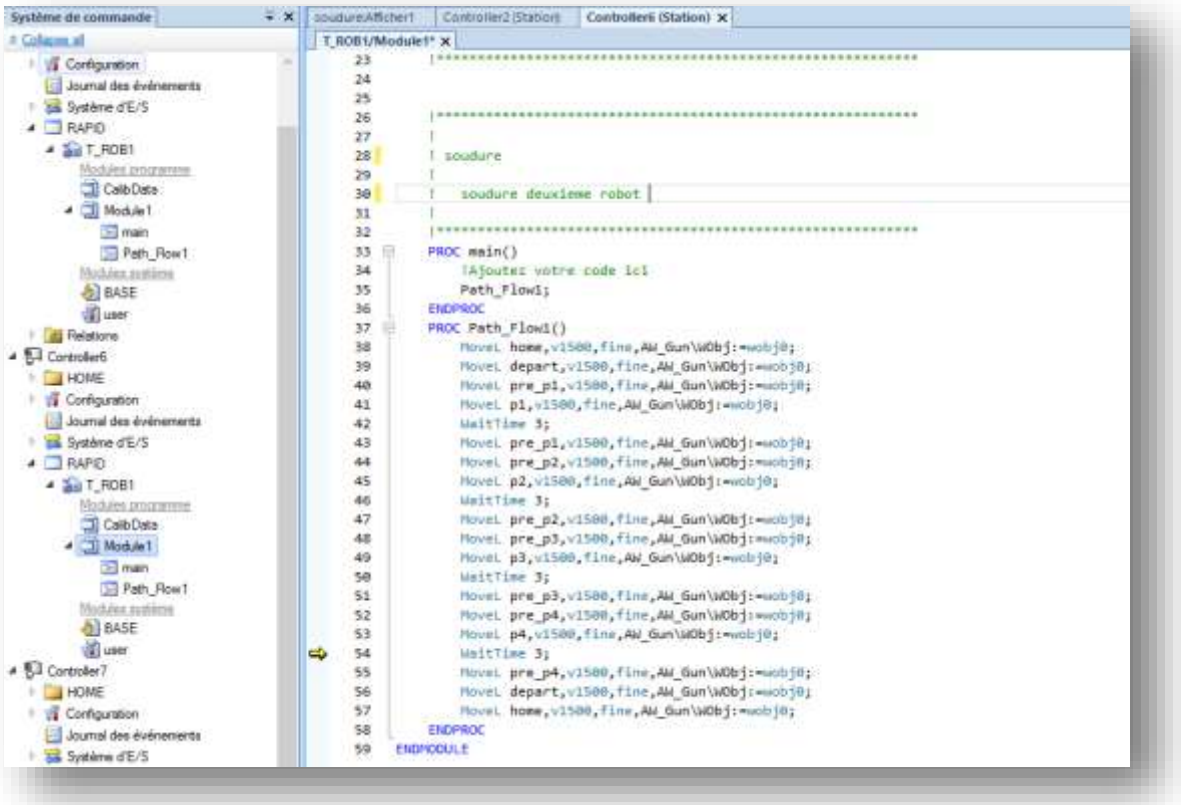


Figure (2.9) : programme RAPID du deuxième Robot

Après la synchronisation avec RAPID on va redémarrer le système et faire l'exécution des tâches de soudage pour les deux Robots.

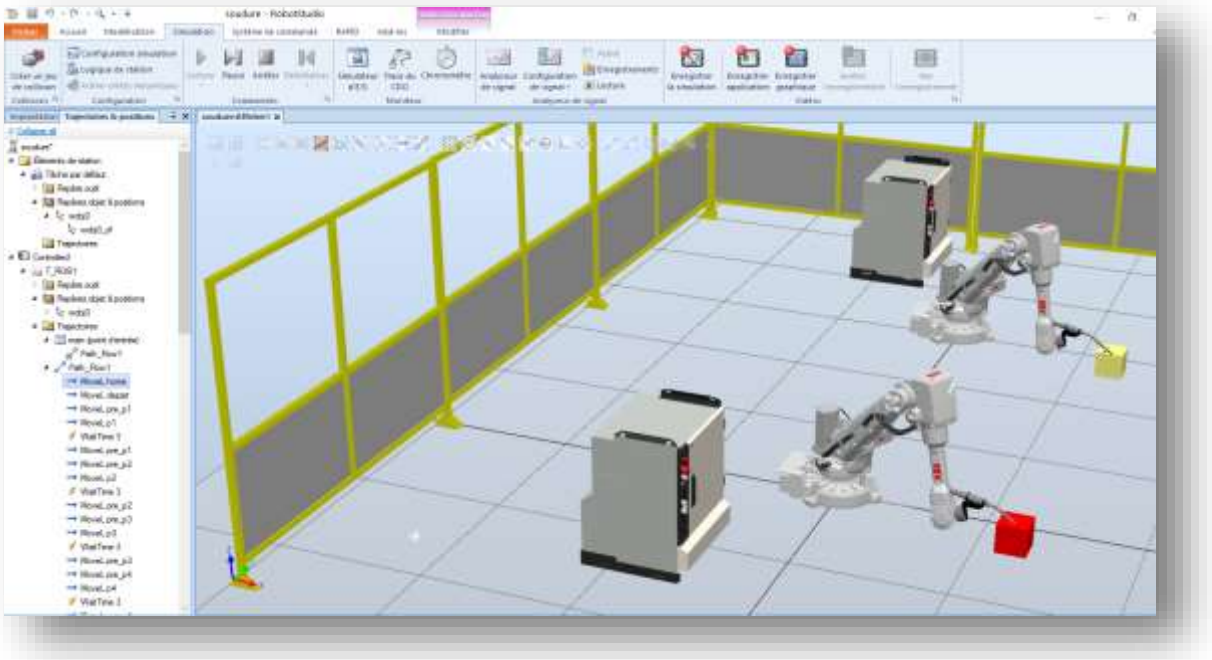


Figure (2.10) : la simulation des tâches de soudage

- Maintenant on importe le troisième Robot de type IRB 6640 et l'outil « Gripper ».
- ensuite, on attache L'outil avec Le Robot.

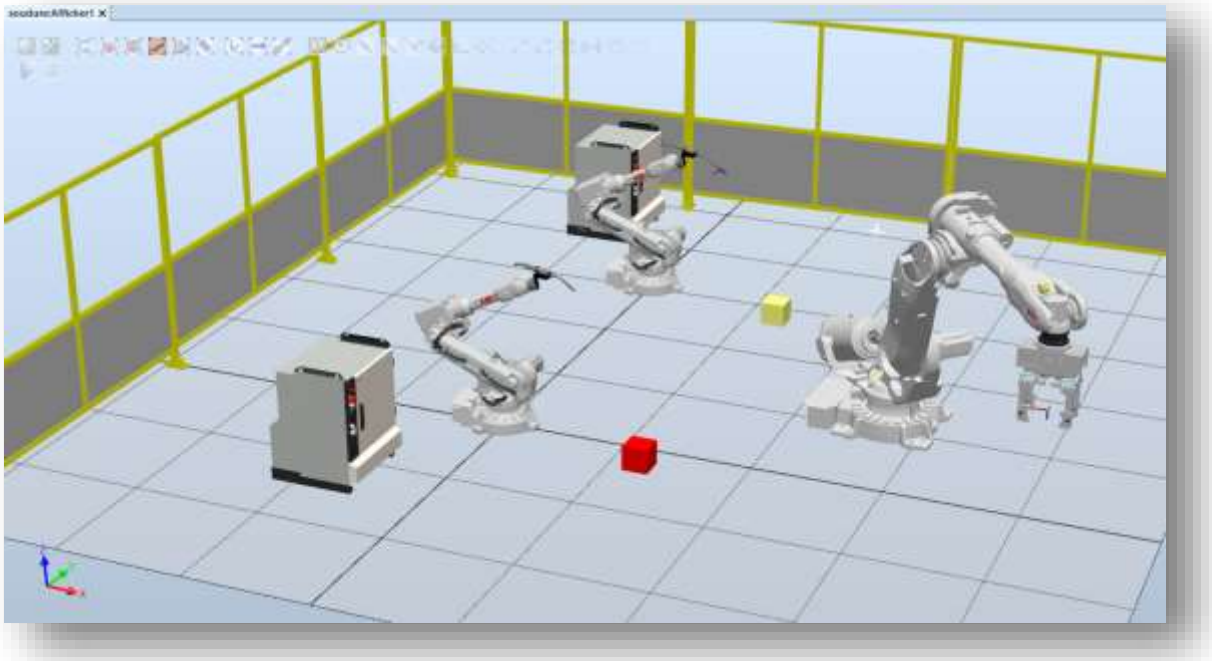


Figure (2.11) : importation et attachement du robot avec l'outil

On ajoute le convoyeur dans Accueil ► importer bibliothèque ► Equipements

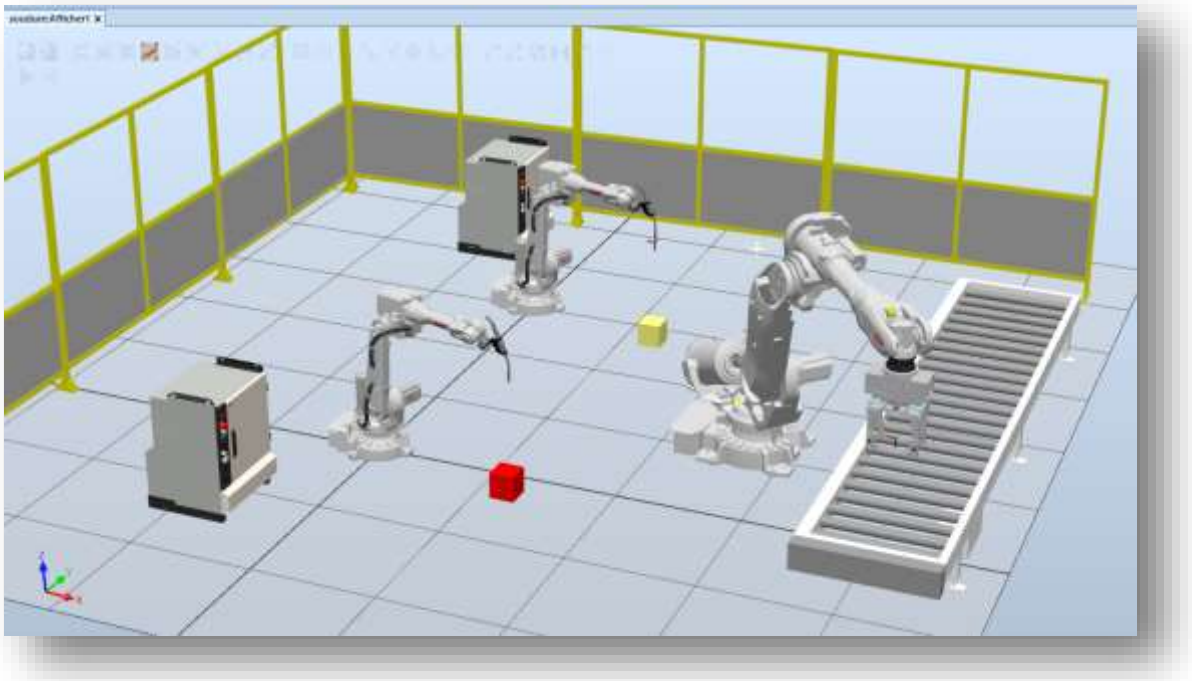


Figure (2.12) : importation du convoyeur

Pour compléter ce système on a besoin des composants intelligents

On clique sur Modélisation ► Smart component

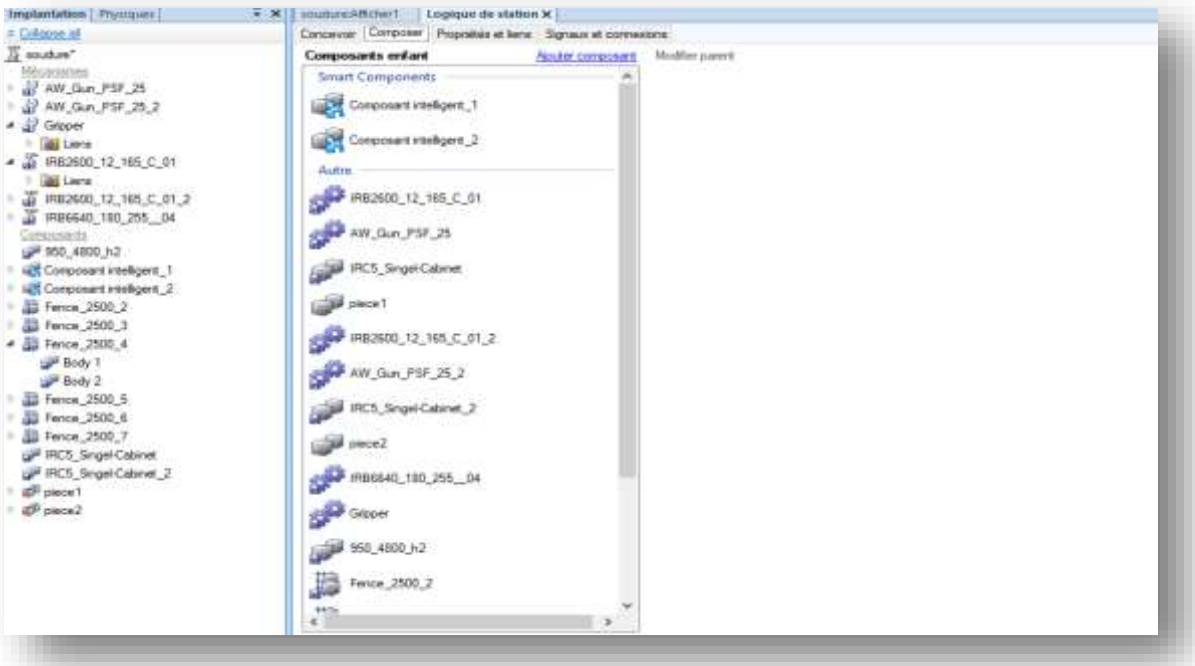


Figure (2.13) : ajouter deux composants intelligents

Maintenant on va ajouter des composants enfant dans le composant intelligent 2 pour assurer tous les taches du robot, les composants sont les suivants :

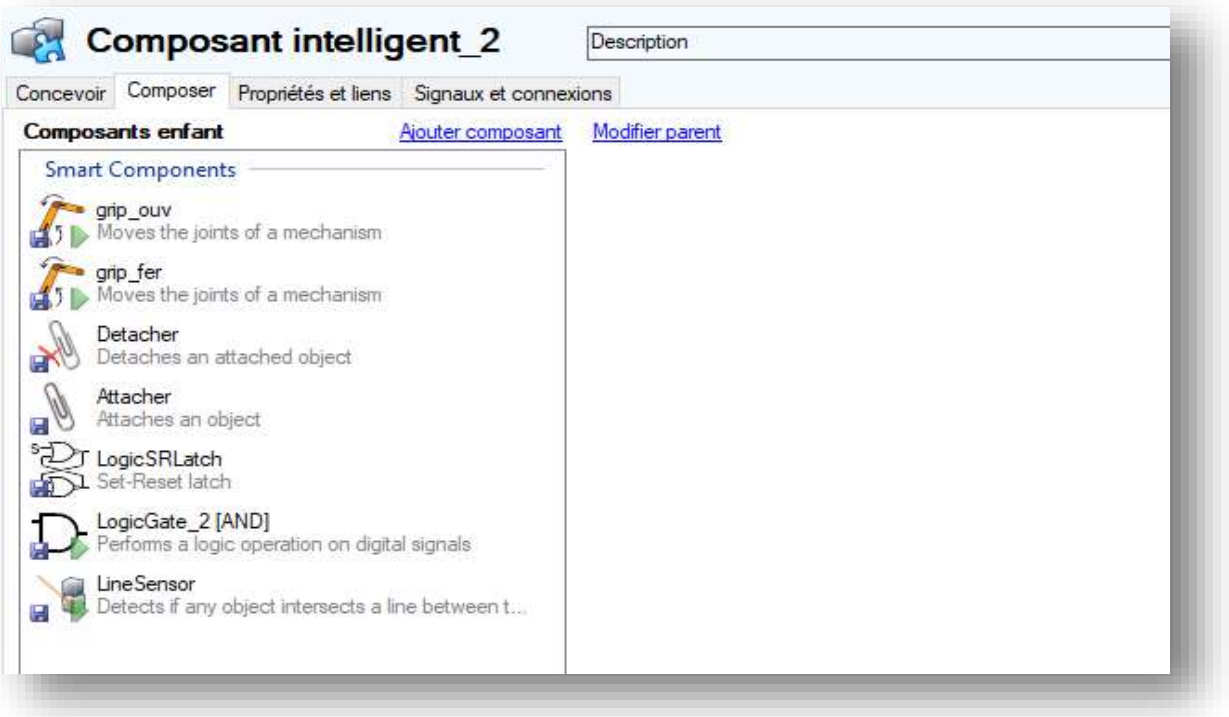


Figure (2.14) : composants enfant du composant intelligent 2

Ensuite on va ajouter des signaux numériques E/S pour le composant intelligent2

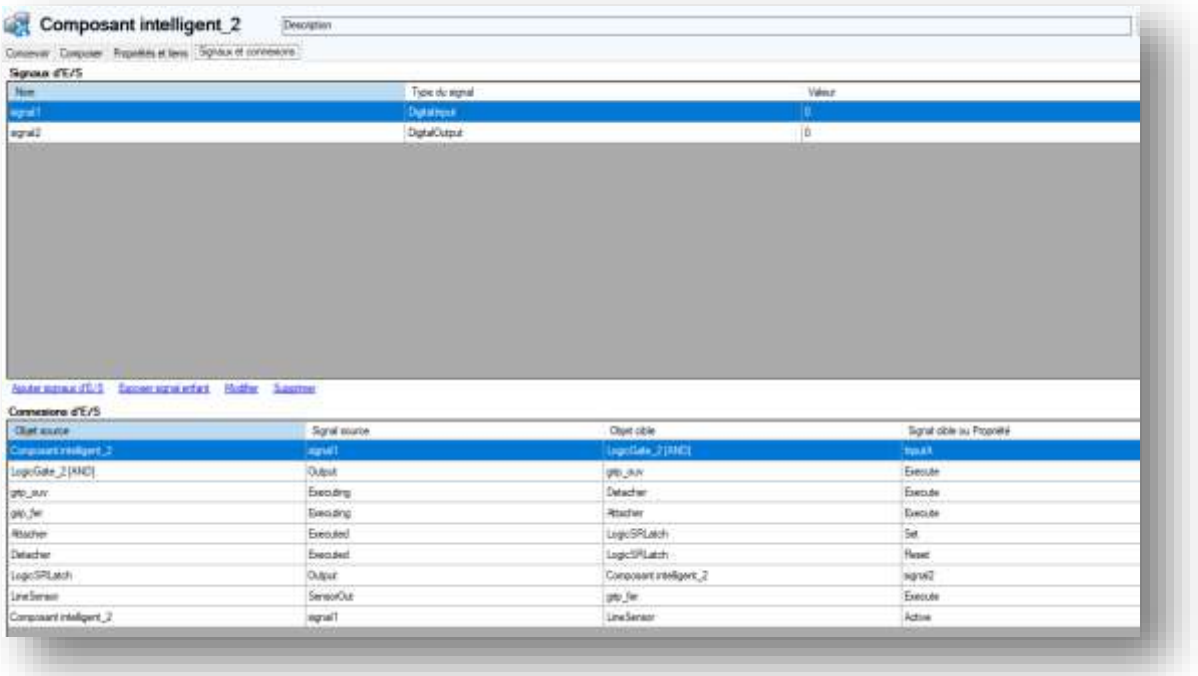


Figure (2.15) : signaux et connexions du composant intelligent 2

Faire la connexion entre les composants dans « concevoir »

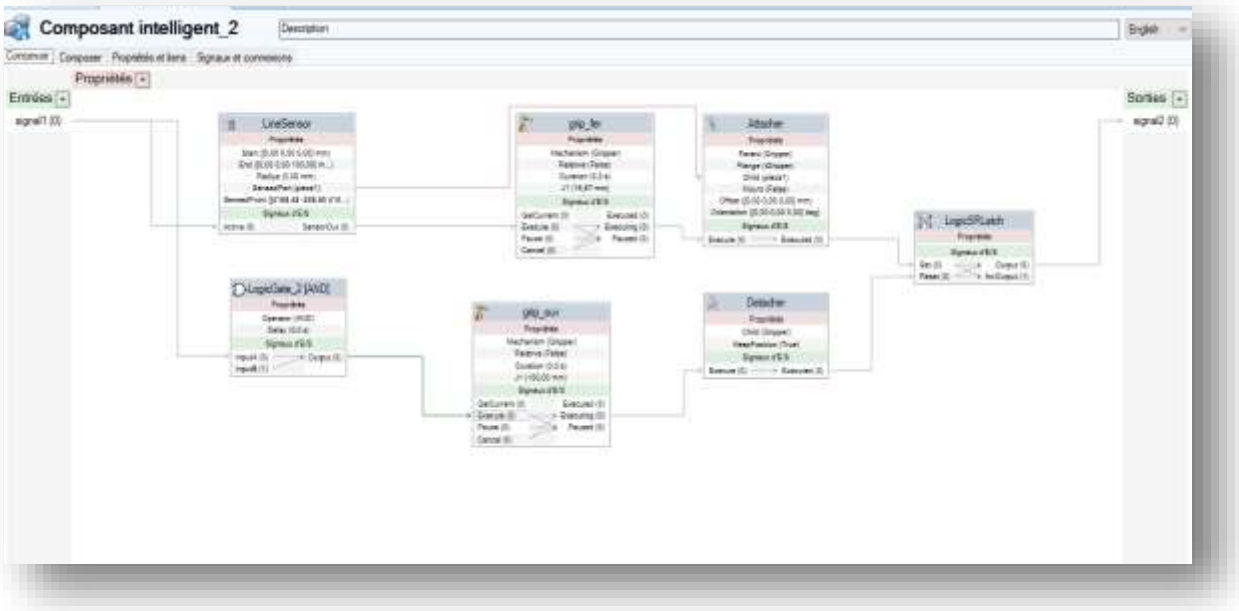


Figure (2.16) : concevoir les composants enfant

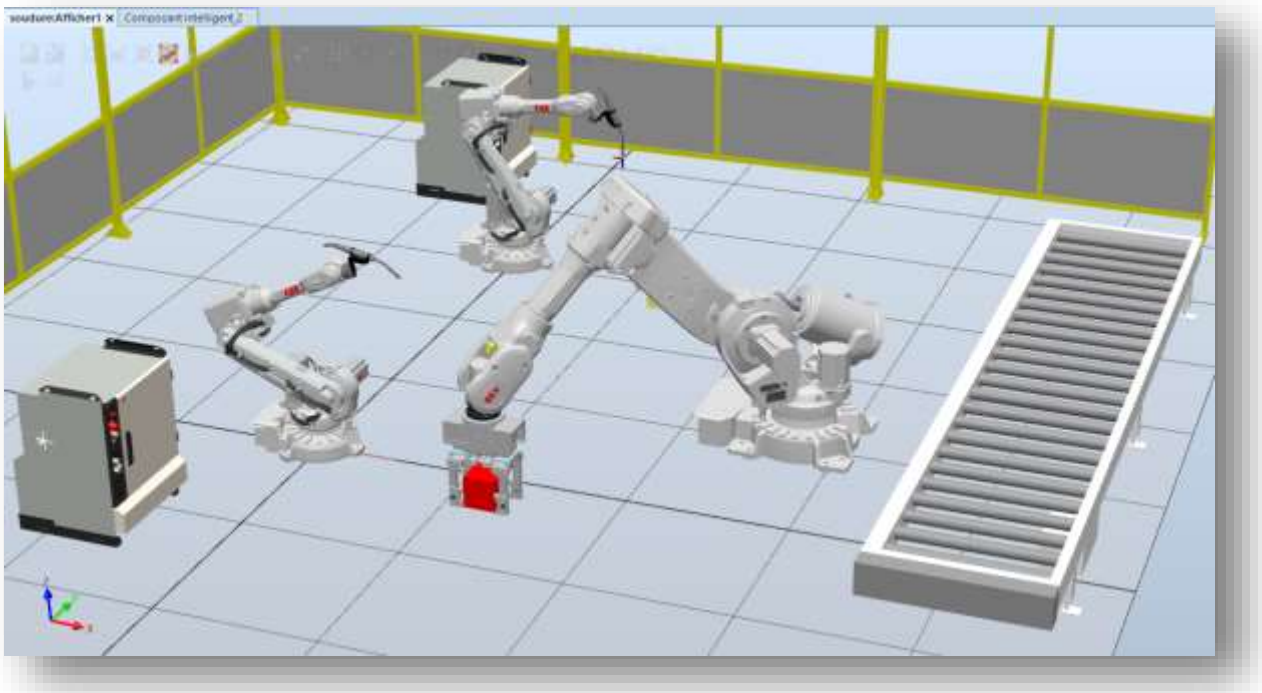


Figure (2.17) : Attachement de la pièce 1 a l'outil Gripper

Après avoir attaché la pièce à l'outil on va créer des points cibles et une trajectoire qui permet le déplacement de la pièce sur le convoyeur.

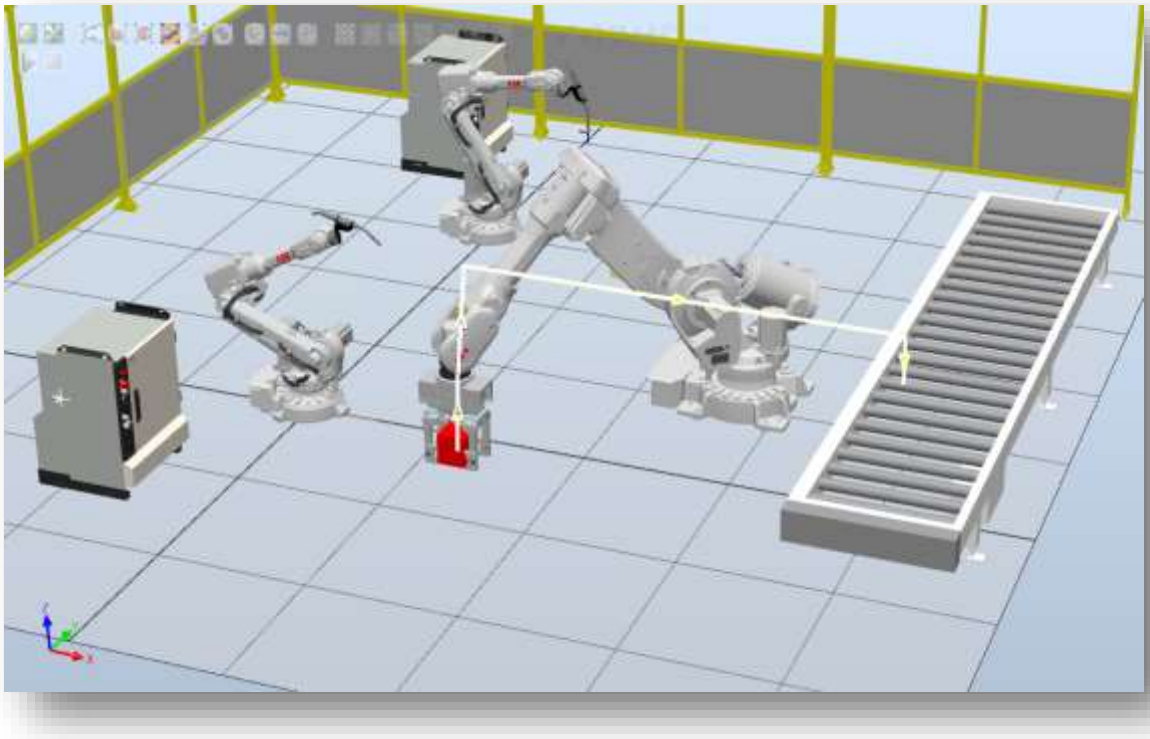


Figure (2.18) : création de la trajectoire et des positions

Ensuite on va créer des composants enfant du composant intelligent 1 au reste on va associer les deux composants.

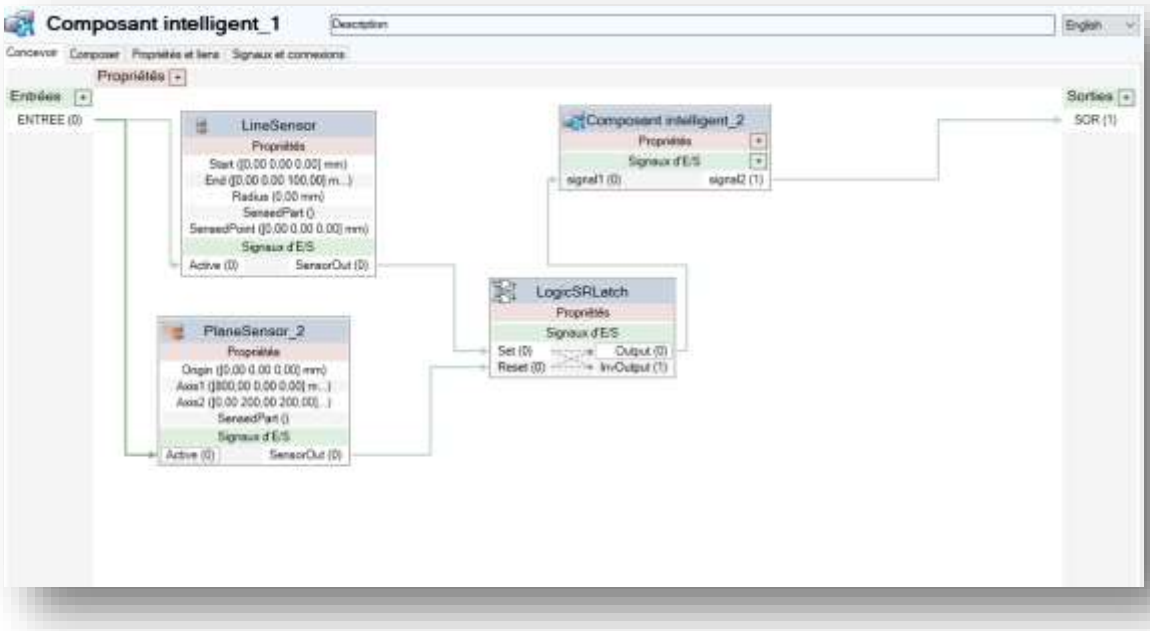


Figure (2.19) : concevoir le composant intelligent 1

Pour compléter la logique de la station et configurer les entrés/ sorties des composants on va sur simulation ► Logique de station ► Concevoir.

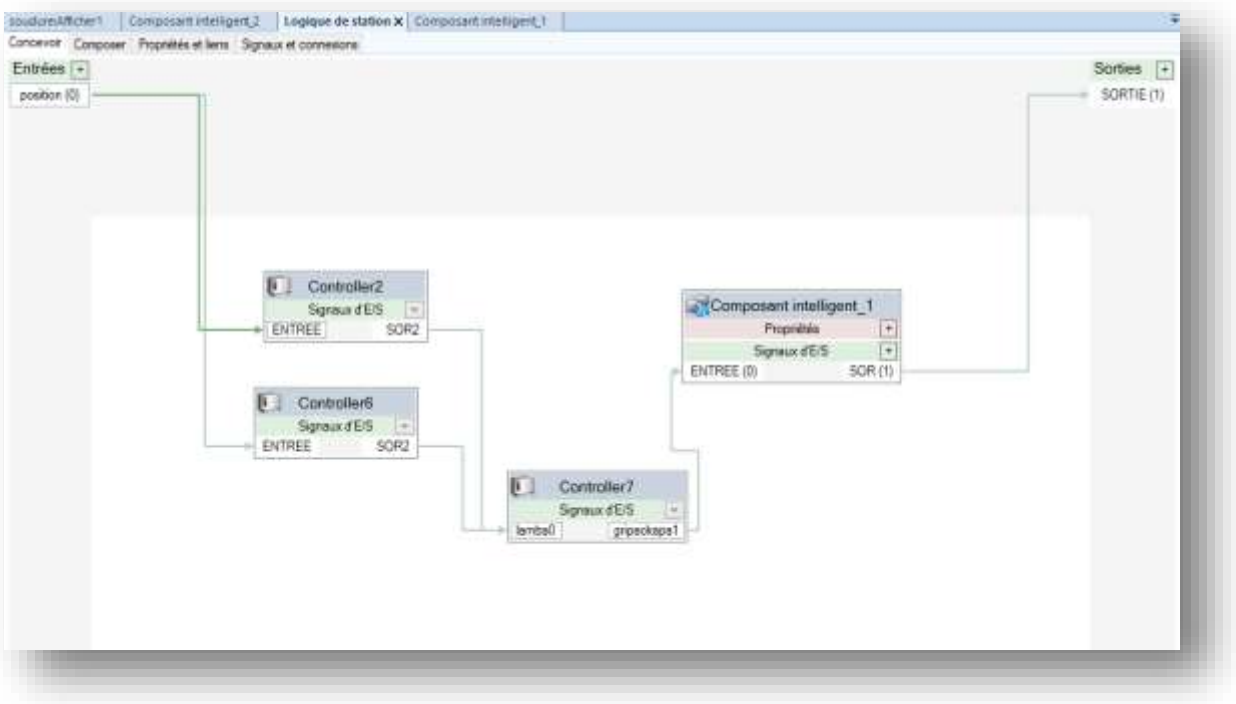


Figure (2.20) : concevoir la logique de station

La dernière étape avant la simulation est de faire la synchronisation du deuxième robot avec RAPID

```
23      !
24      ! pick and place program
25      !
26      ! Déplacer la piece dans le convoyeur
27      !
28      !*************************************************************
29      PROC main()
30          WaitDI start1,1;
31          WaitDI start1,0;
32          Path_10;
33
34      ENDPROC
35      PROC Path_10()
36          MoveL home2,v1000,fine,tGripper\WObj:=wobj0;
37          MoveL home3,v1000,fine,tool0\WObj:=wobj0;
38          MoveL pret1,v1000,fine,tool0\WObj:=wobj0;
39          MoveL pret2,v1000,fine,tool0\WObj:=wobj0;
40          MoveL pret3,v1000,fine,tool0\WObj:=wobj0;
41          WaitTime 16;
42
43      ENDPROC
44  ENDMODULE
```

Figure (2.21) : paragramme RAPID du troisième Robot

Après avoir fait la synchronisation avec RAPID maintenant on va démarrer la simulation de la cellule robotisée, il suffit juste d'y aller sur Simulation ► Lecture

Remarque : pour des raisons de sécurité on a ajouté des cages dans la station.

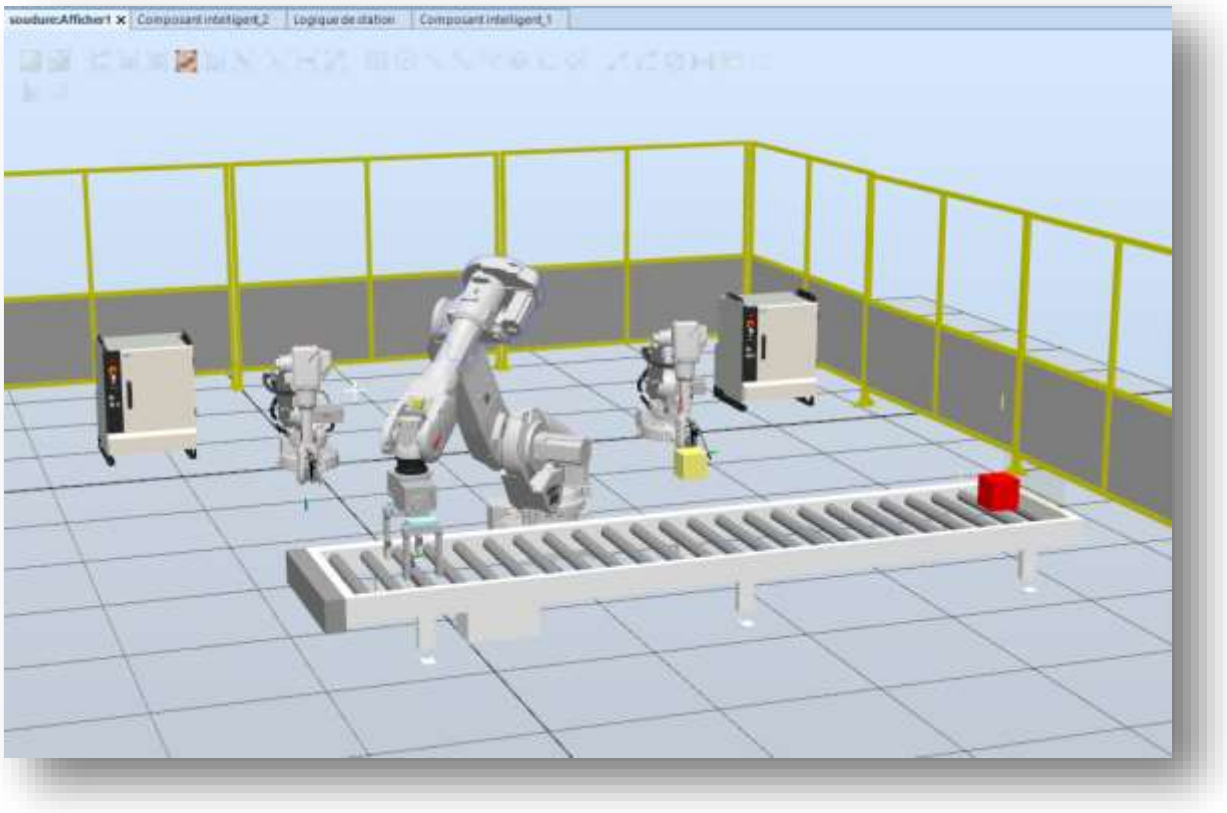


Figure (2.22) : démarrage de la simulation

Conclusion :

Après un apprentissage approfondi sur l'utilisation de ce logiciel et la liaison avec des différentes bases en automatisme et en informatiques, on a réussi à créer toute une cellule robotisé qui travaille en parallèle pour faire le soudage et le déplacement des pièces dans un convoyeur avec une rentabilité, précision et rapidité remarquable assuré par le logiciel ABB Robotstudio.

Conclusion générale

La robotique industrielle est définit comme étant un système commandé automatiquement, multi-applicatif, reprogrammable et manipulateur.

Les applications typiques incluent les Robots de soudage, de peinture, d'assemblage, les systèmes à convoyeurs etc.

L'intérêt de ce travail porte sur la planification et la programmation des tâches excitables de ramener la précision requise et la facilité d'exécution.

La première partie est dédié a donner une vision globale sur la robotique, l'historique des robots, les différentes types et leurs domaines d'application.

Ensuite dans la seconde partie on a démontré la modélisation mathématique pour la commande des robots, les modèles géométriques cinématiques et dynamique ainsi que tous les calculs nécessaires pour définir les positions et les paramètres des éléments du Robot.

Dans le troisième chapitre on intéresse sur la programmation graphique des taches avec le logiciel ABB robotstudio.

Enfin dans le quatrième et le dernier chapitre on a déterminé les différents étapes suivies pour arriver a la simulation d'une cellule robotisée avec le logiciel robotstudio qui offre a son utilisateur la facilité et la richesse de conception et de planification des taches habituellement utilisé dans l'industrie .

Grâce a ce modeste travail on a encaissé plusieurs connaissances a propos de la robotique, ce projet nous a permis d'enrichir nos propres compétences dans la simulation et le domaine industriel en général.

Bibliographie

[1] Introduction à la robotique – Hamdi Houcine – Univ Mentouri Constantine – 2002-2003

[2] Cours robotique – Jean Louis Boimond – Univ Angers

[3] Introduction à la robotique – Licence 2011-2012 – L.Matignon – Univ de Caen

[4] Univ Oran 2- IMSI – Modélisation et commande du bras manipulateur à cinq degrés de liberté robot moteur

[5] Cours robotique fondamental – Julien Alexandre – INRIA 20 janvier 2012

[6] Alain Liégeois. Modélisation et commande des robots manipulateurs. Techniques de l'ingénieur, page 3, Juin 2000.

[7] Summers'. 2005 "Robot Capability Test and Development of Industrial Robot Positioning System for the Aerospace Industry" *SAE Transactions*. 1108–1118.

[8] Sirinterlikci.M, Tiryakioglu.M, Bird.A, Harris.A, and Kweder.K. 2009 "Repeatability and Accuracy of an Industrial Robot: Laboratory Experience for a Design of Experiments Course," *Technology. Interface Journal.* 9 (2). ISSN 1523-9926

[9] Abele.E, Weigold.M, and Rothenbecher.S.2007. "Modeling and Identification of an Industrial Robot for Machining Applications," *CIRP Annals. - Manufacturing. Technology.* 56 (1): 387–390.

[10] Bases de la modélisation et des robots de la robots-manipulateurs de type série commande Wisama KHALIL, Etienne DOMBRE

[11] Modélisation et commande des robots manipulateurs Bernard BAYLE Télécom Physique Strasbourg

[12] Modélisation des Robots Master Automatique S2 rédigé par Mme S.BORSALI

[13] Sciavicco Lorenzo and Siciliano Bruno. Modelling and Control of Robot Manipulators.Springer-Verlag London, 1999.

[14] Les enjeux de l'IA dans la robotique – JITEC JOURNAL INNOVATIONS ET TECHNOLOGIES N°210 - Juillet/Août 2018

[15] Wisama. K, Dombre. E**,** 'Modélisation Identification et Commande desrobots', 2éme édition, Hermes Science Publications, Paris, 1999.

[16] Karim.A and Verl.A. "Challenges and obstacles in robot-machining," *Robotics (ISR), 44th International Symposium*. 1–4.Seoul, South Korea. 2013

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

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

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

[20]Khaled Arrouk. Techniques de conception assistée par ordinateur (CAO) pour lacaractérisation de l'espace de travail de robots manipulateurs parallèles. Autre. UniversitéBlaise Pascal - Clermont- Ferrand II, 2012.

[21]Ilian Bonev, ing. Yanick Noiseux, ing. 21 septembre 2014

[22]Manuel d'utilisation, Site officiel de l'entreprise ABB « abb.com/robotics »