الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne Démocratique et Populaire
وزارة التعليم العالي والبحث العلمي
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Département : Génie électrique Et Informatique Industrielle

Mémoire de projet de fin d'études

Pour l'obtention du diplôme d'ingénieur d'état en Automatique

Spécialité
Automatique Et Informatique Industrielle

Amélioration d'un processus de finition de pièces mécaniques à base du robot ABB-IRB -6700

Réalisé par

Aboubakeur BENBOUZID ***Mohamed Amine RAHMANI***

Présent(e) et soutenue publiquement le (06 / 07 /2022)

Composition du Jury :

Président	M. Amar REZOUG	M.C.A	ENST
Promoteur	M .Samir KHELOUAT	M.C.B	ENST
Promoteur	M .Mohamed Amine MAMOU	M.C.B	USTHB
Examinateur	Mme. Zahira OUSAADI	M.A.A	ENST
Examinateur	M .Nait Slimani BOUKHALFA	M.A.A	ENST

REMERCIEMENTS

Nous remercions **ALLAH** qui nous a donné la force et la volonté d'accomplir ce travail.

La réalisation de ce mémoire a été possible grâce à **ALLAH** en premier lieu, et au concours de plusieurs personnes à qui nous voudrions témoigner toutes nos reconnaissances.

Nous voudrions tout d'abord adresser toute nos gratitudes à nos promoteurs de ce mémoire, **M .Samir KHELOUAT** et **M .Mohamed Amine MAMOU** pour leur patience, leur disponibilité et surtout leur judicieux conseils.

Nous tenons également à exprimer notre immense gratitude envers tous les membres du jury, qui ont accepté d'évaluer ce travail.

Le travail sur ce mémoire durant toute une année n'aurait pas été aussi fructueux sans l'aide et le courage précieux de **nos parents** et **notre famille**, ainsi que de **mes amis**
Pour cela, nous les remercions tous pour leur soutien moral, intellectuelle, douaa et le courage qu'ils nous ont donné.

Enfin, nous adressons nos sincères remerciements à tous les professeurs, intervenants et toutes les personnes qui par leurs paroles, leurs écrits, leurs conseils et leurs critiques ont guidé notre chemin et ont accepté à nous rencontrer et répondre à nos questions durant à tout moment possible.
À tous ces intervenants, nous présentons nos remerciements, notre respect et nos gratitudes.

SOMMAIRE

REMERCIEMENTS ..... **I**

SOMMAIRE ..... **II**

LISTE DES FIGURES ..... **III**

LISTE DES TABLEAUX ..... **V**

INTRODUCTION GENERALE **1**

I. CHAPITRE I : DESCRIPTION DE LA CELLULE ROBOTISÉE **3**

I.1 INTRODUCTION 4
I.3 PRÉSENTATION DE LA CELLULE 5
I.4 DESCRIPTION DU MATÉRIELS DE CONDUITE 7
I.5 DESCRIPTION DES CYCLES DE FINITION 11
I.6 CONSIGNES DE SECURITE 12
I.7 ENTRETIEN ET REPARATION 12
CONCLUSION 13

II. CHAPITRE II : MODÉLISATION GÉOMÉTRIQUE DU ROBOT MANIPULATEUR ABB-IRB 6700 ..... **14**

II.1 INTRODUCTION 15
II.2 DESCRIPTION DU ROBOT ABB-IRB 6700 15
II.3 MODÈLE GÉOMÉTRIQUE DIRECT 15
II.4 MODÈLE GÉOMÉTRIQUE INVERSE 18
CONCLUSION 21

III. CHAPITRE III : *MODÉLISATION, CONTRÔLE ET SUPERVISION DE LA STATION ROBOTISÉE* **22**

III.1 INTRODUCTION 23
III.2 PRÉSENTATION DE ROBOT-STUDIO 23
III.3 LA CELLULE DU ROBOT ABB-IRB 6700 (MODÈLE SOUS ROBOTSTUDIO) 31
III.4 CONTRÔLE ET SUPERVISION DE LA STATION ROBOTISÉE 33
III.5 IDENTIFICATION ET CREATION DES TACHES DE FINITION 47
CONCLUSION 49

IV. *CHAPITRE IV : EXTRACTION DE LA TRAJECTOIRE D'ÉBAVURAGE* **50**

IV.1 INTRODUCTION 51
IV.2 GÉNÉRALITÉS 51
VI.3 PROGRAMME DE TRAITEMENT D'IMAGE 52
IV.4 L'ENVOI DE DONNÉES VERS LE ROBOT 58
IV.5 SIMULATION 59
CONCLUSION : 60

CONCLUSION GENERALE **61**

BIBLIOGRAPHIE **63**

ANNEXE ..... **I**

LISTE DES FIGURES

FIGURE I-1: EXEMPLE DE DRESS PACK D'UN ROBOT MANIPULATEUR [1] 5
FIGURE I-2 : PLAN D'ENSEMBLE DE LA CELLULE ROBOTISEE. 6
FIGURE I-3 : LA TABLE TOURNANTE 7
FIGURE I-4 : POSTE DE FINITION 7
FIGURE I-5: ARMOIRE ELECTRIQUE 8
FIGURE I-6: SCHEMA DE COMMANDE ET D'ALIMENTATION DE VFD M300 [4] 9
FIGURE I-7 : LA CONVERSION ENTRE DEUX APPAREILLES DE MEME /DIFFERENT RESEAU AVEC UN X-GATEWAY 9
FIGURE I-8: FLEX-PENDANT 10
FIGURE I-9: ABB IRC5 10
FIGURE I-10: TOPOLOGIE DE RESEAU 11
FIGURE II-1: L'ANATOMIE DE POSITIONS ET DES ARTICULATIONS DU ROBOT ABB- IRB 6700 15
FIGURE III-1: LA BARRE D'OUTILS DU STUDIO-ROBOT 26
FIGURE III-2: CREATION D'UNE STATION 31
FIGURE III-3: CREATION DE LA STRUCTURE CELLULAIRE 31
FIGURE III-4: LA CELLULE ROBOTISEE COMPLETE CREEE SOUS ROBOT STUDIO 32
FIGURE III-5: LES CONSTITUONS DE NOTRE PROJET 34
FIGURE III-6: ILLUSTRATION DE LA CONFIGURATION MATERIELLE CHOISIE 35
FIGURE III-7 : ILLUSTRATION DE LA CONFIGURATION RÉSEAU 36
FIGURE III-8: ILLUSTRATION DE L'INTERFACE DU ROBOT AU NIVEAU DU TIA 36
FIGURE III-9: TABLE DE VARIABLE DES ENTREES 37
FIGURE III-10 : TABLE DE VARIABLE DES SORTIES 37
FIGURE III-11: ILLUSTRATION DES BLOCS CREES 39
FIGURE III-12: SCHEMA REPRESENTATIF DES APPELS DES BLOCS ET LEURS INTERACTIONS 39
FIGURE III-13: PARTIE DE MISE EN ŒUVRE DES OUTILS 41
FIGURE III-14: PROGRAMME DE L'ENVOI DE LA CONSIGNE DE L'OUTIL (VARIATEUR) 3 41
FIGURE III-15: PROGRAMME DE DETECTION DE DEFAUT DE LA TABLE. 42
FIGURE III-16: ILLUSTRATION DE LA NAVIGATION ENTRE LES VUES 42
FIGURE III-17: ILLUSTRATION DE LA NAVIGATION ENTRE LES VUES 43
FIGURE III-18: VUE DES ETATS DES OUTILS 43
FIGURE III-19: VUE DES CYCLES DU ROBOT 44
FIGURE III-20: VUES DE COMMANDE 44
FIGURE III-21: VUE DES ALARMES 45
FIGURE III-22: METHODES D'EXECUTION DES TACHES 47
FIGURE III-23 ORGANIGRAMME DE FONCTIONNEMENT DE LA CELLULE. 49
FIGURE IV-1 : PRINCIPE DE FONCTIONNEMENT D'UN SCANNER3D [12] 51
FIGURE IV-2FORMAT STL 52
FIGURE IV-3:REPRESENTATION DE PCL DANS L'ESPACE 52
FIGURE IV-4 : LES PIÈCES À TRAITER (1, 2,3) 53
FIGURE IV-5 : PROTOTYPAGE DE LA SEGMENTATION 54
FIGURE IV-6: VALIDATION D'UN POINT 54
FIGURE IV-7: RÉSULTATS OBTENUS SUR LES PIÈCES DE TEST 55
FIGURE IV-8: BRUIT 55
FIGURE IV-9: DONNÉE 55
FIGURE IV-10: LES RÉSULTATS DU FILTRAGE 55
FIGURE IV-11: MÉTHODE DE LA RÉGRESSION LINÉAIRE [13] 56
FIGURE IV-12: RÉSULTATS DE L'EMPLACEMENT DE L'ADDITION SUR LA PIÈCE DE RÉFÉRENCE 57
FIGURE IV-13 : LA MATRICE RETOURNER PAR LA FONCTION GET-REAL 57
FIGURE IV-14: STRUCTURE DE PROGRAMME DE COMMUNICATION (MATLAB/ROBOT-STUDIO) 58

FIGURE IV-15: LA RÉALISATION D'UNE COMMUNICATION ETHERNET ENTRE PC1 ET PC2 59
FIGURE IV-16: LE CONTOUR DE FINITION DE LA PIÈCE 1 EN MM. 59
FIGURE IV-17: TRAJECTOIRE FINALE RESULTANTE 60

LISTE DES TABLEAUX

TABLEAU I-1: LES ELEMENTS DE L'INTERFACE D'APPLICATION D'IRC5 10
TABLEAU I-2: LISTE DES DISPOSITIFS DE SECURITE 12
TABLEAU II-1: LA TABLE DE DH DU ROBOT 17
TABLEAU II-2: TABLE D'ESSAI 20
TABLEAU III-1: DESCRIPTION DES ONGLETS DE LA BARRE D'OUTILS. 26
TABLEAU III-2: DESCRIPTION DES ELEMENTS DU NAVIGATEUR D'AGENCEMENT 27
TABLEAU III-3: DESCRIPTION DES ELEMENTS DE NAVIGATEUR TRAJECTOIRE ET CIBLE. 28
TABLEAU III-4: DESCRIPTION DES ELEMENTS DU NAVIGATEUR SYSTEME DE COMMANDE. 29
TABLEAU III-5: BILAN DES ENTREES SORTIES PHYSIQUE DE L'API. 34
TABLEAU III-6: BILAN DES ENTREES SORTIES PHYSIQUE DE L'API. 34
TABLEAU III-7: L'ADRESSAGE DE LPI 35
TABLEAU III-8: BITS D'ENTREES DE L'ECHANGE ROBOT/API. 40
TABLEAU III-9: BITS DE SORTIES DE L'ECHANGE ROBOT/API. 40
TABLEAU III-10: SIGNIFICATION DES BITS D'ENTREES ET SORTIES. 40
TABLEAU IV-1LES DONNEES DE LA PIECE (X, Y, Z, $\vec{N}$) EXCRETES DE LA FORME PCL 53
TABLEAU IV-2: LA DISTRIBUTION DES POSITIONS (X, Y, Z) DE (IM1, IM2) 53

INTRODUCTION GENERALE

La plus part des procédés de fabrication de pièces mécanique génèrent des résidus de matière accrochée à la pièce. Ces résidus sont connus sous le nom de bavures, et l'opération qui consiste de minimiser ces bavures est appelée ébavurage. Cette opération d'ébavurage est inévitable lors de la fabrication de pièces mécaniques. Les bavures peuvent être minimisées, mais pas éliminées. Or, l'ébavurage est trop souvent négligé ou peu d'effort sont mis pour l'améliorer. Par contre, pour certains domaines de haute technologie comme l'automobile ou l'aéronautique, les pièces fabriquées possèdent une grande valeur lorsqu'elles arrivent à l'étape d'ébavurage. Il y a alors un grand intérêt à améliorer cette opération pour assurer la qualité du produit jusqu'à la dernière opération, surtout lorsqu'il s'agit d'ébavurage de précision. Dans ce mémoire, la solution retenue pour l'automatisation de l'ébavurage est la robotisation.

L'utilisation des robots sériels pour des opérations d'ébavurage est devenue un enjeu de développement des robots ces dernières années. L'objectif est de proposer une alternative aux machines à outils. Ainsi, des robots gros porteurs 6 axes sont proposés par les fabricants et concepteurs de robots. Toutefois, les robots développés présentent des précisions de suivi de trajectoire uniquement de l'ordre du dixième de millimètre, même si leur rigidité a été améliorée par rapport aux robots utilisés pour des opérations moins exigeantes. En effet, les comportements géométrique, statique et dynamique de ces robots ne permettent pas de réaliser des finitions de pièces mécaniques avec une grande précision, en générale les taches effectuées sont des taches de semi finition.

Dans ce contexte, le département Fonderie de la Société Nationale de Véhicules Industriels SNVI, possède une cellule robotisée à base du robot ABB-IRB 6700, destinée à l'ébavurage de plusieurs types de pièces mécanique. Au sein de ce département que nous avons effectué notre stage. Le travail qui nous a été proposé est d'améliorer le processus de finition de pièces mécaniques. En effet, la programmation du robot pour l'opération d'ébavurage s'effectue par l'enregistrement point par point de la trajectoire des bavures, ceci rend le travail très pénible, et la tâche d'ébavurage ne se réalise pas correctement qu'après plusieurs essaies sur la même pièce à cause de l'imprécision de choix des points de la trajectoire des bavures.

Le choix de type de robot pour ce genre d'application se base sur deux critères importants, flexibilité et adaptabilité [1], qui représentent deux demandes fortes des utilisateurs de robots. Pour y répondre, le robot ABB-IRB 6700, s'inscrit dans cette ligne en affichant des meilleures performances dans le segment des 150 à 235 kg. Plus robuste, avec des intervalles d'entretien espacés et une maintenance simplifiée, son cout de possession est 20 % inférieur à celui de son prédécesseur, l'IRB 6640. Cette nouvelle génération de robot comporte un système de commande IRC5 et du logiciel de commande du robot, RobotWare qui prend en charge tous les aspects du système de robot, notamment le contrôle des mouvements, le développement et l'exécution des programmes applicatifs et la communication.

Au fil du temps, la programmation des robots hors ligne s'est imposé comme le moyen le plus simple, le plus rapide, le plus précis et le plus souple pour créer des programmes robot avant même de toucher tout matériel physique dans le monde réel. Cette programmation hors ligne maximise le retour sur investissement des systèmes robotiques en réduisant les couts, accélère

la mise sur le marché des produits et améliore leur qualité. Pour ces fins, ABB a conçu l'environnement « *RobotStudio »* qui permet de valider une solution robotique avant même de la réaliser ou sans immobiliser la production. Il facilite la programmation des robots et la conception des solutions pour des environnements industriels complexes. Le RobotStudio se distingue par le modèle virtuel, en reproduisant exactement le programme exécuté sur les robots ABB en production. En exploitant des programmes robot réels et des fichiers de configuration identiques à ceux utilisés en atelier, l'utilisateur a l'assurance que tous les aspects du monde virtuel fonctionneront exactement comme dans la réalité.

Le mémoire est constitué de quatre chapitres :

Le premier chapitre consiste à l'étude de la cellule robotisée de la SNVI à base du robot ABB-IRB 6700. La compréhension du fonctionnement des différents constituants de la cellule est d'une grande utilité. Nous abordons le principe de fonctionnement de la cellule en décrivant le robot et ses caractéristiques, la description du cycle de fonctionnement, la partie automatisme gérée par un automate et la communication entre les différents constituants de la cellule.

Le deuxième chapitre est consacré au calcul du modèle géométrique du robot ABB-IRB 6700. En utilisant le formalisme de Denavit-Hartenberg modifié, nous calculons le modèle géométrique directe du robot. Quant au modèle géométrique inverse est obtenu en utilisant la méthode de Paul.

Le troisième chapitre de ce mémoire est constitué de deux parties. La première partie est consacrée au développement de l'environnement de la cellule robotisée sous le logiciel « RobotStudio » afin de valider certaine poursuite de trajectoire programmée en langage Rapid sous RobotStudio. Cependant, la deuxième partie de ce chapitre est consacrée à la partie automatisme du système, ou nous nous intéressons au contrôle et à la supervision de la station robotisée.

En quatrième chapitre nous détaillons notre solution qui permet au robot de suivre la trajectoire des bavures sur la pièce mécanique. Nous proposons une solution qui consiste à faire un traitement de la pièce avec bavures sous Matlab, le résultat du traitement est traduit par un ensemble de points dans l'espace opérationnel ainsi que l'orientation de l'effecteur pour une bonne poursuite de la trajectoire des bavures sur la pièce.

Enfin, des remarques finales et des suggestions d'idées de recherche futures sont discutées en conclusion générale.

I. Chapitre I : Description de la cellule robotisée

I.1 INTRODUCTION

Afin de réduire le cout de fabrication des pièces, l'industrie s'oriente vers la technique de moulage vis-à-vis de l'utilisation de la machine numérique comme CNC.... Mais le problème majeur des techniques de moulage réside dans la pénétration de la fonte liquide dans les joints de moule ou du noyau, créant ainsi des bavures. L'ébavurage consiste donc à éliminer ces imperfections dans le but de donner à la pièce un aspect désiré. Plusieurs solutions peuvent être envisagées, parmi elles, l'utilisation des robots industriels. Pour cela, la cellule robotisée de la SNVI, ou notre stage s'est effectué, est destinée à cette tâche d'ébavurage sur plusieurs types de pièces mécaniques, en utilisant le robot ABB-IRB 6700. Ce chapitre est consacré à la description de cette cellule robotisée afin de proposer par la suite des solutions à l'opération effectuée par les robots.

I.2 GÉNÉRALITÉS SUR LE ROBOT ABB-IRB 6700

La série IRB 6700 représente la 7ème génération de robots industriels à charge utile élevée et à haute performance d'ABB Robotics. Issue de la série IRB 6640, avec un large rayon d'action, un couple poignet très élevé, une construction modulaire conviviale pour la maintenance et un taux élevé de disponibilité. Cette catégorie de robot possède des propriétés et caractéristiques très concurrentielles, elle se distingue par une capacité de production élevée, une conception compacte et un poids faible, un entretien simple et des coûts de maintenance réduits, le robot IRB 6700 est idéal pour les applications de processus, quel que soit le domaine industriel. Les domaines habituels sont, par exemple, la manutention de matériaux, la conduite de machine, le soudage par points. [1]

I.2.1 Système d'exploitation

Cette nouvelle génération de robot comporte un système de commande IRC5 (Robot-Ware), il prend en charge tous les aspects du système de robot, notamment le contrôle des mouvements, le développement et l'exécution des programmes applicatifs et la communication. [2].

I.2.2 Fonctionnalités complémentaires

En ce qui concerne les fonctionnalités supplémentaires, le robot peut être équipé d'un logiciel optionnel d'applications (comme l'encollage et le soudage), de fonctions de communication (*communication* réseau) et de fonctions avancées (fonctionnement multitâche, contrôle par capteur).

I.2.3 Dress-Pack

Comprend des options pour le bras supérieur, le bras inférieur et le socle dans les positions B, C et D. « reportez-vous à la Figure (I-1) », Ces éléments sont :

- Le système Dress-Pack de la base du robot comprend des signaux client.
- Le Dress-Pack pour bras inférieur contient des paquets de câbles de processus comprenant des signaux, destinés au client.
- Solutions DressPack adaptées aux différents besoins des utilisateurs.

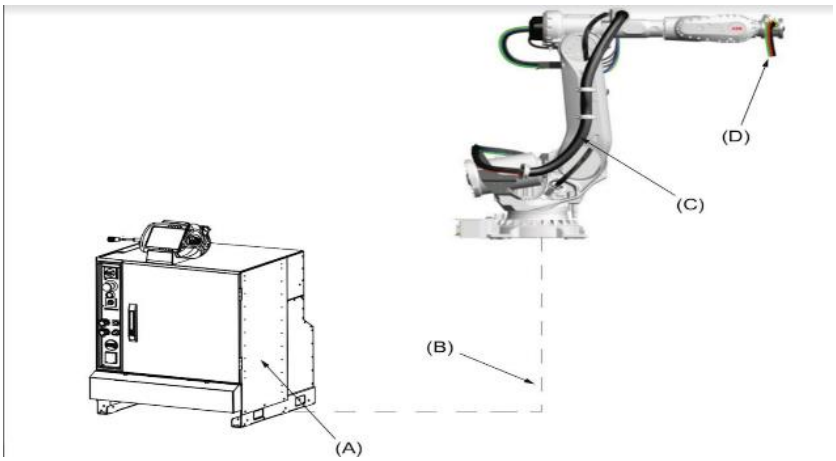


Figure I-1: Exemple de Dress Pack d'un robot manipulateur [1]

Les différents types de robot peuvent recevoir un ensemble de câbles et de gaines intégrés dans les options Dress-Pack.

I.2.5 Méthode Axis Calibration

Axis Calibration est une méthode d'étalonnage standard pour l'étalonnage de IRB 6700. C'est la méthode recommandée pour obtenir des performances correctes. Les routines suivantes sont disponibles pour la méthode Axis Calibration [1] :

- Étalonnage précis.
- Mise à jour des compte-tours.
- Référence Calibration.

L'équipement d'étalonnage d'Axis Calibration est fourni sous la forme d'un jeu d'outils. On peut trouver les instructions relatives à l'exécution de la procédure d'étalonnage sur le Flex-Pendant. Il guide l'opérateur étape par étape tout au long de la procédure d'étalonnage.

I.3 PRÉSENTATION DE LA CELLULE

I.3.1 Fonctionnement de l'installation

L'installation a pour but d'effectuer l'ébavurage des plans de joints. Qu'est-ce que l'**ébavurage** ? L'ébavurage permet d'éliminer le surplus de métal indésirable qui reste accroché à la pièce lorsqu'elle est moulée.

L'installation de la cellule de finition comprend : « voir la figure (I-2) »

- Une table tournante qui sert au chargement de la pièce mécanique par l'opérateur protégé par une barrière immatérielle.
- Une cabine insonorisée.
- Un robot ABB IRB 6700 – 235Kg / 2.65m - N° : 6700-102437.
- Un changeur d'outil monté sur le robot.
- Un poste de finition composé :
  - d'une unité de sciage Ø500mm, 22 kW - 2945 tr/mn.
  - d'une unité de meulage Ø500mm, 22 kW - 2945 tr/mn.

- d'une unité de fraisage diamanté Ø50mm, 13.2 kW - 5800 tr/mn.
- Une table d'évacuation.

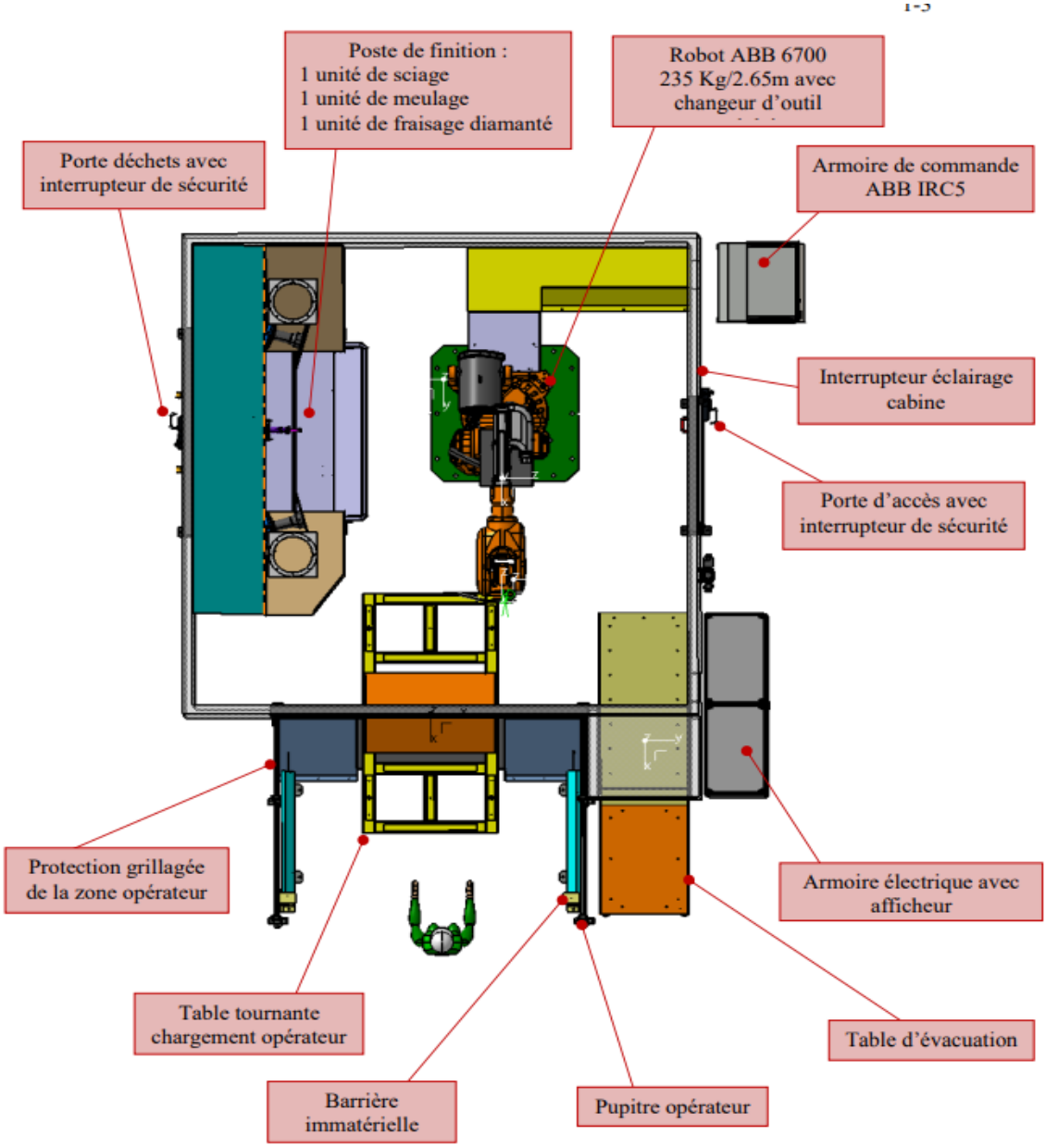


Figure I-2 : Plan d'ensemble de la cellule robotisée.

I.3.2 Description du fonctionnement

La suite de chapitre est consacrée à la description de chaque élément de cellule robotisée afin de bien comprendre le principe de fonctionnement.

a- Poste de chargement

Ce poste est constitué d'une table tournante de chargement entrainée en rotation par un indexeur. Une fois la table est chargée de pièce (300kg maximum de chaque côté). Le détecteur valide, la table transmet la pièce à l'intérieur à fin que le robot se saisit la pièce pour la transférer vers le poste d'ébavurage. La figure (I-3) montre la structure de la table tournante.

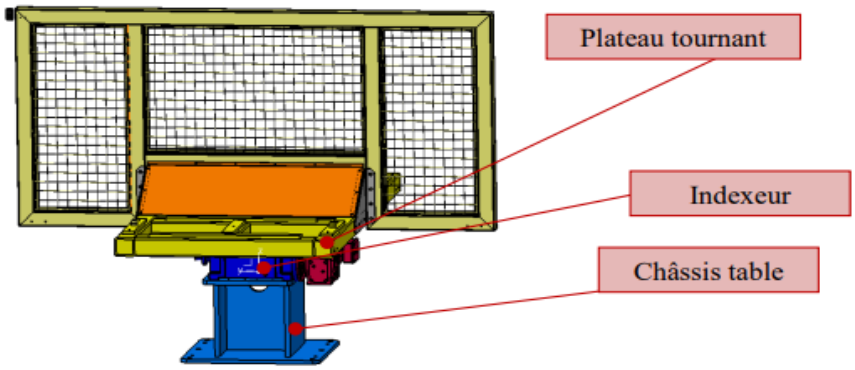


Figure I-3 : La table tournante

b- Poste de finition

Le poste de finition est constitué de deux unités de sciage et de meulage entrainé par un moteur de 22Kw et une électro-broche actionnant le fraisage diamanté, ainsi qu'une trémie placée sous le poste pour la récupération des déchets, comme montre la figure (I-4).

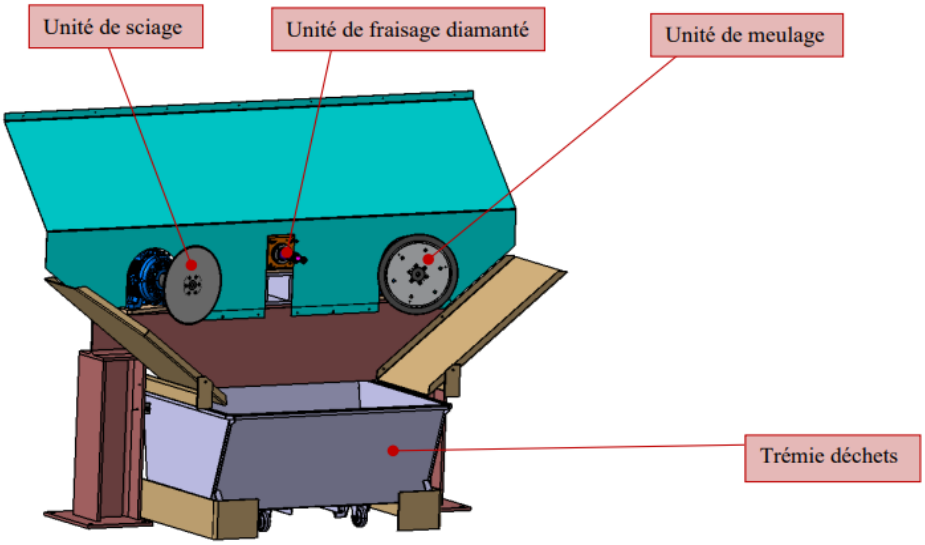


Figure I-4 : Poste de finition

c- Interface Robot

Le robot est équipé d'une interface permettant l'accouplement automatique des broches. Cette interface est composée du système de couplage, d'un détecteur optoélectronique permettant la détection de pièce et le recalage du robot, et d'un système de soufflage.

I.4 DESCRIPTION DU MATÉRIELS DE CONDUITE

I.4.1 Armoire électrique générale

Elle contient des équipements électriques de puissance, alimentation électrique, les transformateurs et les disjoncteurs…, ainsi que la partie commande, La figure (I-7) montre la structure de l'armoire électrique générale.

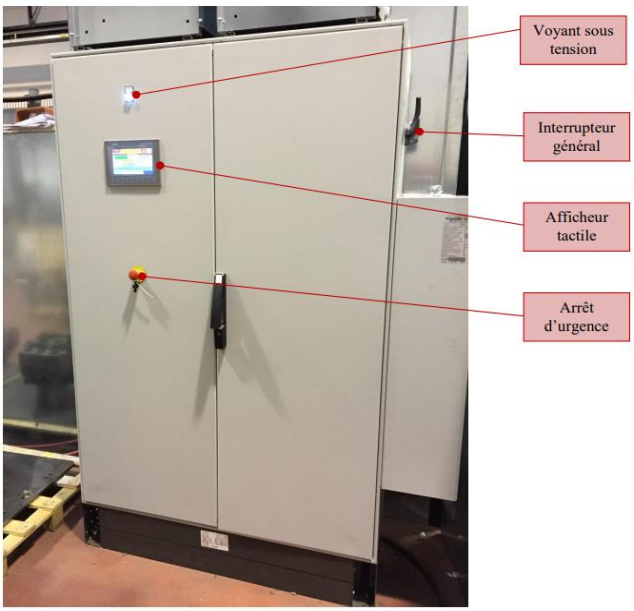


Figure I-5: Armoire électrique

Dans la partie commande on trouve :

a- Détecteurs de zéro speed

Les modules de sécurité Preventa XPSVNE pour la détection de vitesse zéro sont utilisés pour détecter l'état d'arrêt des moteurs électriques. Leurs applications les plus courantes comprennent : fournir le signal de déverrouillage pour les protections de machine coulissantes ou amovibles électriquement verrouillées, contrôler les signaux de direction de rotation pour les moteurs de marche arrière et engager les freins de verrouillage après qu'un moteur est à l'arrêt [3].

b- L'automate (PLC) S7-1200

La famille S7-1200 de Siemens est constituée d'automates programmables (API) utilisables dans des applications d'automatisation variées. Sa forme compacte, son faible prix et son important jeu d'instructions en font une solution idéale pour la commande d'applications très variées. Les modèles S7-1200 ainsi que l'outil de programmation STEP 7 se basant sur Windows. La CPU combine un microprocesseur, une alimentation intégrée, des circuits d'entrée et de sortie, un PROFINET intégré, des E/S rapides de commande de mouvement, ainsi que des entrées analogiques intégrées dans un boîtier compact en vue de créer un contrôleur puissant. Les modules d'entrées-sorties (SM) permettent d'ajouter des fonctionnalités à la CPU. Les SM se raccordent sur le côté droit de la CPU [6].

c- Les VFD (Variable Frequency Drive) Emerson (unidrive_m300)

Uni-drive M300 est idéal pour les applications qui nécessitent une intégration rentable dans les systèmes de sécurité et de contrôle du moteur en boucle ouverte RFC-A avancé. M300 fournit un chemin de mise à niveau pour les utilisateurs de Commander SK qui veulent un ensemble de fonctionnalités similaires avec l'ajout de deux entrées STO intégrées pour fournir la conformité SIL3/PLe et le contrôle du moteur en boucle ouverte RFC-A amélioré [4].

La figure (I-5) montre le schéma de commande et d'alimentation de VFD M300.

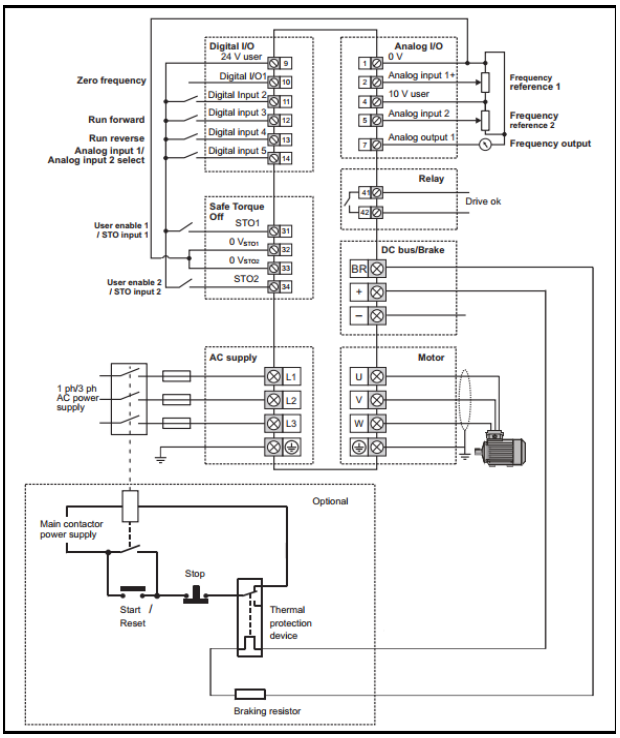


Figure I-6: schéma de commande et d'alimentation de VFD M300 [4]

d- La passerelle HMS (X_Getway)

Anybus « X-Gateway » permet à deux réseaux différents de se communiquer. En termes simples, on pourrait dire qu'il s'agit d'un traducteur en temps réel entre deux réseaux (jouer le rôle d'un convertisseur) « voir Figure (I-6) ». Les passerelles résolvent d'importants problèmes de communication industrielle pour les intégrateurs de systèmes travaillant avec la conception de réseaux industriels et offrent un moyen rapide et facile de connecter deux réseau autrement incompatibles, donc la passerelle X-Gateway Anybus permet de connecter/étendre les systèmes de contrôle PLC et le contrôleur IRC5 entre deux réseaux PROFINET de déférant protocole [6].

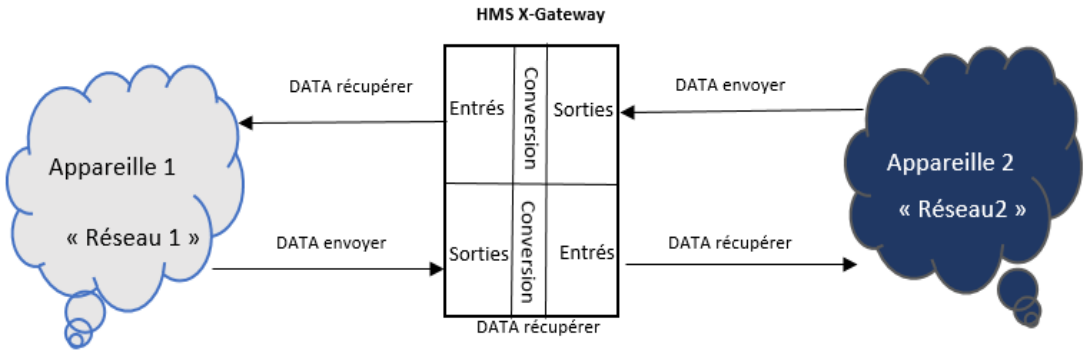


Figure I-7 : la conversion entre deux appareilles de même /différent réseau avec un X-gateway

e- Switch

Un switch désigne un commutateur réseau, équipement ou appareil qui permet l'interconnexion des appareils communicants.

f- HMI

Une interface homme-machine (IHM), est une interface utilisateur ou un tableau de bord qui connecte une personne à une machine.

I.4.2 Flex-Pendant Commandes principales

Le Flex-Pendant est fourni par ABB il est équipe d'un écran tactile serte à commander le robot, « Voir Figure (I-8) ».



Figure I-8: Flex-Pendant

I.4.3 Commande IRC5 (Fourniture ABB)

L'ABB IRC5 est un système de commande robotique de cinquième génération qui combine le contrôle des mouvements, flexibilité, modularité, convivialité, sécurité et interfaces d'application. La Figure I-9 et le Tableau I-1 montrent les interfaces d'applications d'ABB IRC5.

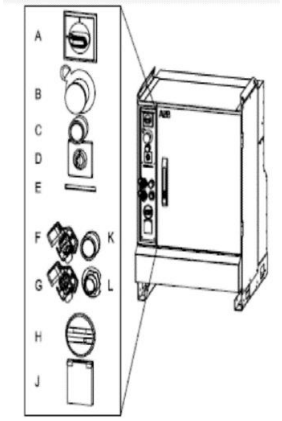


Figure I-9: ABB IRC5

Tableau I-1: Les éléments de l'interface d'application d'IRC5

A	Interrupteur secteur
B	Arrêt d'urgence
C	Marche Moteur
D	Interrupteur de mode
E	Voyant de chaîne de sécurité (option)
F	Port USB
G	Port de service PC (option)
H	Compteur horaire (option)
J	Prise de courant de service (option)
K	Hot plug bouton (option)
L	Connecteur FlexPendant

I.4.4Synoptique du réseau

La communication est assurée par le protocole PROFINET, la figure (I-10) montre la topologie de notre réseau.

PROFINET est une solution Ethernet industrielle ouverte basée sur les normes internationales. Il s'agit d'un protocole de communication conçu pour échanger des données

entre les contrôleurs et les appareils dans un cadre d'automatisation. Il a été introduit au début des années 2000 et est la solution Ethernet industrielle la mieux adoptée, des centaines de fabricants ont développé des produits PROFINET, tels que PLC, PAC, Lecteurs, Robots, OI, et des outils de diagnostic [7].

Pour assurer une performance appropriée, PROFINET fournit des données par les canaux de communication suivants :

- TCP/IP (ou UDP/IP).
- PROFINET Temps réel (RT).
- PROFINET Isochronous Real-Time (IRT).
- Mise en réseau urgente (TSN).

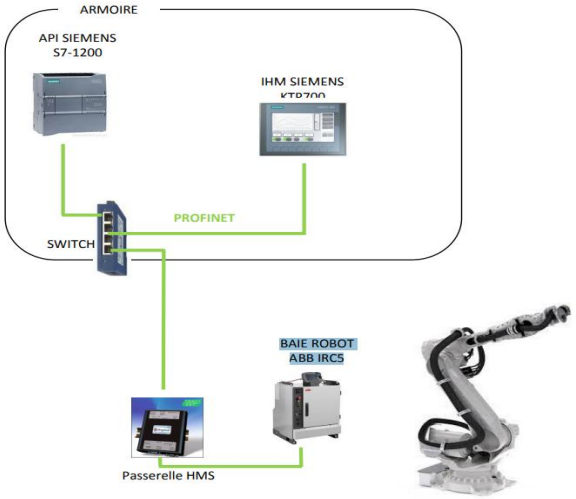


Figure I-10: Topologie de réseau

I.5 DESCRIPTION DES CYCLES DE FINITION

La phase de finition de la pièce mécanique s'effectue suivant les cycles décrits ci-dessous :

1- Chargement pièce sur la table tournante par l'opérateur
- Autorisation de transfert (verrine verte).
- Appui sur bouton de validation (départ cycle).
- Rotation de la table pour prise robot.
- Table 1 en position (capteur).

2- Départ cycle 1
- Vérification présence pièce.
- Prise pièce par robot.
- Fin de cycle 1.

3- Départ cycle 2
- Transfert pièce au tronçonnage.
- Tronçonnage.
- Fin de cycle 2.

4- Départ cycle 3
- Transfert pièce au meulage.
- Meulage.
- Fin de cycle 3.

5- **Départ cycle 4**
   - Transfert pièce au fraisage.
   - Fraisage.
   - Fin de cycle 4.

6- **Départ cycle 5**
   - Transfert pièce vers table d'évacuation.
   - Dépose pièce.
   - Fin de cycle 5.

I.6 CONSIGNES DE SECURITE

Toute intervention sur la cellule doit s'effectuer à :

- L'arrêt et avec verrouillage mécanique du sectionneur de l'armoire électrique.
- Le coup de poing d'arrêt d'urgence devra être libre en cas d'incident ou d'accident, l'opérateur agit le plus rapidement possible.
- L'arrêt instantané des mouvements peut provoquer des dégâts matériels donc il est recommandé de ne pas utiliser l'arrêt d'urgence sauf dans des problèmes spécifiques causés d'un fonctionnement anormal.
- Il est strictement interdit de bloqué les commande avec n'importe quelle moyenne.
- En mode manuel, le dispositif de sécurité n'est prévu que pour un seul opérateur.
- L'opérateur doit porter les équipements de sécurité :
  - Des chaussures de sécurité.
  - Des vêtements de sécurité incendient.
  - Des gants de manutention résistant aux huiles de coupe.
  - Une paire de lunettes de protection.
  - Un casque de chantier.
  - Ne jamais tenter de se placer sous une pièce levée par le robot.

I.6.1 Liste des dispositifs de sécurité

La liste des dispositifs de sécurité est représentée sur le tableau suivant avec leurs traitements électrique et programmé.

Tableau I-2: liste des dispositifs de sécurité

Dispositif de sécurité	Traitement électrique	Traitement programmé
Arrêt d'urgence FlexPendant (PMA)	Coupure en logique câblée de la puissance robot. Coupure en logique câblée de la commande broches et table. Coupure des sorties robot.	Affichage d'un message sur le FlexPendant (PMA) et sur le pupitre de commande
Arrêt d'urgence baie robot		
Arrêt d'urgence Armoire électrique		
Arrêt d'urgence porte		
Porte d'accès	Coupure en logique câblée de la puissance robot **en mode automatique**.	Affichage d'un message sur le FlexPendant (PMA) et sur le pupitre de commande

I.7 ENTRETIEN ET REPARATION

Avant tout intervention d'entretien ou de réparation il faut que certaines conditions soient réalisées :

- Retour des ensembles mobiles au repos.
- Protection mécanique condamnant les dispositifs de mise en marche des moteurs.

CONCLUSION

Ce chapitre est consacré à la description de la cellule robotisée installée au niveau de l'entreprise SNVI de Rouiba. L'ébavurage des pièces mécanique est effectué par des robots de type ABB-IRB 6700. Une description fonctionnelle et matérielle a été détaillée au cours de ce chapitre, afin de bien cerner la tâche principale à effectuer le long de ce mémoire.

II. Chapitre II : Modélisation géométrique du robot manipulateur ABB-IRB 6700

II.1 INTRODUCTION

La modélisation des robots manipulateurs est une étape primordiale pour l'étude et la synthèse d'une loi de commande. En effet, pour commander ou simuler le comportement d'un robot, on doit disposer d'un modèle. Plusieurs niveaux de modélisation sont possibles selon les objectifs, les contraintes de la tâche et les performances recherchées. On distingue trois classes des modèles existant :

- Le modèle géométrique direct et inverse, qui donne la position cartésienne de l'effecteur en fonction des variables articulaires du mécanisme et vice versa.
- Le modèle cinématique direct et inverse, qui exprime la vitesse opérationnelle de l'effecteur en fonction des vitesses des articulations et vice versa.
- Le modèle dynamique qui décrit la relation entre les couples et forces, en fonction de la position, vitesse et accélération articulaires.

Ce chapitre concerne l'étude et développement du modèle géométrique direct et inverse du robot ABB-IRB 6700.

II.2 DESCRIPTION DU ROBOT ABB-IRB 6700

Le robot ABB-IRB 6700 est constitué de six articulations rotoïdes, comme le représente la figure (II-1), qui montre l'anatomie et la position des articulations du robot, où les parties de couleur contrastée désignent respectivement les axes 1,3 et 5 et les parties de couleur blanche désignent respectivement les axes 2, 4 et 6 *[1]*.

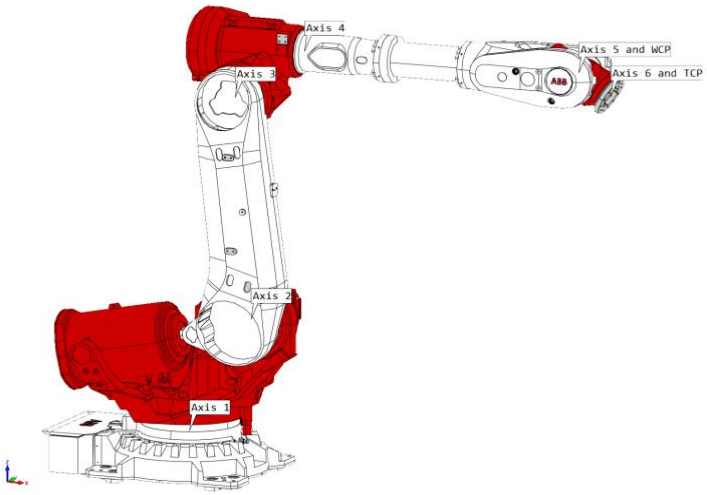


Figure II-1: l'anatomie de Positions et des articulations du robot ABB- IRB 6700

La cellule de finition de la SNVI, sur laquelle porte notre étude comprend un robot de type ABB IRB 6700 de capacité de manutention de 235Kg et de portée de 2.65m.

II.3 MODÈLE GÉOMÉTRIQUE DIRECT

Le modèle géométrique direct donne la position et l'orientation de l'effecteur en fonction des variables articulaires.

$$\boldsymbol{X} = \boldsymbol{f}(\boldsymbol{\theta}) \qquad (II.1)$$

Où θ désigne le vecteur des variables articulaires, et x est le vecteur des cordonnees opérationnelles de l'effecteur, il est donné par :

$$\boldsymbol{X = (p_x, p_y, p_z, R_x, R_y, R_z)} \qquad (II.2)$$

La méthode la plus largement utilisée pour la modélisation géométrique des robots est la méthode de Denavit-Hartenberg (DH) modifiée. Le déplacement du repère de base au repère de l'effecteur final est réalisé en se déplaçant à travers deux repères consécutifs placés aux articulations.

L'ensemble de quatre paramètres concerne la transformation entre le repère i-1 et le repère i par $d_i, \theta_i, a_{i-1}, \alpha_{i-1}$ [9].

Où :

- α_{i-1} : l'angle de rotation entre les axes Z_{i-1} et Z_i correspondant à une rotation autour de l'axeX_{i-1}.
- a_{i-1} : la distance entre Z_{i-1}et Z_i mesurée le long de l'axe X_{i-1}
- d_i : la distance entre X_{i-1}et $X_i$mesurée le long de l'axeZ_i.
- θ_i : l'angle de rotation entre les axes X_{i-1} et X_i correspondant à une rotation autour de l'axeZ_i.

Les quatre transformations élémentaires qui définissent le repère $R_i(x_i, y_i, z_i)$ dans le repère $R_i(x_{i-1}, y_{i-1}, z_{i-1})$ donnent la matrice de transformation homogène suivante :

$$\boldsymbol{{}^{i-1}_{i}T = Rot(x, \alpha_{i-1}).Trans(x, a_{i-1}).rot(z, \theta_i).Trans(z, d_i)} \qquad (II.3)$$

Après le développement de l'équation$(II.3)$, on obtient la matrice de transformation homogène ${}^{i-1}_{i}T$ comme suit :

$$\boldsymbol{{}^{i-1}_{i}T = \begin{pmatrix} \cos\theta_i & -\sin\theta_i & 0 & a_{i-1} \\ \sin\theta_i.\cos\alpha_{i-1} & \cos\theta_i.\cos\alpha_{i-1} & -\sin\alpha_{i-1} & -\sin\alpha_{i-1}.d_i \\ \sin\theta_i.\sin\alpha_{i-1} & \cos\theta_i.\sin\alpha_{i-1} & \cos\alpha_{i-1} & \cos\alpha_{i-1}.d_i \\ 0 & 0 & 0 & 1 \end{pmatrix}} \qquad (II.4)$$

Le modèle géométrique directe est obtenu en calculant le produit des matrices de transformation homogènes des différentes articulations du robot manipulateur.

$$\boldsymbol{{}^{n}_{0}T = \begin{pmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{pmatrix}} \qquad (II.5)$$

Avec : la matrice ${}^{n}_{0}T$ est la matrice de transformation de l'effecteur par rapport à la base. Sachant que la matrice :

$$\boldsymbol{Rot = \begin{pmatrix} n_x & o_x & a_x \\ n_y & o_y & a_y \\ n_z & o_z & a_z \end{pmatrix}} \qquad (II.6)$$

Désigne la matrice de rotation du repère lié à l'organe terminal du manipulateur par rapport au repère de base.

Nous souhaitons obtenir le vecteur X en fonction des éléments de la matrice de transformation homogène donnée par (II.5).

La table DH permettant de calculer les différentes matrices de transformations homogènes pour chaque articulation est donnée comme suit :

Tableau II-1: La table de DH du robot

Articulation	Paramètres d'articulation			
	a_i (mm)	α_i	d_i (mm)	θ_i
B	0	0	d_b	0
0	0	0	0	0
1	$a_1(320)$	$-\pi/2$	$d_1(780)$	$\theta'_1 = \theta_1$
2	$a_2(1135)$	0	0	$\theta'_2 - \frac{\pi}{2} = \theta_2$
3	$a_3(200)$	$-\pi/2$	0	$\theta'_3 = \theta_3$
4	0	$\pi/2$	$d_4(1182.5)$	$\theta'_4 = \theta_4$
5	0	$-\pi/2$	0	$\theta'_5 = \theta_5$
6	atcp (0)	0	0	$\theta'_6 + \pi = \theta_6$
7	0	0	$d_{TCP}(0)$	

Les matrices de transformation homogène sont données comme suit :

$$ {}^{0}_{1}T = \begin{pmatrix} cos\theta_1 & -sin\theta_1 & 0 & 0 \\ sin\theta_1 & cos\theta_1 & 0 & 0 \\ 0 & 0 & 1 & d_1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (II.7) $$

$$ {}^{1}_{2}T = \begin{pmatrix} cos\theta_2 & -sin\theta_2 & 0 & a_1 \\ 0 & 0 & 1 & 0 \\ -sin\theta_2 & -cos\theta_2 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (II.8) $$

$$ {}^{2}_{3}T = \begin{pmatrix} cos\theta_3 & -sin\theta_3 & 0 & a_2 \\ sin\theta_3 & cos\theta_3 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (II.9) $$

$$ {}^{3}_{4}T = \begin{pmatrix} cos\theta_4 & -sin\theta_4 & 0 & a_3 \\ 0 & 0 & 1 & d_4 \\ -sin\theta_4 & -cos\theta_4 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (II.10) $$

$$ {}^{4}_{5}T = \begin{pmatrix} cos\theta_5 & -sin\theta_5 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ sin\theta_5 & cos\theta_5 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (II.11) $$

$$ {}^{5}_{6}T = \begin{pmatrix} cos\theta_6 & -sin\theta_6 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -sin\theta_6 & -cos\theta_6 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (II.12) $$

$$ {}^{6}_{TCP}T = \begin{pmatrix} 1 & 0 & 0 & a_{TCP} \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_{TCP} \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (II.13) $$

Et le modèle géométrique direct s'obtient par la multiplication des différentes matrices de transformations homogènes comme suit :

$$ \boldsymbol{{}^{0}_{n}T = {}^{0}_{1}T \,.\, {}^{1}_{2}T \,.\, {}^{2}_{3}T . {}^{3}_{4}T . {}^{4}_{5}T . \; {}^{5}_{6}T} \quad (II.14) $$

Et par rapport au point de TCP, on multiplie encore par la matrice ${}^{6}_{TCP}T$.

II.4 MODÈLE GÉOMÉTRIQUE INVERSE

Le problème inverse consiste donc à calculer les coordonnées articulaires correspondant à une situation donnée de l'organe terminal [10]. Lorsqu'elle existe, la forme explicite qui donne toutes les solutions possibles (il y a rarement unicité de solution) constitue ce que l'on appelle le modèle géométrique inverse **(MGI).**

$$ \boldsymbol{\theta = \begin{bmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \\ \theta_4 \\ \theta_5 \\ \theta_6 \end{bmatrix} = f(X)^{-1}} \quad (II.15) $$

Dans ce qui suit, nous utilisons la méthode de Paul qui consiste à multiplier successivement l'équation (II.14) par l'inverse des matrices homogènes, on trouve un ensemble d'équations à résoudre pour déterminer les variables articulaires en fonction de la position finale P. En général on peut écrire :

$$ {}^{0}_{i}T^{-1}\,{}^{0}_{j}T = {}^{i}_{j}T \quad (II.16) $$

1- Axe 1 (Calcul deθ_1)

$$ {}^{0}_{1}T^{-1}\,{}^{0}_{6}T = {}^{1}_{6}T \quad (II.17) $$

On trouve :

$$ \theta_1 = atan2(y, x) \quad (II.18) $$

PS : Où atan2 (y, x) est une autre version de atan(x/y) que renvoie la tangente inverse à quatre quadrants

2- Axe 2 (Calcul deθ_2)

On utilise la relation suivent :

$$ \boldsymbol{{}^{0}_{2}T^{-1} \times {}^{0}_{6}T = {}^{2}_{6}T} \quad (\boldsymbol{II.19}) $$

$$ \boldsymbol{\theta_2 = atan2(\, -c \times e_1 + a \times e_2 \quad , \quad d \times e_1 - b \times e_2)} \quad (II.20) $$

Avec :

$a = -a_1 + ysin(\theta_1) + xcos(\theta_1) \quad ; \; b = d_1 - z$

$d = a_1 - ysin(\theta_1) - xcos(\theta_1) \quad ; \; c = d_1 - z$

$e_1 = a_2 + a_3cos(\theta_3) - d_4\, sin(\theta_3) \; ; \; e_2 = d_4cos(\theta_3) + a_3sin(\theta_3)$

Notons que θ_3 calculée avant de calculer θ_2 .

3- Axe 3 (Calcul deθ_3)

En utilise la relation suivant pour trouver θ_3 :

$$ {}^0_2T^{-1}\,{}^0_6T = {}^2_6T $$

On trouve :

$$ \theta_3 = \text{atan2}\left(\frac{(a_3\cos(\theta_3) - K)}{d_4}, \frac{a_3K \pm \sqrt{(a_3K)^2 - ({a_3}^2 + {d_4}^2)(K^2 - {d_4}^2)}}{{a_3}^2 + {d_4}^2}\right) \qquad (II.21) $$

Avec :

$$ K = \frac{1}{2a_2}\left((-a_1 + xcos(\theta_1) + ysin(\theta_1))^2 - {a_2}^2 - {a_3}^2 - {d_4}^2 + (z - d_1)^2\right) $$

4- Axe 4 (Calcul de θ_4)

Le problème de l'orientation inverse est un peu plus léger en termes d'équations ou la position inverse. Pour cela on s'intéresse à la matrice de rotation **R** de l'équation :

$$ {}^0_3T^{-1}\,{}^0_6T = {}^3_6T \qquad (II.22) $$

On trouve :

$$ \theta_4 = atan2(b, a) \qquad (II.23) $$

Tel que :

$a = r_{23} \times cos(\theta_1) - \; r_{13} \times sin(\theta_1)$

$b = r_{33} \times sin(\theta_2 \; + \theta_3) - \; r_{13} \times cos(\theta_2 \; + \theta_3) \times cos(\theta_1) - \; r_{23} \times cos(\theta_2 + \theta_3) \times sin(\theta_1)$

5- Axe 5 : (Calcul de θ_5)

Même qu'axe 4, On utilise la relation $(II.22)$:

$$ \theta_5 = atan2(\pm\sqrt{a^2 + b^2}\,, C) \qquad (II.24) $$

Tel que :

$a = r_{23} \times cos(\theta_1) - \; r_{13} \times sin(\theta_1)$

$b = r_{33} \times sin(\theta_2 \; + \theta_3) - \; r_{13} \times cos(\theta_2 \; + \theta_3) \times cos(\theta_1) - \; r_{23} \times cos(\theta_2 + \theta_3) \times sin(\theta_1)$

$C = r_{33} \times cos(\theta_2 \; + \theta_3) - \; r_{13} \times sin(\theta_2 \; + \theta_3) \times cos(\theta_1) - r_{23} \times sin(\theta_2 + \theta_3) \times sin(\theta_1)$

6- Axe 6 : (Calcul de θ_6)

Finalement

$$ {}^{0}_{5}T^{-1}\,{}^{0}_{6}T = {}^{5}_{6}T \qquad (II.25) $$

Donc

$$ \theta_6 = atan2(a, b) \qquad (II.26) $$

Tel que

$$ a = -r_{12} \times \left(c_5 s_1 s_4 - s_{2,3} c_1 s_5 + c_{2,3} c_5 c_1 c_4\right) + r_{22} \times \left(s_{2,3} s_1 s_5 + c_5 c_1 c_4 - c_{2,3} c_5 s_1 c_4\right) + r_{32} \times \left(c_{2,3} s_5 + s_{2,3} c_4 c_5\right) $$

$$ b = r_{12} \times \left(c_4 s_1 - c_{2,3} c_1 s_4\right) - r_{22} \times \left(c_1 c_4 - c_{2,3} s_1 s_4\right) + r_{32} s_{2,3} s_4 $$

Ou c_i = cos (θ_i), de même s_i = sin (θ_i).

Dans le cas d'un suffixe de structure i, j, cela correspond à $\theta_i + \theta_j$.

II.4.1 Validation de model géométrique inverse

On utilise MATLAB pour calculer l'erreur de position, le tableau (II.2) montre les résultats d'essai :

Méthode de travail :

1- On donne différentes valeurs au vecteur des variables articulaire et on calcule la matrice $\boldsymbol{{}^{0}_{6}T}$ en utilisant le MGD.

2- A partir de $\boldsymbol{{}^{0}_{6}T}$ calculé précédemment on calcule le vecteur des variables articulaires en utilisant le MGI.

3- On recalcule une autre fois la matrice $\boldsymbol{{}^{0}_{6}T}$ en utilisant les variables articulaire calculées à l'étape précédente.

4- Enfin, on calcule l'erreur de position entre les deux matrices obtenues respectivement aux étapes 1 et 3.

Tableau II-2: Table d'essai

essai	1
θ(degré)	$[45 \;\; 72 \;\; 25.78 \;\; 86.20 \;\; 198 \; 158.22]$
[x,y,z](mm)	$[-373.32\,, -373.32, -337.4]$
θinv (degré)	$[-135 \;\; 15.72 \;\; 41.4 - 43.5 \;\; 26.04 \;\; 114.66]$
[x,y,z](mm)	$[-373.32\,, -373.32, -337.4]$
Erreur (mm)	[0,0,0]

Sachant que la solution du MGI peut ne pas être unique. Ceci est justifié qu'une position dans l'espace opérationnel peut être atteinte plusieurs combinaisons des variables articulaires.

CONCLUSION

Ce chapitre est consacré à la modélisation géométrique et cinématique du ABB-IRB 6700. Après avoir introduit la convention de Denavit-Hartenberg pour la modélisation des robots manipulateurs à chaine ouverte, un modèle géométrique direct est calculé pour le robot. L'obtention de son modèle géométrique inverse est d'une grande utilité pour le suivi d'une trajectoire dans l'espace opérationnel ainsi que pour le calcul de la commande. Pour cela, nous avons appliqué la méthode de Paul qui convient à presque tous les robots à chaine ouverte.

III. Chapitre III : *Modélisation, contrôle et supervision de la station robotisée*

III.1 INTRODUCTION

Ces dernières années la simulation des cellules robotisées devient possible grâce à l'apparition de certaines applications comme Grasshopper, HAL, ABB machining PowerPac. Cette dernière est à l'origine de l'apparition d'un outil très puissant pour la simulation des robots ABB appelé Robot-Studio. Cette application offre la possibilité de modéliser l'environnement du robot et sa programmation. L'avantage d'utiliser le propre logiciel d'ABB est que l'utilisateur est assuré que si le chemin est exécutable dans la simulation, il le sera également sur le robot. Une fonctionnalité que les autres logiciels n'ont pas. Pour cela, la première partie de ce chapitre est consacrée à la création de l'environnement de la cellule robotisé en utilisant le Robot-Studio.

La deuxième partie de ce chapitre est consacré au côté automatisme de la cellule. En fait, la cellule est contrôlée et supervisée par un automate de type Siemens 1200. La configuration de la communication de type Profinet entre le Robot (Langage Rapide) et l'automate via Tia-Portal sera présentée.

III.2 PRÉSENTATION DE ROBOT-STUDIO

Robot-Studio est une application PC permettant la modélisation, la programmation hors ligne et la simulation des cellules robotiques. Le Robot-Studio permet également de travailler avec un système de commande hors ligne, c'est-à-dire un système de commande IRC5 virtuel exécutable localement sur PC. Ce système de commande hors ligne est également appelé système de commande virtuel (VC). Robot-Studio permet également de travailler avec un système de commande IRC5 physique réel, simplement appelé système de commande réel. Lorsque Robot-Studio est utilisé avec des systèmes de commande réels, on parle de mode en ligne. Lorsque on travaille sans être connecté à un système de commande réel, ou en étant connecté à un système de commande virtuel, Robot-Studio est dit en mode hors ligne [10].

III.2.1 Termes et concepts

a- Concepts matériels : Cette section est consacrée aux matériels généralement utilisés dans une cellule robotisée IRC5.

a.1 Matériel standard :

- **Robot Manipulateur :** Un robot industriel ABB.
- **Module de commande :** Contient l'ordinateur principal qui commande les mouvements du manipulateur. Ce module gère également l'exécution RAPID et le traitement des signaux. Un module de commande peut être connecté à 1 à 4 modules d'entraînement.
- **Drive module :** Un module contient l'électronique qui anime les moteurs d'un manipulateur. Il peut contenir jusqu'à neuf variateurs, chacun contrôle une articulation du manipulateur. Les manipulateurs standards disposant de six articulations.
- **Flex-Controller :** L'armoire de commandes des robots IRC5. Elle comprend un module de commande et un module d'entraînement pour chaque manipulateur du système.
- **Flex-Pendant :** Le pupitre de programmation, connecté au module de commande. La programmation sur le Flex-Pendant est appelée « programmation en ligne ».

- **Outil :** Un dispositif généralement installé sur le robot manipulateur pour lui permettre de réaliser certaines tâches (par exemple : serrage, découpe, soudage), L'outil peut également être fixe.

a.2 Matériel facultatif

- **Manipulateur de translation :** Une pince mobile qui maintient le manipulateur de robot pour augmenter son espace de travail. Lorsque le module de commande contrôle les mouvements d'un manipulateur de translation, on l'appelle « axe externe de translation ».
- **Manipulateur positionneur :** Une pince mobile qui tient habituellement une pièce de travail ou une installation fixe. Lorsque le module de commande contrôle les mouvements d'un manipulateur positionneur, il est également appelé « axe externe ».
- **Flex-Positioner :** Une pince mobile qui tient habituellement une pièce de travail ou une installation fixe. Lorsque le module de commande contrôle les mouvements d'un manipulateur positionneur, il est également appelé « axe externe ».
- **Outil fixe :** Un appareil qui se trouve sur une place fixe. Le robot manipulateur prélève le porteur de pièce et le conduit vers l'outil pour y réaliser une tâche spécifique (par exemple : encollage, serrage, soudage).
- **Pièce de travail :** Le produit sur lequel on travaille.
- **Installation fixe :** Une installation fixe maintient le porteur de pièce dans une position spécifique et permet de conserver la répétabilité de la production.

b- Concepts de RAPID

- **Déclaration de données :** Elles permettent de créer des instances de variables ou des types de données, comme Num ou Tool-data.
- **Instruction :** Les commandes de code réelles qui permettent de déclencher des événements (par exemple, la définition d'une donnée sur une valeur spécifique ou le mouvement d'un robot). On peut uniquement créer des instructions à l'intérieur d'une routine
- **Instructions de mouvement :** Créer les mouvements du robot. Ils consistent en une référence à une position indiquée dans une déclaration de données, et des paramètres qui définissent les spécifications du mouvement et des processus. Si des positions en ligne sont utilisées, la position est déclarée dans l'instruction de mouvement.
- **Instruction d'action :** Instructions exécutant des actions différentes de celles permettant de déplacer le robot, comme définir des données ou des propriétés de synchronisation.
- **Routine :** Généralement, un ensemble de déclarations de données, suivi d'un ensemble d'instructions permettant la mise en place d'une tâche. Les routines peuvent être divisées en trois catégories : les procédures, les fonctions et les routines d'interruption.
- **Procédure :** Ensemble d'instructions qui ne renvoient pas de valeur.
- **Fonction :** Ensemble d'instructions qui renvoient une valeur.
- **Routine d'interruption :** Ensemble d'instructions déclenchées par une interruption.
- **Module :** Un ensemble de déclarations de données suivi d'un ensemble de routines. On peut enregistrer, charger et copier les modules comme des fichiers. Les modules sont divisés en modules de programme et modules de système.
- **Module de programme (.mod) :** Ils peuvent être chargés et déchargés au cours de l'exécution.

- **Module de système (.sys) :** Ils sont principalement utilisés pour les routines et les données propres au système (un module de système « arcware » utilisé pour tous les robots d'arc).
- **Fichiers de programme (.pgf) :** Dans IRC5, un programme RAPID est un ensemble de fichiers de module (.mod) et le fichier de programme (.pgf.) qui fait référence à tous les fichiers de module. Au chargement d'un fichier de programme, tous les anciens modules de programme sont remplacés par ceux référencés dans le fichier .pgf. Les modules du système ne sont pas affectés par le chargement du programme.

c- Concepts de programmation

c.1 Programmation en ligne : Programmation lorsqu'on est connecté à un système de commande réel. Cette expression implique également l'utilisation du robot pour créer des positions et des mouvements.

c.2 Programmation hors ligne : Programmation sans être connecté au robot ou au système de commande réel.

c.3 Programmation réelle hors ligne : La programmation réelle hors ligne se rapporte au concept d'ABB Robotics consistant à connecter un environnement de simulation à un système de commande virtuel. Cette opération permet de créer des programmes, mais aussi de tester et d'optimiser les programmes hors ligne.

c.4 Système de commande virtuel : C'est un logiciel qui émule un Flex-Controller pour permettre au logiciel (le système Robot-Ware) qui contrôle les robots d'être exécutés sur un PC. On peut ainsi observer le comportement des robots hors ligne comme s'ils étaient en ligne.

c.5 Multi-Move : Exécution de plusieurs manipulateurs de robot à l'aide du même module de commande.

c.6 Systèmes de coordonnées : Permet de définir des positions et des orientations. Lors de la programmation d'un robot, on peut utiliser différents systèmes.

c.7 Frame : Synonyme de système de coordonnées.

c.8 Étalonnage de repères objet : Si toutes vos positions se rapportent à des repères objets, On peut uniquement étalonner ces repères objets lorsque on déploie des programmes hors ligne.

d- Positions et trajectoires

Les positions (cibles) et les trajectoires (suites d'instructions de déplacement vers les positions) sont utilisées lors de la programmation des déplacements des robots dans Robot-Studio. Lorsqu'on synchronise la station RobotStudio avec le système de commande virtuel, les programmes RAPID sont créés à partir des trajectoires.

d.1 Positions : Une position est une coordonnée que le robot doit atteindre. Elle contient les informations suivantes :

- **Position :** Emplacement de la position défini dans un système de coordonnées de repère objet
- **Orientation :** Orientation de la position par rapport à l'orientation du repère objet. Lorsque le robot atteint la position, il aligne l'orientation du CDO sur celle de la position

- **Configuration :** Valeurs de configuration qui indiquent comment le robot doit atteindre la position.

Les positions sont converties en instances du type de données rob-target lorsqu'elles sont synchronisées avec le système de commande virtuel.

d.2 Trajectoires : Les trajectoires sont des séquences d'instructions de mouvement, utilisées pour déplacer le robot en fonction d'une suite de positions.

Les trajectoires sont converties en procédures lorsqu'elles sont synchronisées avec le système de commande virtuel.

d.3 Instructions de déplacement : Une instruction de mouvement se compose des éléments suivants :

- une référence à une position.
- des données relatives aux déplacements (par exemple, le type de mouvement, la vitesse et la zone).
- une référence à une donnée d'outil.
- une référence à un repère objet.
- Instructions d'action : Une instruction d'action est une chaîne RAPID qui sert à définir et à modifier des paramètres. Les instructions d'action peuvent être insérées avant, après ou entre les instructions de position des trajectoires.
- Systèmes de coordonnées : Les référentiels sont en relation hiérarchique. Le point d'origine de chaque référentiel est défini comme une position de l'un de ses prédécesseurs.

e- Interface utilisateur

L'illustration suivante affiche le ruban, les onglets et les groupes de l'interface utilisateur graphique et le tableau (III-1) montre leurs descriptions.

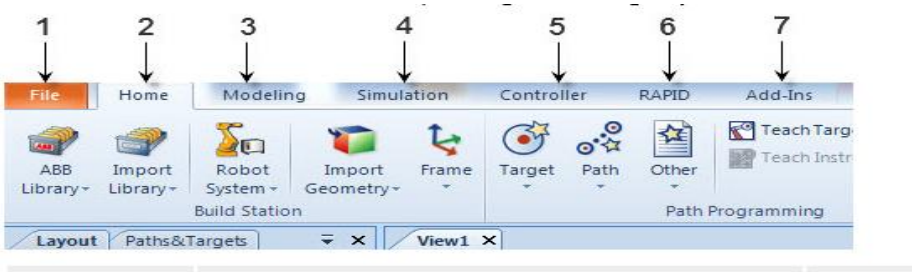


Figure III-1: La barre d'outils du Studio-Robot

Tableau III-1: Description des onglets de la barre d'outils.

	Tabulation	Description
1	Fichier	Comporte les options de création d'une nouvelle station ou d'un nouveau système robotique, de connexion à un système de commande, d'enregistrer une station en tant que visualiseur et les options RobotStudio.

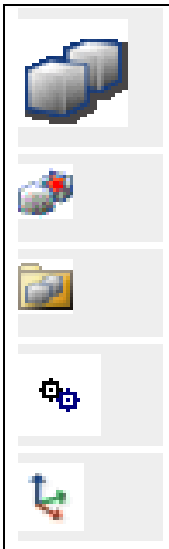
2	Début	Il contient les commandes nécessaires à la création des stations, à la création de systèmes, à la programmation de trajectoires et au placement d'éléments.
3	Modèles	Il contient les commandes de création et de groupement des composants, la création des corps, les mesures et les opérations de CAO.
4	Simulation	Il contient les commandes d'installation, configuration, commande, surveillance et enregistrement des simulations.
5	Système de commande	Elle contient les commandes de synchronisation, de configuration et de tâches affectées au système de commande virtuel (VC). Elle contient également les commandes de gestion du système de commande réel.
6	RAPID	Contient l'éditeur RAPID intégré, utilisé pour modifier toutes les tâches du robot autres que le mouvement du robot.
7	Ajouts	Contient la commande de PowerPacs.

e.1 Navigateur d'Agencement

Le navigateur d'agencement est une vue hiérarchique des éléments physiques tels que les robots et les outils, le tableau (III-2) montre la description des éléments du navigateur d'agencement.

Tableau III-2: Description des éléments du navigateur d'agencement

Nœud		Description
	Robot	Les robots dans la station.
	Outil	Un outil.
	Ensemble de liens	Contient tous les liens des objets.
	Liaison	Un objet physique dans une jonction d'axe. Chaque liaison est constituée d'une ou de plusieurs pièces.
	Repères	Contient tous les repères d'un objet.
	Groupe de composants	Un groupe de pièces ou d'ensembles doté de son propre système de coordonnées. Il permet de structurer une station.
	Pièce	Un objet physique dans RobotStudio. Les pièces dotées d'informations géométriques sont constituées d'une ou plusieurs entités à 2 ou 3D. Les pièces dépourvues d'informations géométriques (comme les fichiers .jt importés) sont vides.
	Jeu de collisions	Contient tous les jeux de collisions. Chaque jeu de collisions contient deux groupes d'objets.
	Groupe d'objets	Contient des références aux objets concernés par la détection des collisions.
	Mécanismes de jeu de collisions	Les objets du jeu de collisions.

	Repère	Les repères dans la station.

e.2 Navigateur de trajectoires et cible

Le navigateur de trajectoires et positions est une vue hiérarchique des éléments non physiques. Le tableau (III-3) montre la description des éléments du navigateur trajectoire et cible.

Tableau III-3: Description des éléments de navigateur trajectoire et cible.

Icône	Nœud	Description
	Station	la station dans RobotStudio
	Système de commande virtuel	Le système qui permet de contrôler les robots, comme un véritable système de commande IRC5.
	Tâche	Il contient tous les éléments logiques de la station, tel que les positions, les trajectoires, les repères objet, les repères outils et les instructions.
	Ensemble de repères outil	Contient tous les repères outil.
	Repère outil	Les données outil d'un robot ou d'une tâche.
	Repères objet et positions	Contient tous les objets de travail et les positions de la tâche ou du robot.
	Ensemble de positions Joint target et position Joint target	Position spécifique des axes du robot.
	Ensemble de repères objet et repère objet	Le noeud Ensemble de repères objet et les repères objet qu'il contient.
	Position	Une position et une rotation définies pour un robot. Une position équivaut à une position RobTarget dans un programme RAPID.
	Ensemble de trajectoires	Contient toutes les trajectoires de la station.

	Trajectoire	Contient les instructions pour les mouvements du robot.
	Instruction de mouvement linéaire	Mouvement CDO linéaire vers une position. Si aucune configuration valide n'est attribuée à la position, l'instruction de déplacement est associée aux mêmes symboles d'avertissement que la position.
	Instruction de mouvement articulaire	Mouvement articulaire vers une position. Si aucune configuration valide n'est attribuée à la position, l'instruction de déplacement est associée aux mêmes symboles d'avertissement que la position.
	Instruction RAPID	Définit une action à effectuer pour le robot, à un endroit spécifique sur sa trajectoire.

e.3 Le navigateur Système de commande

Le navigateur Système de commande affiche de façon hiérarchique le système de commande et les éléments de configuration qui se trouvent dans la vue de l'onglet Système de commande. « Voir Tableau III-4 »

Tableau III-4: Description des éléments du navigateur Système de commande.

Icône	Nœud	Description
	Systèmes de commande	Contient les systèmes de commande connectés à la vue de robot.
	Système de commande connecté	Représente un système de commande avec une connexion en cours de fonctionnement.
	Connexion du système de commande	Représente un système de commande actuellement connecté.
	Système de commande déconnecté	Représente un système de commande ayant perdu sa connexion. Il se peut qu'il ait été fermé ou déconnecté du réseau.
	Connexion refusée	Représente un système de commande auquel l'accès de connexion vous est refusé. Les raisons du refus possibles sont les suivantes : • L'utilisateur ne possède pas les privilèges d'accès requis. • Nombre trop important de clients déjà connectés au système de commande. • La version RobotWare du système exécuté sur le système de commande est plus récente que la version RobotStudio.
	Configuration	Contient les rubriques de configuration.
	Sujet	Chaque rubrique de paramètre est représentée par un nœud : a) Communications b) Système de commande c) I/O

	d) Communication homme-machine e) Mouvement
Journal des évènements	Le journal des événements vous permet de visualiser et d'enregistrer les événements du système de commande.
Système d'E/S	Représente le système d'E/S du système de commande. Le système d'E/S se compose de réseaux industriels et d'unités.
Réseau industriel	Un réseau industriel est un connecteur destiné à un ou plusieurs périphériques.
Périphérique	Un périphérique est une carte, un panneau ou tout autre périphérique comportant des ports grâce auxquels sont envoyés les signaux d'E/S.
Tâches RAPID	Contient les tâches actives (programmes) du système de commande.
Tâche	Une tâche est un programme de robot s'exécutant seul ou associé à d'autres programmes. Un programme est composé d'un ensemble de modules.
Modules de programme	Les modules de programme contiennent un ensemble de déclarations de données et de routines pour une tâche spécifique. Les modules de programme contiennent des données spécifiques à cette tâche.
Modules système	Les modules système contiennent un ensemble de définitions de type, déclarations de données et routines. Les modèles système contiennent des données qui s'appliquent au système de robot, indépendamment des modules de programme chargés.
Module sans entrée	Un module dans lequel on ne peut entrer lors d'une exécution pas à pas. Cela signifie que toutes les instructions du module sont traitées sous forme d'une instruction unique si le programme est exécuté pas à pas.
Modules de programme en affichage seul et en lecture seule	Une icône pour les modules de programme qui sont en affichage seul ou en lecture seule.
Modules système en affichage seul et en lecture seule	Une icône pour les modules système qui sont en affichage seul ou en lecture seule.
Procédure	Une routine qui ne renvoie aucune valeur. Les procédures servent de sous-programmes.
Fonction	Une routine qui renvoie une valeur d'un type spécifique.
Routine d'interruption	Une routine qui donne un moyen de réagir aux interruptions.

III.3 LA CELLULE DU ROBOT ABB-IRB 6700 (Modèle sous RobotStudio)

Pour créer la cellule robotisée on doit suivre les étapes suivantes :

a- Nous créons d'abord une station vide (voir Figure III-2), puis nous ajoutons un robot 6700 de la bibliothèque (ABB Library).

b- A partir de l'interface Modélisation et dans la boîte « créer » nous cliquons sur solide, nous choisissons le cube pour la création de la structure cellulaire. « Voir Figure III-3 »

Robot-Studio utilise des systèmes de commande virtuels pour piloter les robots. Ces systèmes 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.

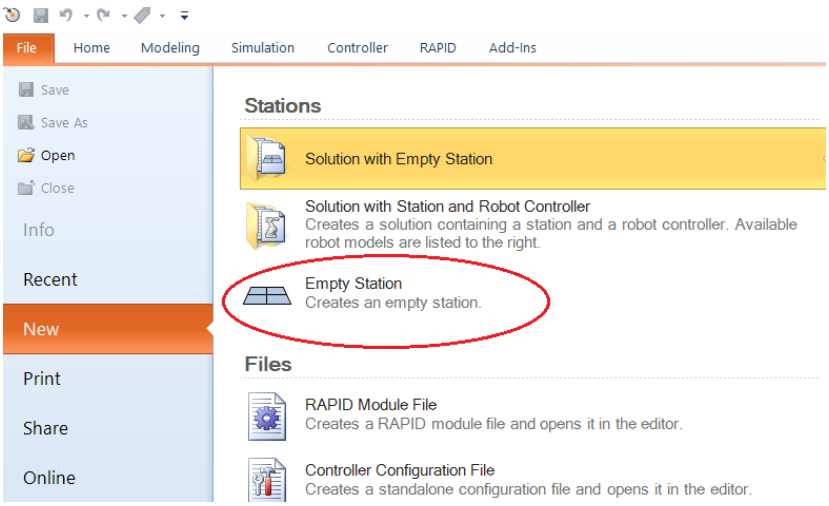


Figure III-2: Création d'une station

Remarque : Lors du démarrage d'un système de commande virtuel, on indique quel système doit y être exécuté. Comme le système contient des informations sur les robots à utiliser ainsi que des données importantes, telles que les configurations et les programmes de robot, nous devons sélectionner un système approprié à la station.

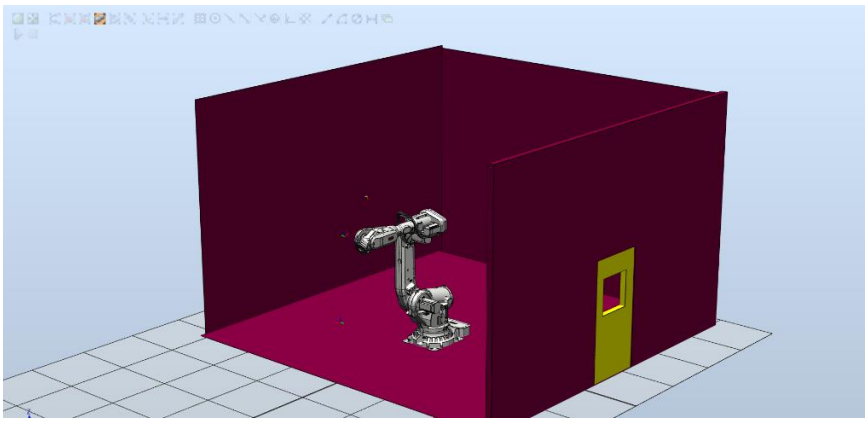


Figure III-3: Création de la structure cellulaire.

c- ensuite nous relions le robot avec un système de commande IRC5 virtuel que est un système de commande robotique de cinquième génération qui combine contrôle des mouvements, flexibilité, modularité, convivialité, interfaces d'application et sécurité [7].

d- Nous créons des modèles géométriques sous forme d'outils tels que (table-tournante, meuleuse …)

e- Nous créons les caractéristiques d'un outil avec la boîte (créer outil) tels que :

e.1 En station

Chaque outil comporte les paramètres suivants :

- Masse (poids) de l'outil en kg.
- Centre de gravité de la charge de l'outil (x, y et z) en mm, exprimée dans le repère poignet.
- L'orientation des axes d'inertie principale du moment de l'outil exprimé dans le repère poignet.
- Les moments d'inerties autour des axes d'inerties du moment en kgm^2.

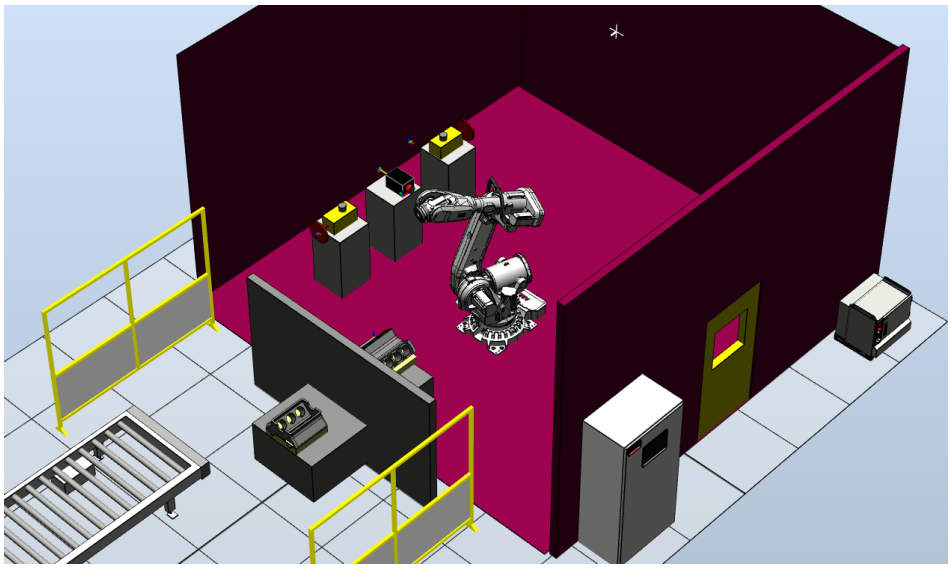


Figure III-4: La cellule robotisée complète créée sous Robot Studio

e.2 Par l'utilisation de LANGAGE RAPID

```
MODULE CalibData

TASKPERS tooldataTooldata_trosen

:=[TRUE,[[422.93,2.96,153.43],[0.70,0, 0.70,0]],[10,[0,0,1],[1,0,0,0],0,0,0]];

ENDMODUL
```

Explication de code : L'outil est décrit à l'aide des valeurs suivantes :

- Le robot maintient l'outil(TRUE).
- Le TCP est situé sur un point à 153.43 mm le long de l'axe Z et à 2.96 mm le long de l'axe Y et 422.93 sur X du repère poignet. ([422.93, 2.96, 153.43])
- Les directions X, Yet Z de l'outil. ([0.70, 0, 0.70, 0])
- La masse de l'outil est de 10 kg.

- Le centre de gravité est situé sur un point à 1 mm tout droit en dehors de la bride de montage.

PS : Les données outil affectent les mouvements du robot de l'une des manières suivantes :

- Le point central de l'outil (TCP) fait référence à un point satisfaisant les performances de chemin et de vitesse spécifiées. Si l'outil est réorienté ou si les axes externes de coordination sont utilisés, seul ce point suit le chemin souhaité à la vitesse programmée.

- Si un outil fixe est utilisé, la vitesse et la trajectoire programmées font référence à l'objet de travail, maintenu par le robot.

- Les positions programmées font référence à la position du TCP actuel et à l'orientation relative au repère outil. Cela signifie que si, par exemple, un outil est remplacé car il est endommagé, l'ancien programme peut toujours être utilisé si le repère outil est redéfini.

f- Nous créons les Works Object qui permettent de décrire l'objet de travail de robot, dans lequel il se déplace. Avec ses composants :

- robhold (robot hold) « True /false » ; Indique si le robot maintient l'objet de travail dans la tâche de programme réelle.
- ufprog (user frame programmed) « True /false » ; Définit si un repère utilisateur fixe est utilisé.
- ufmec (user frame mechanical unit) ; Indiquez le nom de l'unité mécanique définie dans les paramètres système.
- Uframe (user frame) ; Repère utilisateur, c'est-à-dire la position de la surface de travail actuelle
- oframe (object frame) ; Repère objet, c'est-à-dire la position de l'objet de travail actuel.
- La configuration du système de commande (signal d'entrée et sortie (DI, DO), IP adresse…)

g- Création d'un mécanisme (table-tournant, attache outil….), avec la (Create Mecahnism) ensuite on choisit le type de mécanisme (outil, périphérique….), chaque type a une configuration spécifique (parent, Childe, rotation, signal d'exécution de mécanisme temps de transmission …).

III.4 CONTRÔLE ET SUPERVISION DE LA STATION ROBOTISÉE

La plateforme Totally-Integrated Automation Portal (TIA Portal) est le nouvel environnement de travail Siemens qui permet de mettre en œuvre des solutions d'automatisation avec un système d'ingénierie intégré comprenant les logiciels SIMATIC STEP 7 et SIMATIC Win CC.

III.4.1 Cahier de charge :

Le fonctionnement défini doit satisfaire les conditions du cahier de charge suivantes :

- Les variateurs peuvent fonctionnés en mode automatique ou en mode manuelle (par API, par le Robot, opérateur via l'IHM, en plus de la détection des erreurs et reset le VFD …).
- La table doit pouvoir tourner en mode manuelle par l'opérateur ou bien en mode automatique.
- L'échange de données entre le robot et l'API doit se faire via la communication Profinet.

- Mise en œuvre de la supervision du système via une IHM.
- Les alarmes (défaut) doivent être Générées et affichées sur l'IHM.

III.4.1 Architecture du projet

Notre projet est constitué de trois éléments « figure III.5 », en l'occurrence le robot qui vas exécuter la tâche à accomplir, l'API qui fait l'acquisition et le traitement des données en prévenance du robot et l'IHM qui permettra à l'utilisateur de modifier des consignes où des tâches. Nous allons mettre en œuvre une communication Profinet entre ces éléments.

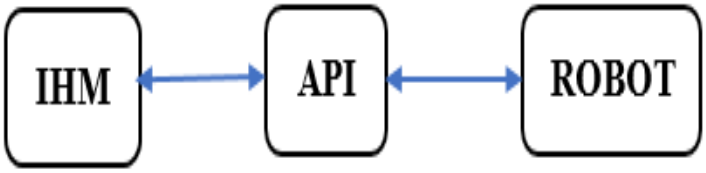


Figure III-5: les constituons de notre projet

III.4.2 Configuration matériels

Nous avons commencé par faire le bilan des entrées et sorties nécessaires afin de définir notre besoin en matière de module et pouvoir mettre en œuvre la configuration matérielle de l'automate. Le tableau (III.5) représente le bilan des entrées sorties physique dont nous avons besoin.

Tableau III-5: bilan des entrées sorties physique de l'API.

	Nombre	Type	Signal
Entrées	27	TOR	24V DC
	0	Analogique	(4-20 mA) (-10/10V)
Sorties	14	TOR	24V DC/ 0.5A
	3	Analogique	(4-20 mA) (-10/10V)

En se basant sur le tableau précédent et la documentation technique de siemens [6], nous avons choisi la configuration présentée sur la figure (III-6). Les détails de cette configuration sont donnés sur le tableau (III.6).

Tableau III-6: Bilan des entrées sorties physique de l'API.

Modules	**Références**
CPU	6ES7 214-1HG40-0XB0
DI 16 bit /DO 16 bit	6ES7 223-1PL32-0XB0

AQ 4x14bits	6ES7 232-4HD32-0XB0
DI 8 bit	6ES7 221-1BF32-0XB0

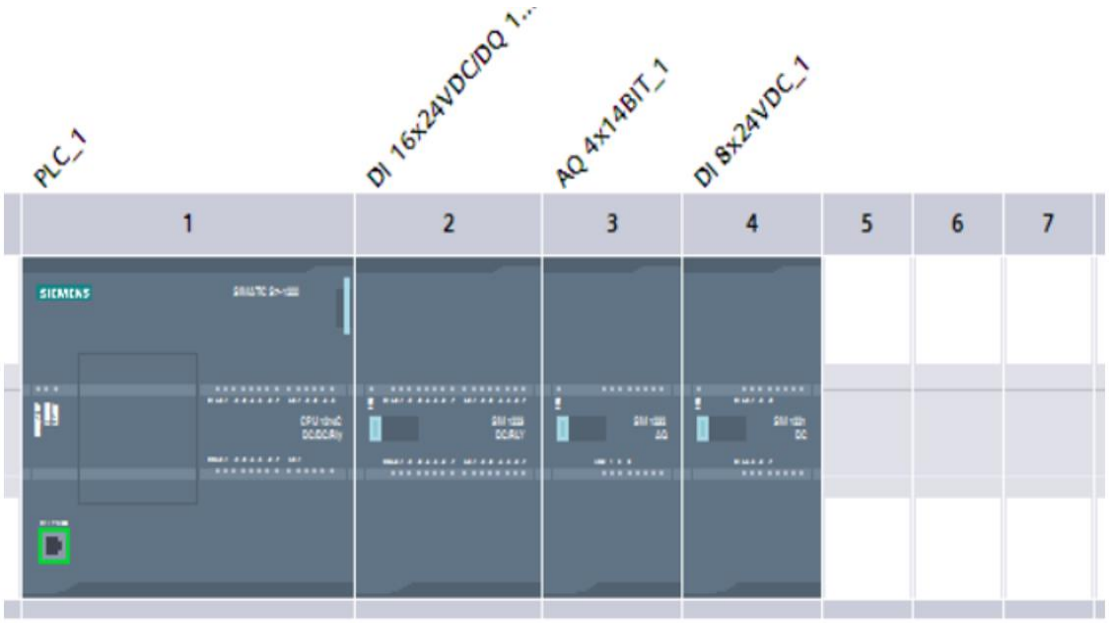


Figure III-6: Illustration de la configuration matérielle choisie

L'adressage : le tableau (III.7) montre l'adressage de LPI.

Tableau III-7: l'adressage de LPI

IP adresse CPU	**198.168.43.150**
DI CPU	(0.0)…, (1.5)
DO CPU	(0.0)…. (1.5)
DI 16bit	(2.0)… (3.7)
DO 16bit	(2.0)… (3.7)
AI 4_14bit	(100)… (104)
DI 8bit	(4.0)… (4.8)

Dans notre projet nous ajoutons IHM de type KTP700 basic PN et un processeur de communication (x_getway) pour le robot en installant son fichier GSD (pilote) qui nous permettra l'échange de données entre l'API et le robot. Ce pilote est un esclave (appareil profinet) dans un réseau profinet, il dispose d'une zone d'échange (adresses non physique) de 16 mots en entrée et 16 mots en sorties qui serons respectivement des zones de la mémoire image des entrées et la mémoire image des sorties du maitre (contrôleur profinet).

Nous avons configuré notre API en tant que maitre (contrôleur) sur le réseau profinet et le pilote du robot en tant qu'esclave. Nous avons défini notre zone d'échange du maitre (contrôleur) à partir de l'octet 300 pour les entrées mais aussi à partir de 300 pour les sorties. La figure ci-dessous représente la configuration du réseau mise en œuvre.

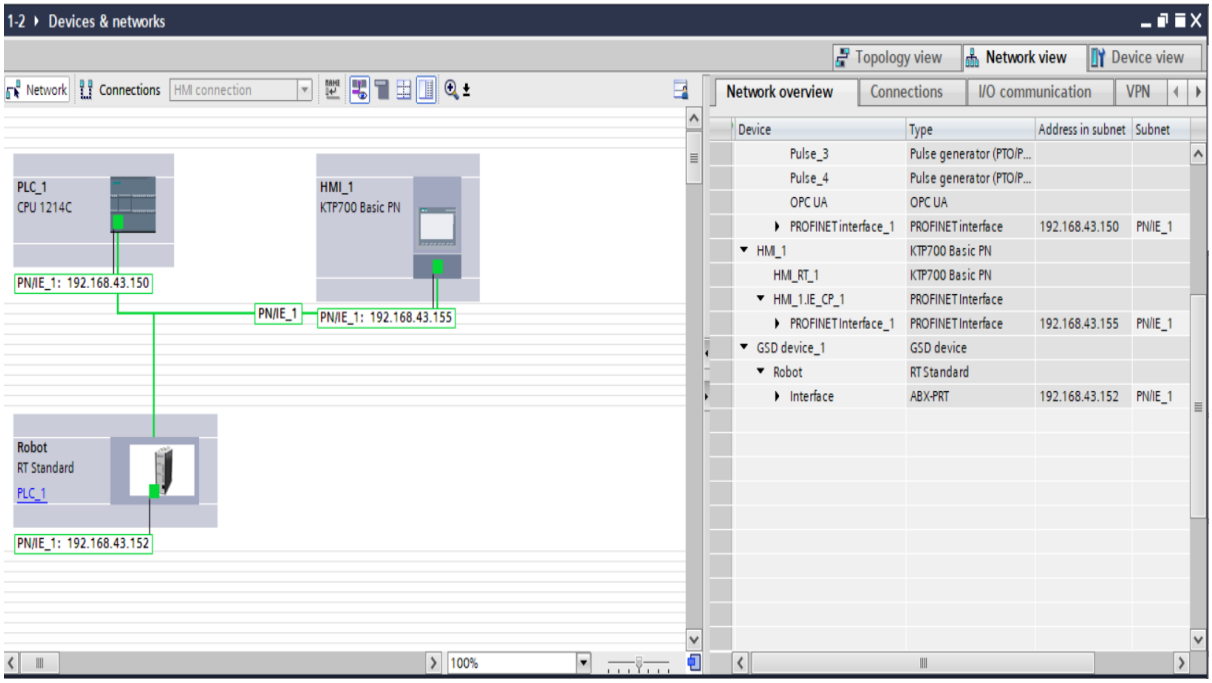


Figure III-7 : Illustration de la configuration réseau

Le fichier GSD (Pilote) du robot est représenté sur la figure (III.8) avec les zones d'échange configurées du maitre.

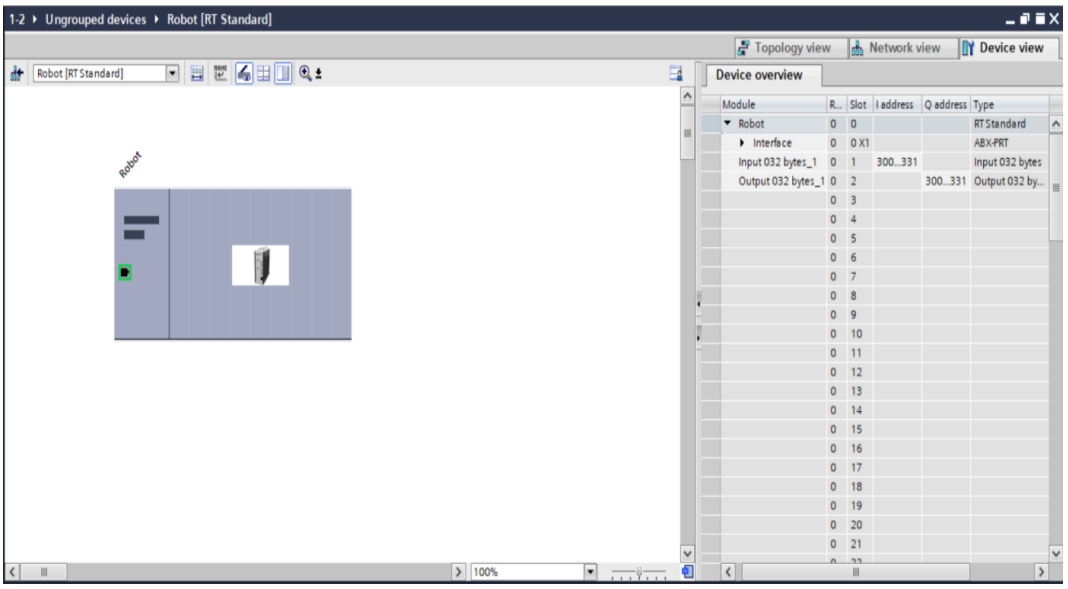


Figure III-8: Illustration de l'interface du robot au niveau du TIA

III.4.3 Variable API

Afin de faciliter la compréhension du programme lors de la mise en œuvre ou lors de la maintenance, nous avons défini des mnémoniques (symboles) pour les adresses que nous allons utiliser. A cet effet, nous avons trois tables de variables, la première regroupe toutes les entrées (figure III.9), la seconde regroupe toutes les sorties (figure III.10).

input

	Name	Data type	Address	Retain	Acces...	Writa...	Visibl...
1	Turn_Table	Bool	%I0.0		☑	☑	☑
2	invert-1 fault	Bool	%I0.1		☑	☑	☑
3	invert-2 fault	Bool	%I0.2		☑	☑	☑
4	invert-3 fault	Bool	%I0.3		☑	☑	☑
5	EMS Door1	Bool	%I0.4		☑	☑	☑
6	app_open door1	Bool	%I0.5		☑	☑	☑
7	app_restart cycle(door1)	Bool	%I0.6		☑	☑	☑
8	door lock 1(fault)	Bool	%I0.7		☑	☑	☑
9	push valid	Bool	%I1.1		☑	☑	☑
10	zero speed 1	Bool	%I1.2		☑	☑	☑
11	zero speed 2	Bool	%I1.3		☑	☑	☑
12	zero speed 3	Bool	%I1.4		☑	☑	☑
13	EMS cabin	Bool	%I1.5		☑	☑	☑
14	speed 1_ok	Bool	%I2.0		☑	☑	☑
15	speed 2_ok	Bool	%I2.1		☑	☑	☑
16	speed 3_ok	Bool	%I2.2		☑	☑	☑
17	Door1closed	Bool	%I2.3		☑	☑	☑
18	light_curtains ok	Bool	%I2.4		☑	☑	☑
19	table side_A	Bool	%I2.5		☑	☑	☑
20	table side_B	Bool	%I2.6		☑	☑	☑
21	door1 closed_loked	Bool	%I2.7		☑	☑	☑
22	Air_ok	Bool	%I3.0		☑	☑	☑
23	Ems door2	Bool	%I4.0		☑	☑	☑
24	door2_Error	Bool	%I4.3		☑	☑	☑
25	door2 closed	Bool	%I4.4		☑	☑	☑
26	door2 closed_loked	Bool	%I4.5		☑	☑	☑
27	ER1	Word	%IW302		☑	☑	☑
28	R_C_CYcle	Byte	%IB300		☑	☑	☑

Figure III-9: Table de variable des entrées

output

	Name	Data type	Address	Retain	Acces...	Writa...	Visibl...
1	open_door1	Bool	%Q0.0		☑	☑	☑
2	autorisation-open_D1	Bool	%Q0.1		☑	☑	☑
3	colon_vert	Bool	%Q0.2		☑	☑	☑
4	colon_rouge	Bool	%Q0.3		☑	☑	☑
5	reset PLC	Bool	%Q0.4		☑	☑	☑
6	reset VFD	Bool	%Q0.5		☑	☑	☑
7	reset cycle_door1	Bool	%Q0.6		☑	☑	☑
8	autr_open door2	Bool	%Q0.7		☑	☑	☑
9	reset Door 1	Bool	%Q1.0		☑	☑	☑
10	reset Door 2	Bool	%Q1.1		☑	☑	☑
11	Run_2	Bool	%Q2.2		☑	☑	☑
12	Run_3	Bool	%Q2.3		☑	☑	☑
13	Run-Table	Bool	%Q2.4		☑	☑	☑
14	R_N_piece	Byte	%QB300		☑	☑	☑
15	EV	Word	%QW302		☑	☑	☑
16	TURN_0°	Bool	%Q1.2		☑	☑	☑
17	TURN_180°	Bool	%Q1.3		☑	☑	☑
18	V_inv1	Real	%QD100		☑	☑	☑
19	V_inv2	Real	%QD102		☑	☑	☑
20	V_inv3	Real	%QD104		☑	☑	☑

Figure III-10 : Table de variable des sorties

III.4.4 Programmation de l'API :

Nous avons mis en œuvre une programmation segmentée en divisant le programme sur plusieurs blocs afin d'obtenir une meilleure maintenabilité. En effet, nous avons créé trois blocs fonctionnel (FB), quatre fonctions (FC) et six blocs de donnés (DB). Afin que le programme soit exécuté, il doit être appelé dans un bloc d'organisation (OB). Les OB sont l'interface entre le programme utilisateur et le système d'exploitation. Dans notre cas nous avons besoin d'une exécution cyclique du programme. De ce fait, nous avons fait l'appel de nos blocs dans l'OB1 qui s'exécute cycliquement, tel que :

a- Main OB1 : est un bloc d'organisation qui sert à l'exécution cyclique de programme.

b- Les Bloc de Fonction FB :

- **Port_état :** pour la commande et la surveillance des portes de la cellule (ouverture,reset..) en retenant l'état de cette dernière (fermer, condamne, les erreurs…).
- **Table :** pour la commande et la surveillance de la table (tourne, rest …etc), en retenant l'état de cette dernière (la position de la table, les erreurs…etc)**.**
- **VFD_Set (Varible Frequency Drive**) : pour la commande et la surveillance des variateurs.

c- Les Fonctions FC

- **Ecreture_PF (Ecretureprofinet) :** sert à émettre les données au robot en écrivent l'information souhaiter sur la sortie de passerelle.
- **EMC /defaulte:** détection les défauts de la cellule, afin de les regroupe dans un défaut globale et générer le mots d'alarme (type/étiqueté).
- **Gestion_cellule :** traitement des informations cumuler afin d'assurer le fonctionnement de la cellule (ROBOT, Outille, Machine …).

c.4 Red_profile: sert à la réceptions des données envoyer par le robot.

d- Les blocs de données DB :

- **Gestion_cellule :** enregistrer tous les états de la cellule
- **HMI_DB :** sauvegarder les donner envoyer par IHM (pour assurer l'échange entre **IHM** et la **CPU**)
- **Port_etat_DB :** débit d'instance du bloc FB « port_etat_DB » sert à enregistrer et modifier les états des variables internes pour les utiliser ultérieurement.
- **Robot/robot :** assure l'enregistrement et l'émission des informations échanger par le **ROBOT** et la **CPU**.
- **Table_DB :** débit d'instance du bloc FB « Table_DB » sert à enregistrer et modifier les états des variables internes pour les utiliser ultérieurement.
- **VFD_set_DB :** débit instance du bloc FB « VFD_set_DB » sert à enregistrer et modifier les états des variables internes pour les utiliser ultérieurement.

Les blocs crées sur step7 professionnel du TIA sont illustrés sur la figure (III.11). Les appels de ces derniers et leurs interactions sont représentés sur la figure (III-12).

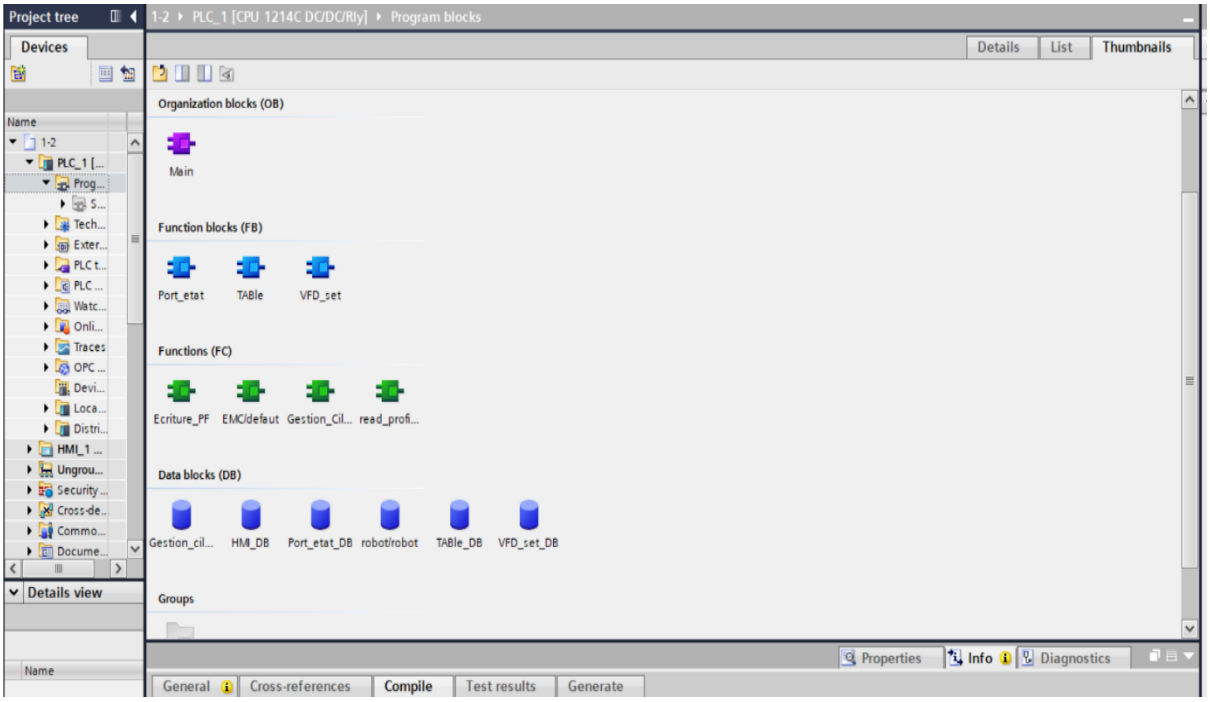


Figure III-11: Illustration des blocs crées

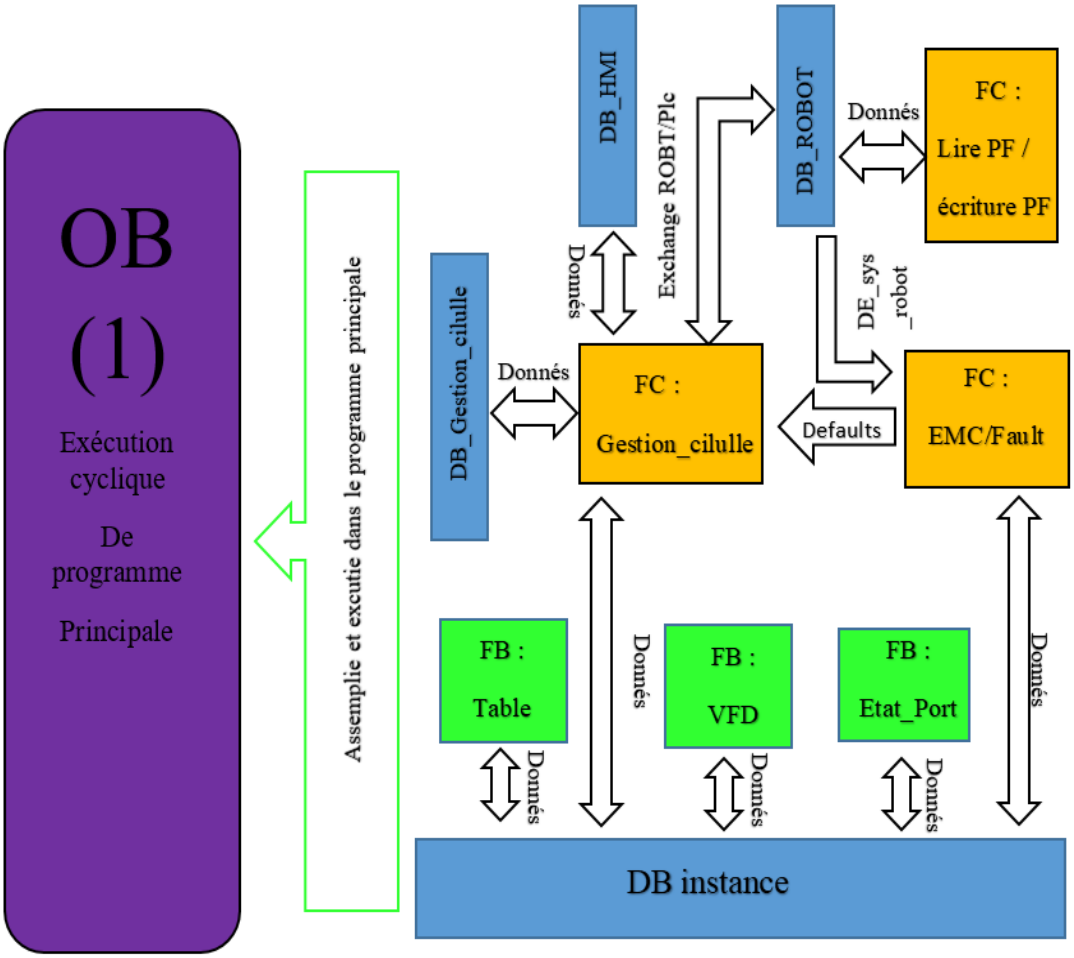


Figure III-12: schéma représentatif des appels des blocs et leurs interactions

Nous avons regroupé les entrées et sorties binaires sur des mots car l'échange de données entre l'API et le robot se font via des mots. Le regroupement des entrées et sorties sont représentés respectivement sur le tableau (III.8) et le tableau (III.9).

Tableau III-8: Bits d'entrées de l'échange Robot/API.

N° de bit															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
N de la pièce															
					EV 10	EV 9	EV 8	EV 7	EV 6	EV 5	EV 4	EV 3	EV 2	EV 1	Stop

Tableau III-9: Bits de sorties de l'échange Robot/API.

N° de bit															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
N de cycle															
				OR 9	OR 8	OR 7	OR 6	OR 5	OR 4	OR 3	OR 2	OR 1	Ar_UR	Auto	Def_sys

La signification des bits des deux tableaux précédents sont définis sur le tableau (III.10)

Tableau III-10: signification des bits d'entrées et sorties.

stop	Stop robot	EV7	Fin de dépose reçue	OR3	Demande de marche Outille 3
EV1	Autorisation d'accès pos A	EV8	Fin de travaille reçu	OR4	Demande Pos A
EV2	Autorisation d'accès pos B	EV9	Exécute Main	OR5	Demande Pos B
EV3	Autorisation d'accès Outille 1	EV10	Reset robot	OR6	Fin de prise
EV4	Autorisation d'accès Outille 2	N_C	N° de cycle	OR7	Fin de travaille
EV5	Autorisation d'accès Outille 3	OR1	Demande de marche Outille 1	OR8	Fin d'Outille
EV6	Fin de prise reçu	OR2	Demande de marche Outille 2	OR9	ROBOT en exécution
N_p	N° de la pièce	Def_sys	Default système	Ar_UR	Arête d'urgence

Nous présentons ci-dessous des exemples de programmes mis en œuvre pour le fonctionnement de notre système. La figure (III.13) représente une partie du programme de fonctionnement des outils, en l'occurrence, leur activation.

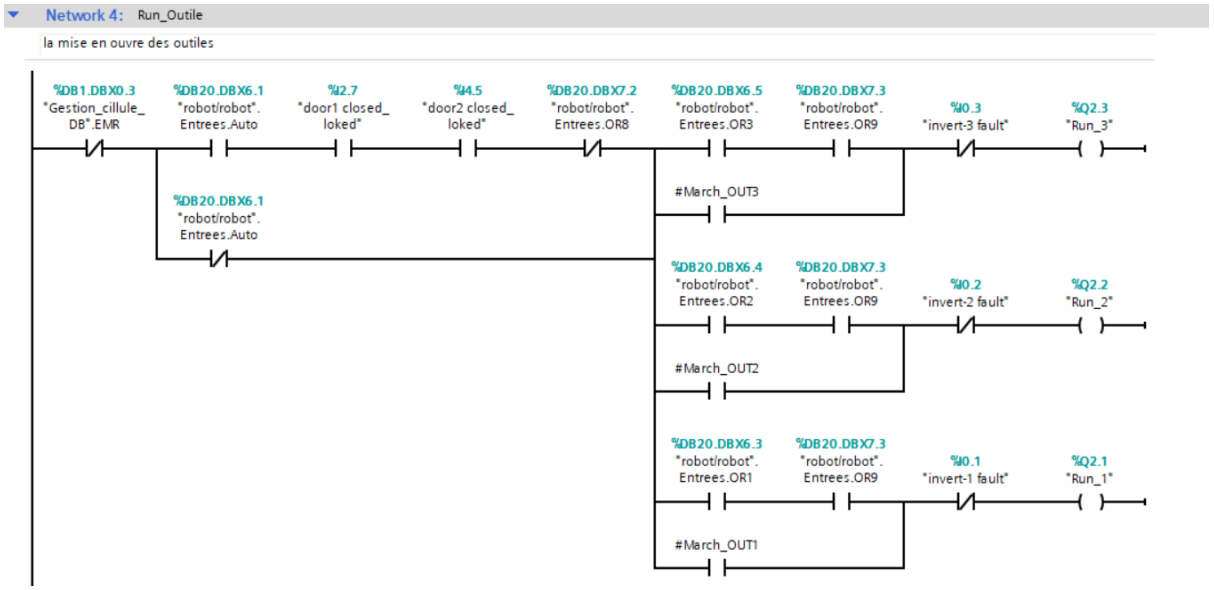


Figure III-13: Partie de mise en œuvre des outils

Le programme de contrôle de vitesse (gestion des variateurs de vitesse) des outils est représenté sur la figure(III.14). Celle-ci montre uniquement l'envoi de la consigne via le mot de commande du variateur 3.

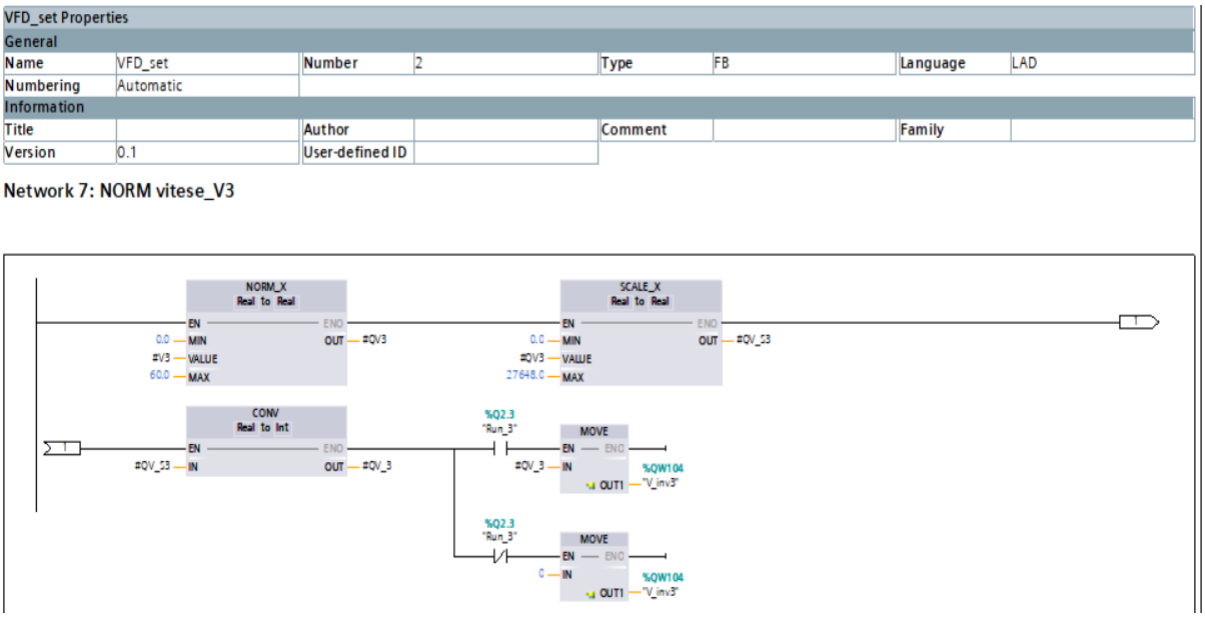


Figure III-14: Programme de l'envoi de la consigne de l'outil (variateur) 3

Un exemple de programme de détection de défauts de la table générer grâce à la combinaison de conditions obtenues des capteurs est représenté sur la figure (III.15)

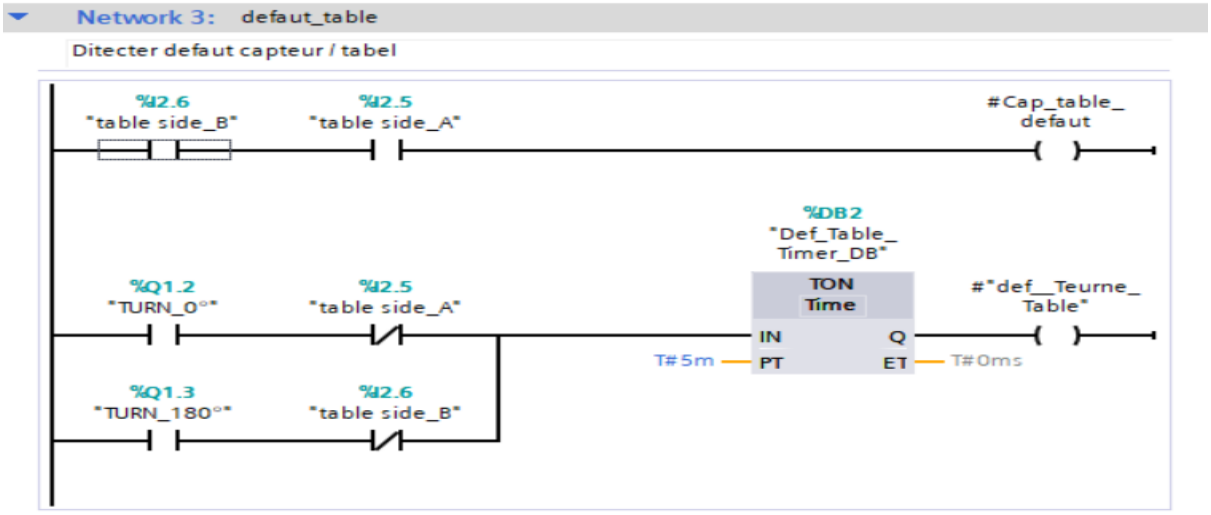


Figure III-15: Programme de détection de défaut de la table.

III.4.5 Configuration de l'interface IHM

Afin de pouvoir donner accès aux opérateurs et avoir une meilleure flexibilité de gestion de notre système nous avons élaboré une IHM avec 7 vues. La navigation entre les vues est illustrée sur la figure (III.16) .

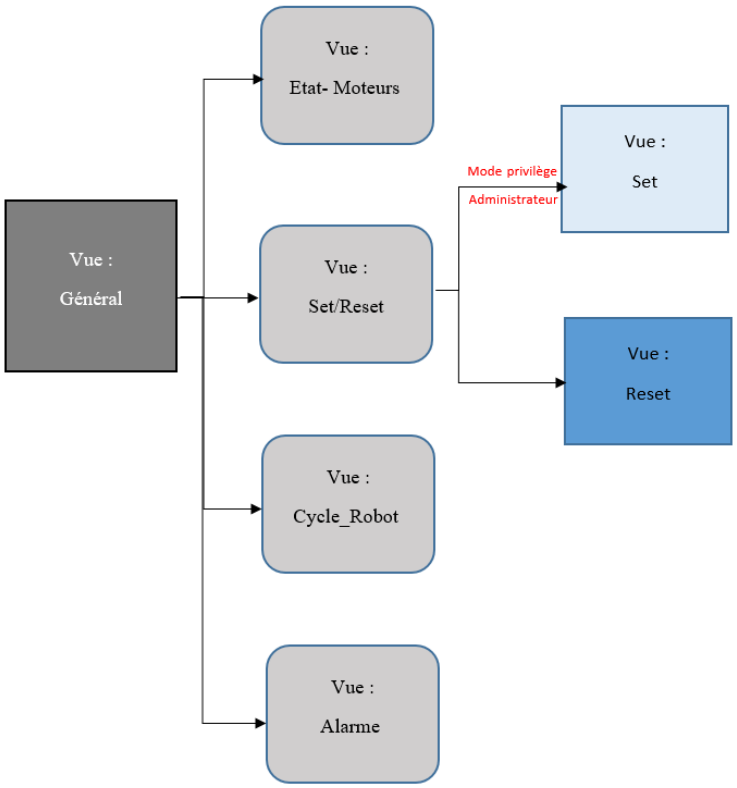


Figure III-16: illustration de la navigation entre les vues

La navigation entre les vues s'effectue grâce à la vue représentée sur la figure (III.17), c'est la vue initiale à partir de laquelle on peut naviguer vers toutes les vues et c'est la vue qui s'affiche en lançant le RUNTIME de l'IHM .

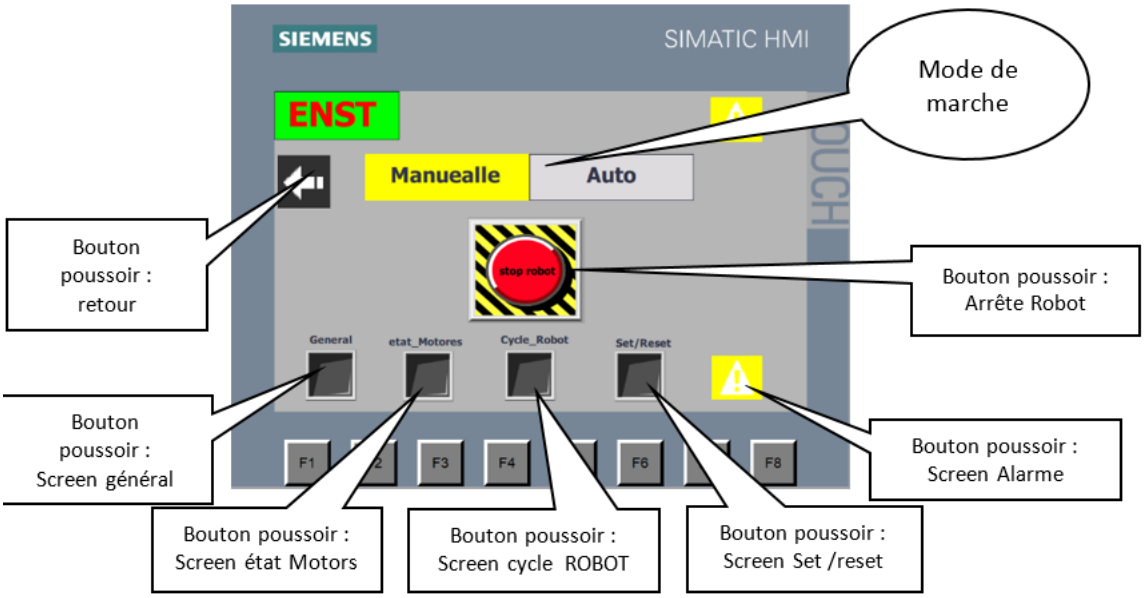


Figure III-17: illustration de la navigation entre les vues

En cliquant sur le bouton « état-Moteurs », on bascule vers la vue de la figure (III.18) qui donne tous les états de moteurs.

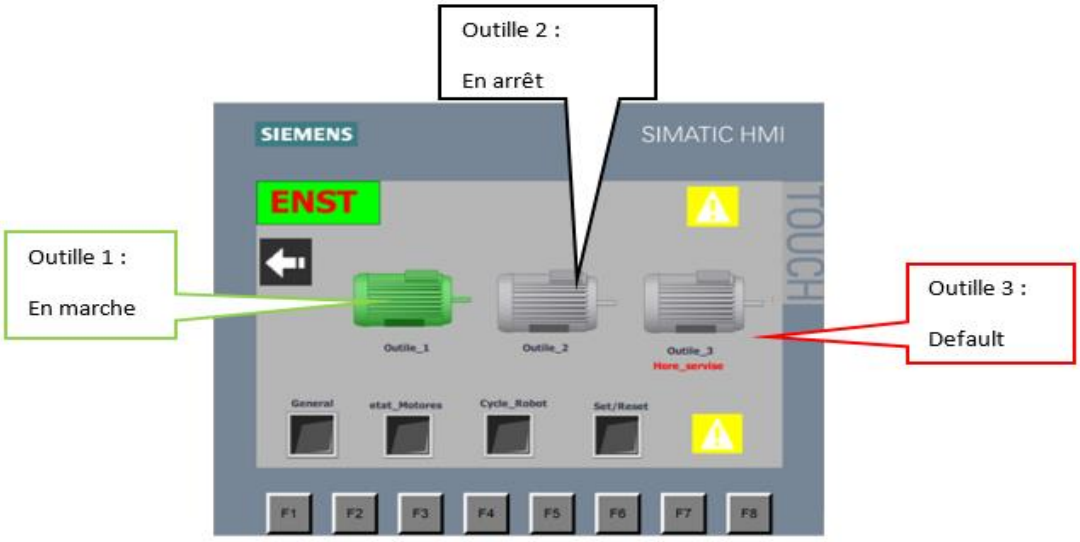


Figure III-18: Vue des états des outils

En cliquant sur le bouton « Cycle_Robot », en bascule vers la vue cycle du robot illustré sur la figure (III.19).

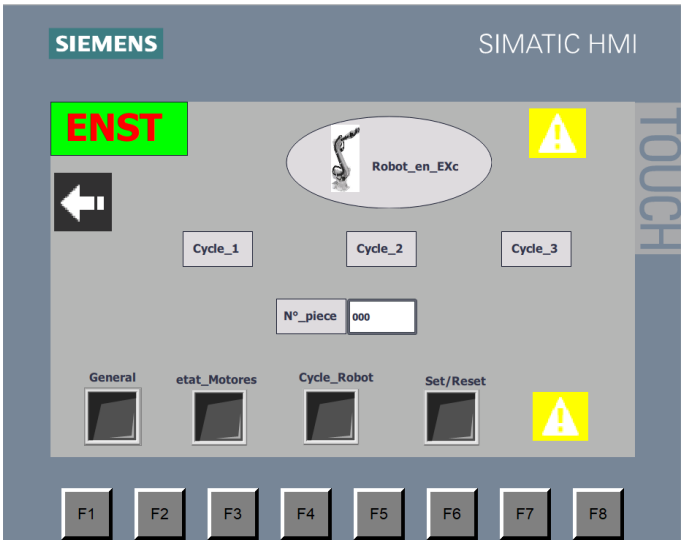


Figure III-19: Vue des cycles du robot

Les interfaces de commande sont accessibles en cliquant sur le bouton « Set/Reset » et à partir de cette vue si nous avons l'autorisation requise en passe en cliquant sur le bouton « Set » à la vue des consignes où on peut donner les vitesses des moteurs et le nombre de pièces. Cependant, en cliquant sur le bouton « Reset » permet de basculer vers la vue de réinitialisation de nos éléments. « Voir Figure (III.20) »

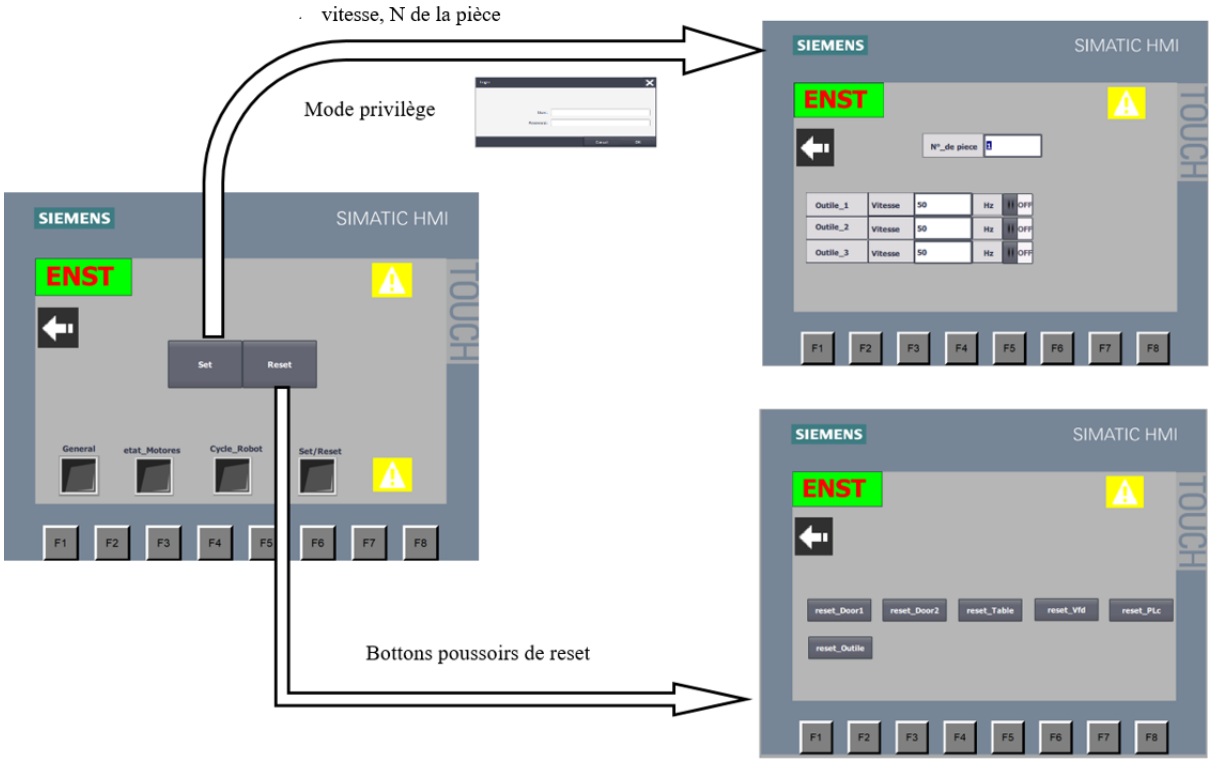


Figure III-20: Vues de commande

Nous disposons également de la vue des alarmes représentée sur la figure (III.21) Nous avons configuré cette vue à partir des défauts que nous avons programmé et les alarmes générées par l'API et par le pupitre, à titre d'exemple, pas de communication entre l'API et IHM.

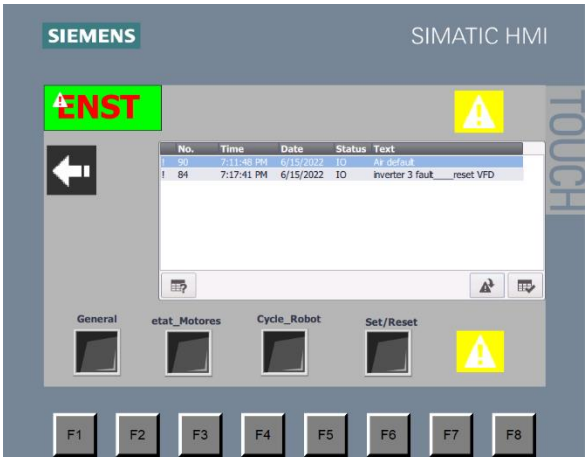


Figure III-21: Vue des alarmes

III.4.5 Programmation de la passerelle sur RCW5 :

Nous avons configuré la passerelle sur RCW5 pour assurer la communication PROFINET avec l'automate [8]Les instructions nécessaires à cette tache sont données sur le tableau (III.8) .

Table III-1: Instruction nécessaires sur le RCW5

Objectif	Programme au Rapide
Création un Indus Net avec adresse 16	INDUSTRIAL_NETWORK: -Name"DeviceNet_Anybus" -Label"Anybus" - Address"16"
Réserver un canal de 32 Octet d'entrées et 32 Octet de sorties sure ce réseaux	DEVICENET_INTERNAL_ANYBUS_DEVICE: -Name"Automate" -VendorName"HMS" - ProductName"Anybus " -OutputSize 32\ -InputSize 32
Création des signaux d'échange avec l'API	EIO_SIGNAL : -Name"N_piece" -SignalType"GI" - Device"Automate"\ -Label"N° de la piece qui doit excuté" - DeviceMap"0-15"\ -Category"safety" -Access"All" -Name"EV_1" -SignalType"DI" - Device"Automate" -Label"Auto_d'accé A"\ -DeviceMap"17" -Access"All" -Name"EV_2" -SignalType"DI" - Device"Automate" -Label"Auto_d'accé B"\ -DeviceMap"18" -Access"All" -Name"EV_3" -SignalType"DI" - Device"Automate"\ -Label"Auto_d'accéOutille 1" - DeviceMap"19" -Access"All" -Name"EV_4" -SignalType"DI" - Device"Automate"\

	-Label"Auto_d'accéOutille 2" -DeviceMap"20" -Access"All" -Name"EV_5" -SignalType"DI" -Device"Automate"\ -Label"Auto_d'accé Outille3" -DeviceMap"21" -Access"All"
Relie les signaux d'état du robot avec la sortie du réseau	SYSSIG_OUT: -Status"EmStop" -Signal"Areet_Urg" -Status"AutoOn" -Signal"Auto" -Status"Error" -Signal"def_sys" -Arg2"T_ROB1" -Status"CycleOn" -Signal"OR_9"
Relie les entrées à la commande d'interruption	SYSSIG_IN: -Signal"Stop" -Action"Stop" -Signal"EV_10" -Action"StartMain" -Arg1"CONT"

Nous présentons ci-dessous des exemples de programmes d'échange de données entre l'API et le Robot :

```
MODULE PLC_exgenge(SYSMODULE)

PROC PLC_tool_take(nummotor\switch ON
| switch OFF)
            TEST motornum
CASE 1: IF Present (ON) Set OR_1;
        IF Present (OFF) Reset OR_1;
CASE 2: IF Present (ON) Set OR_2;
        IF Present (OFF) Reset OR_2;
CASE 3: IF Present (ON) Set OR_3;
        IF Present (OFF) Reset OR_3;
ENDTEST
      ENDPROC
```

Ce programme (**MODULE**) donne les ordres au PLC :

***- PROC PLC-tool-take :** l'alimentions des moteurs d'outils, en attribuant le Numéro d'outil et la demande (ON ou OFF)

```
PROC End_Of_Work ()
SetDO OR_7,1;
IF ((EV_8)=0 )THEN
TPWrite"WAIT FOR EV8";
ENDIF
WaitDI EV_8, 1;
Reset OR_7;
      ENDPROC
PROC PLC_table(\switch A | switch B)
            IF Present (A) reSet OR_5;
Set OR_4;
            IF Present (B) reSet OR_4;
               set OR_5;
ENDPROC
ENDMODULE
```

***- PROC END-Of-Work :** l'envoi d'ordre de fin de travail et l'attente d'un réponse d'accusé de réception

***- PROC PLC-table :** demande de positon de la table (position A ou B)

III.5 IDENTIFICATION ET CREATION DES TACHES DE FINITION

Les pièces traitées passent en dernier par la cellule de finition afin d'obtenir des pièces de qualité. A cet effet, nous méthodes pour y parvenir (figure III.23) où la première méthode est de fixer la pièce et l'outil va être manipulé par le robot ou bien la seconde méthode qui consiste à fixer l'outil et à mouvoir la pièce par le robot.

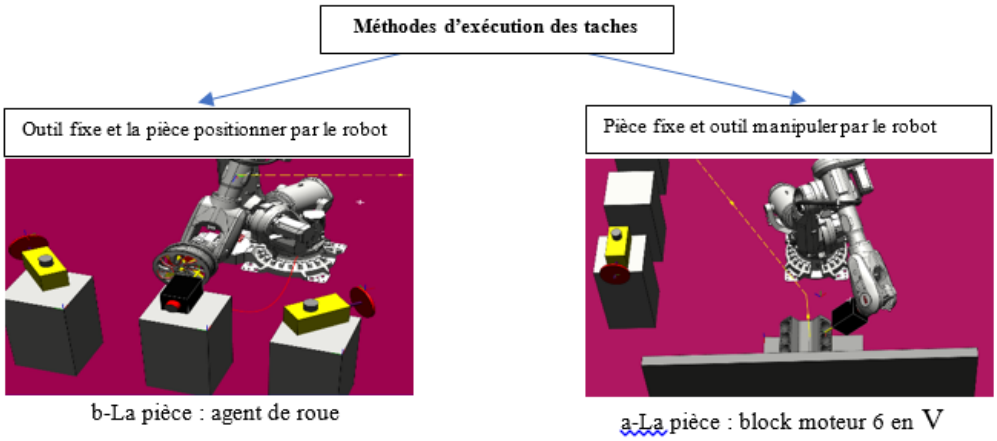
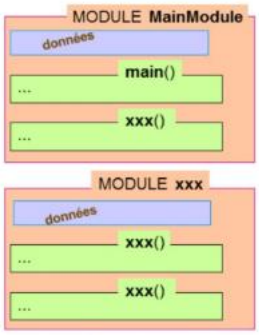


Figure III-22: Méthodes d'exécution des taches

Afin de traiter la pièce et éliminer tous les surplus de la pièce indésirable on va suivre les étapes suivantes :

Au départ, on va définir les surplus de la pièce indésirable qui doivent être éliminé (Tache), chaque tache a un cycle de finitions et chaque cycle a ces propres propriétés (Tools, Works-Object, Vitesse), le Works-Object contient des positions (Target) concrète. Afin de créer une trajectoire à suivre par le robot avec une suite de position et d'instructions du mouvement.



Chaque pièce à son propre module à l'intérieur de chaque module on va déclarer les procédures (les cycles) afin de les appelés dans notre programme principal.

Le programme RAPID de déclaration des cibles et le suivi est données ci-dessous :

MODULE P1

Déclaration des Target

CONSTrobtarget
offs_Tr1_D_2:=[[106.822962877,214.110091085,357.189101692],[0.382680918,-0.923873596,0.000904457,0.003474932],[-1,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];

CONSTrobtarget
Target_410_2:=[[106.822703013,277.940187083,296.121043832],[0.382680965,0.923873576,0.000904455,0.003474999],[0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];

```
CONSTrobtarget
Target_420_2:=[[170.04605743,233.101535778,250.769055945],[0.382680965,-
0.923873576,0.000904455,0.003474999],[0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E
+09]];

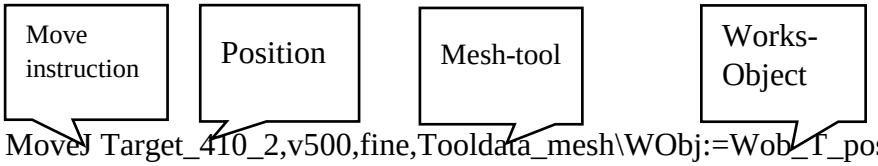
CONSTrobtarget
Target_430_2:=[[106.272686837,188.137354043,206.320438822],[0.382680965,-
0.923873576,0.000904455,0.003474999],[0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E
+09]];

CONSTrobtarget
Target_440_2:=[[43.04933242,232.976005349,251.672426709],[0.382680965,-
0.923873576,0.000904455,0.003474999],[0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E
+09]];

CONSTrobtarget
Target_450_2:=[[106.822703013,277.940187083,296.121043832],[0.382680965,-
0.923873576,0.000904455,0.003474999],[0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E
+09]];

PROCP1_C2 ()

MoveJ offs_Tr1_D_2,v1000,fine,Tooldata_mesh\WObj:=Wob_T_pos2;
```




```
MoveJ Target_410_2,v500,fine,Tooldata_mesh\WObj:=Wob_T_pos2;

MoveC Target_420_2,Target_430_2,v500,fine,Tooldata_mesh\WObj:=Wob_T_pos2;

MoveC Target_440_2,Target_450_2,v500,fine,Tooldata_mesh\WObj:=Wob_T_pos2;

MoveJ offs_Tr1_D_2,v500,fine,Tooldata_mesh\WObj:=Wob_T_pos2;

ENDPROC

ENDMODULE
```

Tous les cycles et le fonctionnement de la cellule sont résumés par l'organigramme suivant :

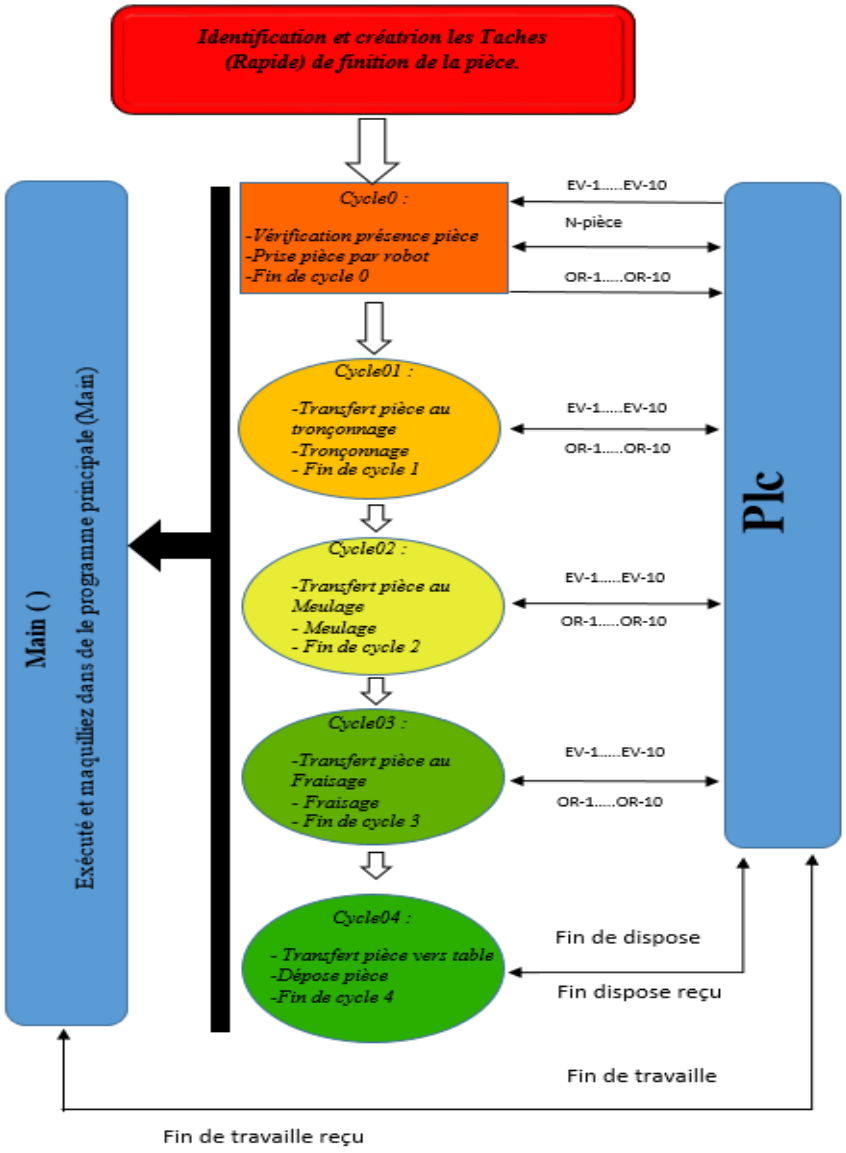


Figure III-23 Organigramme de fonctionnement de la cellule.

CONCLUSION

Nous avons mis en œuvre sur ce chapitre les configurations de nos éléments ainsi que la programmation nécessaire à leur fonctionnement.

Après la réalisation de ces cellules virtuelle (ROBOT-STUDIO, TIA-PORTAL) on peut l'utiliser pour la programmation hors ligne, cette méthode de programmation permet d'améliorer les processus de productions dans un modèle virtuel et de diminuer considérablement l'arrêt de production lors de la création d'une nouvelle référence, cela par préparer des programmes de ces nouvelles références puis les injecter sur les contrôleurs des Robots.

IV. *Chapitre IV : Extraction de la trajectoire d'ébavurage*

IV.1 INTRODUCTION

La génération de mouvements désigne la fonction de calcul des consignes articulaires du robot destinées à réaliser une tache interprétée sous forme de positions successives de l'outil de robot et de contraintes cinématiques ou dynamique. Cette fonction de génération de mouvement peut s'exercer dans l'espace articulaire, ou elle consiste à déterminer une loi horaire pour chaque articulation comme dans l'espace opérationnel où elle vise à déterminer directement une trajectoire cartésienne de l'outil et de son orientation.

Dans la littérature plusieurs techniques sont utilisées pour la génération de trajectoire à suivre par l'outil de robot. Dans ce contexte, nous proposons une méthode basée sur le traitement d'image de la pièce destinée à sa finition des bavures, pour la génération d'une trajectoire dans l'espace opérationnel qui permet d'accomplir la tâche d'ébavurage.

IV.2 GÉNÉRALITÉS

IV.2.1 Vision industrielle

La vision industrielle est l'extraction automatique d'informations à partir d'images numériques. Un environnement typique serait une chaîne de production de fabrication où des centaines de produits circulent sur la ligne devant une caméra intelligente.

IV.2.2 Scanner 3D

Le scan 3D est un processus d'analyse d'un objet du monde réel, destiné à collecter toutes les données nécessaires afin de recréer sa forme et son apparence, de manière numérique. Grâce à ce processus, l'objet peut devenir un modèle 3D.

Le scan laser 3D est certainement la technique de numérisation 3D la plus répandue et la plus utilisée. Cela consiste en a capturer numériquement la forme de l'objet à l'aide d'une lumière laser pour obtenir une représentation numérique de l'objet réel. Ces scanners 3D sont capables de mesurer des détails très petits et de capturer des formes libres pour générer des nuages de points extrêmement précis sous un format STL, Cette technique de balayage laser est parfaite pour la mesure et l'inspection de géométries complexes [12]

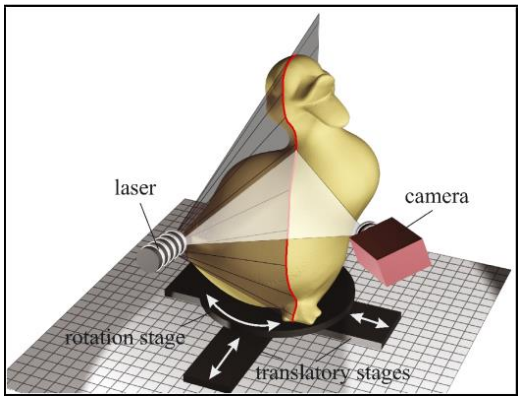


Figure IV-1 : Principe de fonctionnement d'un scanner3D [12]

IV.2.3 Format STL :

STL est un format de fichier natif du logiciel de CAO de stéréo lithographie créé par 3D Systems, il a plusieurs acronymes tels que "Standard Triangle Language" et "Standard Tessellation Language". Les fichiers STL sont compatibles avec les logiciels de CAO et

enregistrent les géométries en inclinant la surface avec des millions de triangles. Ces triangles créent un maillage sans interstices ni chevauchements, chaque triangle est défini par trois points et une norme de surface.

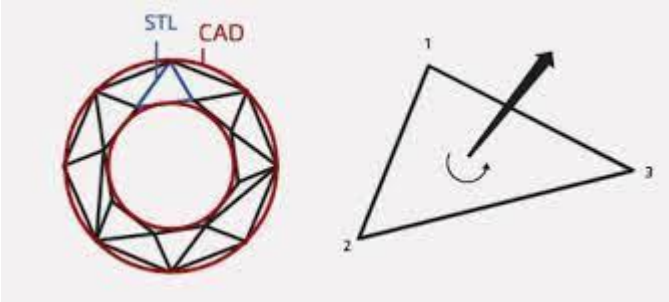


Figure IV-2Format STL

IV.2.4 Points Cloud (PCL) :

Un nuage de points est un ensemble de points de données dans l'espace. Les points peuvent représenter la surface d'une forme ou un objet 3D. Chaque position de point a son ensemble de coordonnées cartésiennes (X, Y, Z), couleur, luminance et la normale.

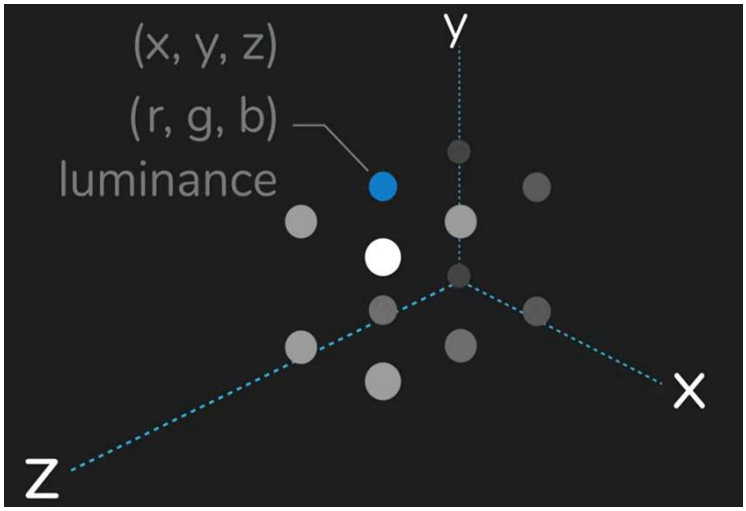


Figure IV-3:représentation de PCL dans l'espace

VI.3 PROGRAMME DE TRAITEMENT D'IMAGE

La trajectoire suivie par l'outil du robot qui permet l'achèvement de l'opération de l'ébavurage réalisée par le robot n'est pas programmée d'une manière automatique, mais il s'agit d'un ensemble de points sur la pièce enregistrés suivi par l'outil point par point.

Cette tâche qui est pénible pour l'opérateur surtout quand il s'agit de plusieurs et nouvelles pièces mécaniques pour effectuer l'opération de l'ébavurage. Pour améliorer l'obtention de la trajectoire à suivre, nous avons opté pour la vision industrielle en en récoltant les informations par le scanner et avoir par la suite un modèle 3D de la pièce.

Ensuite un algorithme de traitement d'image 3D est réalisé par Matlab, dans le but de déterminer le surplus à ébavurer de la pièce.

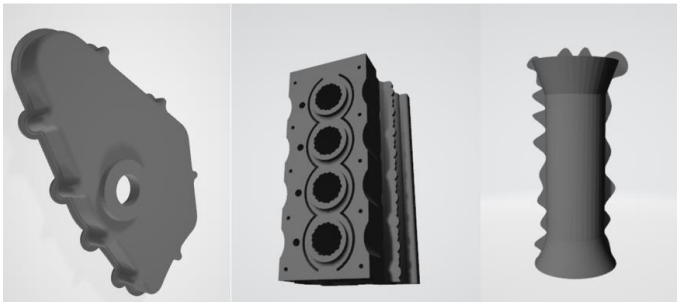


Figure IV-4 : les pièces à traiter (1, 2,3)

Le principe de l'algorithme conçu consiste à récupérer dans une matrice les données suivantes : les coordonnées cartésiennes et la normale des points de l'image PCL.

Tableau IV-1Les données de la pièce (x, y, z, n) excrètes de la forme PCL

299973x6 double

	1	2	3	4	5	6	7
1	261.1913	265.2874	44	9.7752e-04	9.7752e-04	-1.0000	
2	265.2070	278.3825	44	9.7752e-04	9.7752e-04	-1.0000	
3	268.7789	276.3005	44	9.7752e-04	9.7752e-04	-1.0000	
4	263.8061	267.1920	44	9.7752e-04	9.7752e-04	-1.0000	
5	261.5514	267.6667	44	9.7752e-04	9.7752e-04	-1.0000	
6	263.7923	270.5346	44	9.7752e-04	9.7752e-04	-1.0000	
7	262.1344	267.1293	44	9.7752e-04	9.7752e-04	-1.0000	

L'algorithme est basé sur la segmentation de l'image qui illustre la pièce, et afin de déterminer les additions indésirables nous allons utiliser une méthode de comparaison en exploitant les points de chaque segment de l'image avec celles des segments de l'image de référence (la pièce finie sans défauts). Ces points sont générés d'une façon aléatoire, autrement dit les points récupères d'une image quelconque ne sont pas forcément identiques aux points d'une autre image de la même pièce. Pour cela on va définir un seuil que l'on considère comme une limite des points indésirables. Les tableaux VI.2 montrent la différence des distributions des positions dans les matrices.

Tableau IV-2: La distribution des positions (x, y, z) de (im1, im2)

	1	2	3
1	-8.0274	213.4060	3.9106
2	-7.9047	214.2175	3.1931
3	-7.7767	215.0639	0.9080
4	-7.8940	214.2881	5.2508

	1	2	3
1	261.1913	265.2874	44
2	265.2070	278.3825	44
3	268.7789	276.3005	44
4	263.8061	267.1920	44

VI.3.1 La segmentation

La segmentation se fait par rapport aux trois axes (x, y, z), on obtient une forme cubique dans chaque segmentation et puis on compare cet élément au segment de l'image de référence.

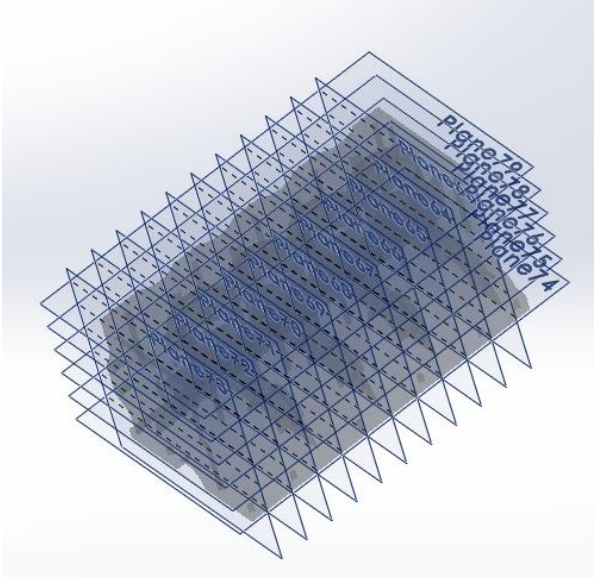


Figure IV-5 : Prototypage de la segmentation

VI.3.2 Extraction de l'addition indésirable

La comparaison est basée sur le calcul de la distance entre chaque point du segment avec ses points de voisinage dans un volume de sphère de rayon égale à la valeur du seuil définie, si cette distance est inférieure à ce rayon c'est-à-dire tout point appartient à la sphère on le considère comme un point valide (figure IV-6). À la fin du traitement de chaque segment (cube) tous les points qui n'appartiennent à aucune sphère sont considérées comme des points de bavures. La sphère est définie par l'équation suivante :

$$(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2 < Seuil^2 \qquad \text{(VI.1)}$$

i : i^{ème} point de la pièce de référence.

j : j^{ém} point de la pièce traité.

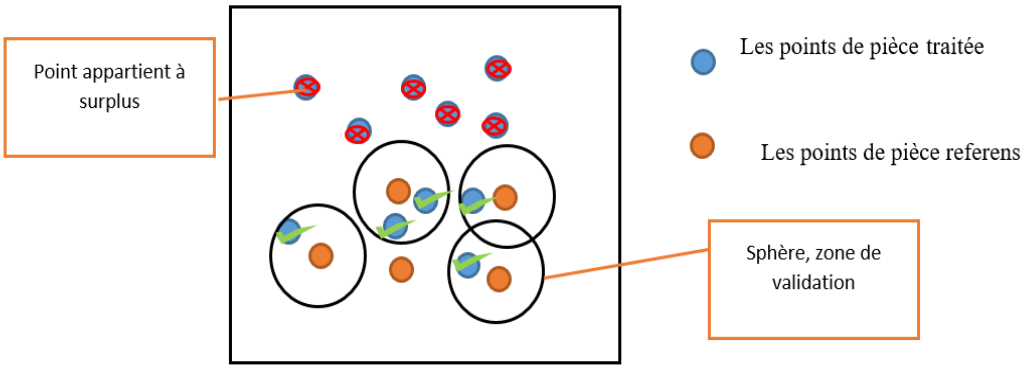


Figure IV-6: Validation d'un point

La figure (VI.7) illustre le résultat obtenu de l'extraction du surplus sur la pièce de test.

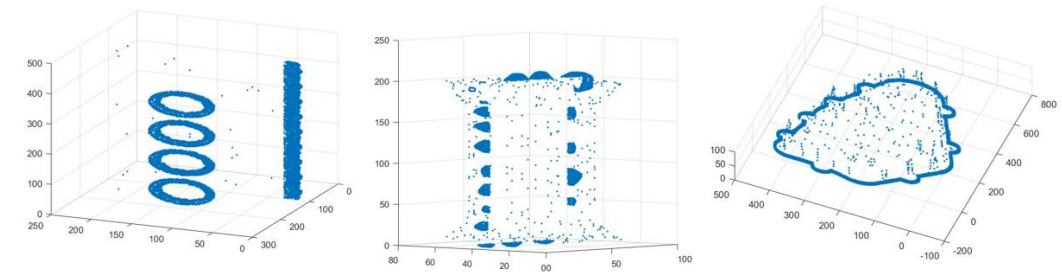


Figure IV-7: Résultats obtenus sur les pièces de test

VI.3.3 Filtrage

Le filtrage est basé sur le comptage de nombre de points existants dans chaque segment, si ce dernier contient un nombre inferieur la valeur μ (équation 2) on le considère comme un bruit à éliminer sinon il appartient aux données. Le calcul du nombre de points est donné par l'équation (2).

$$\mu = \frac{Nombre\ totale\ des\ pointes\ \times Volume\ de\ segment}{la\ surfase\ totale\ de\ la\ pièce \times \rho} \qquad \text{(VI.2)}$$

Telle que $1 \leq \rho \leq 1 + (K \times seuil)$

K >0

À partir des essais on trouve : $\mu_1 = 6$, $\mu_2 = 3$, $\mu_3 = 5$, seuil=2 mm.

La figure (VI.8) et (VI.9) montre la classification des donnés par le calcul du nombre de voisin, et la figure (VI.10) affiche les résultats de filtrage de la pièce mécanique.

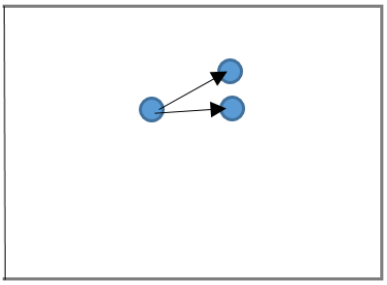


Figure IV-8: Bruit

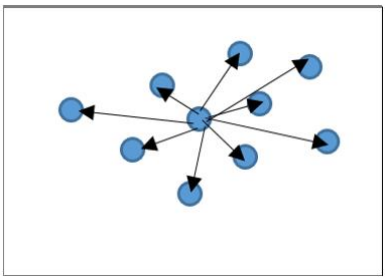


Figure IV-9: Donnée

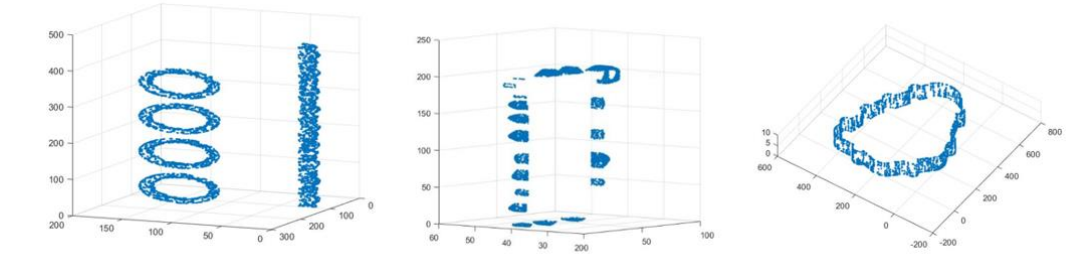


Figure IV-10: les résultats du filtrage

Après le filtrage on va réarranger la distribution des points restants en calculant les points pondérés, équation (3).

$$P_{moy} = \left(\frac{\sum_i^N \alpha_i x_i}{\sum_i^N \alpha_i}, \frac{\sum_i^N \beta_i y_i}{\sum_i^N \beta_i}, \frac{\sum_i^N \gamma_i z_i}{\sum_i^N \gamma_i}\right) \quad \alpha = \beta = \gamma = 1 \tag{VI.3}$$

$$n_{moy} = \left(\frac{\sum_i^N n_{xi}}{N}, \frac{\sum_i^N n_{yi}}{N}, \frac{\sum_i^N n_{zi}}{N}\right) \tag{VI.4}$$

IV.3.4 L'emplacement de l'addition sur la pièce de référence

À partir de résultat de la distribution, on fait apparaitre l'emplacement des bavures sur le corps de la pièce de référence. Pour y arriver, on segmente encore une fois la pièce avec des paramètres différents, puis on fait la régression (ajustement multiple) de chaque groupe de point pour construire un plan. Comme le montre la figure (VI.11).

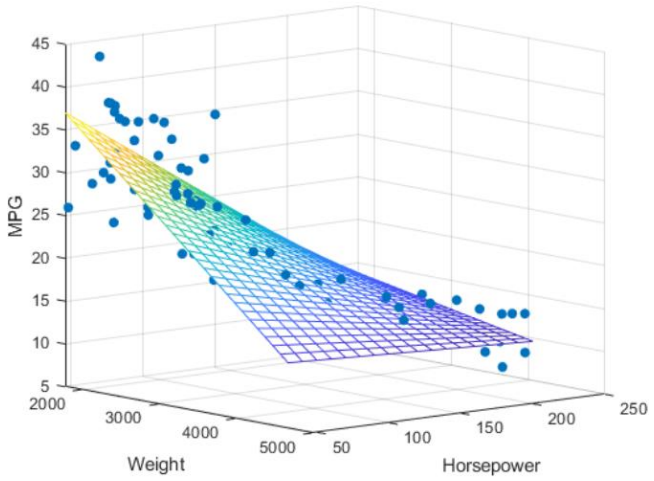


Figure IV-11: Méthode de la régression linéaire [13]

Plan : dans un repère (i, j, k), Tout plan ρ de l'espace admet une équation de la forme $ax + by + cz = d$ avec $(a, b, c) \neq (0,0,0)$. Alors l'ensemble des points M de coordonnées (x, y, z) vérifiant $ax + by + cz = d$est un plan et $n = (a, b, c)$normal à ρ.

Ensuite on prend les points de la pièce de référence (segment) qui appartient au plan c'est à-dire l'intersection du plan et la pièce. Voir l'inégalité donnée par l'équation (VI.5) .Les points qui vérifient la condition suivant :

$$|ax + by + cz - d| \leq m \tag{VI.5}$$

Les points envoyés au robot doivent contenir la position $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$ et l'orientation $(\vec{\boldsymbol{i}}, \vec{\boldsymbol{j}}, \vec{\boldsymbol{k}})$, Mais dans la norme de PCL n'a qu'un seul vecteur d'orientation $\vec{\boldsymbol{k}}$, donc on peut utiliser la normale $\overrightarrow{\boldsymbol{n}}$ du plan de régression comme un vecteur d'orientation. Les deux autres composantes du vecteur d'orientation existent s'ils vérifient les conditions des inégalités (VI.6) et (VI.7).

$$\vec{j} = \vec{n} \text{ Si } \quad max(y) - min(y) < max(x) - min(x) \tag{VI.6}$$
$$\vec{i} = \vec{n} \text{ Si } \quad max(x) - min(x) < max(y) - min(y) \tag{VI.7}$$

Si on trouve l'une des composantes du vecteur d'orientation on utilise la relation $(\vec{\boldsymbol{\imath}} \wedge \vec{\mathbf{j}}) = \vec{\boldsymbol{k}}$ pour trouver le dernier vecteur. La figure (VI.10) montre le résultat de la fonction get-real-contour.

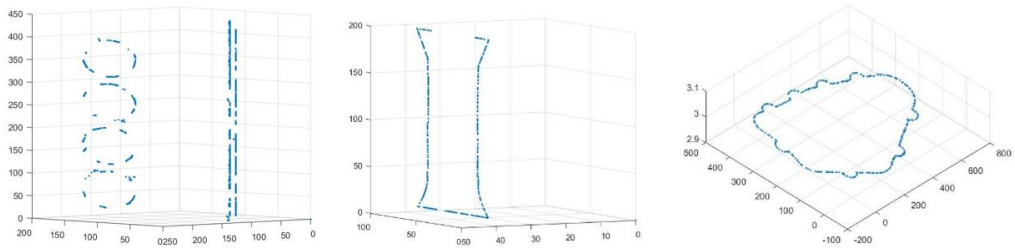
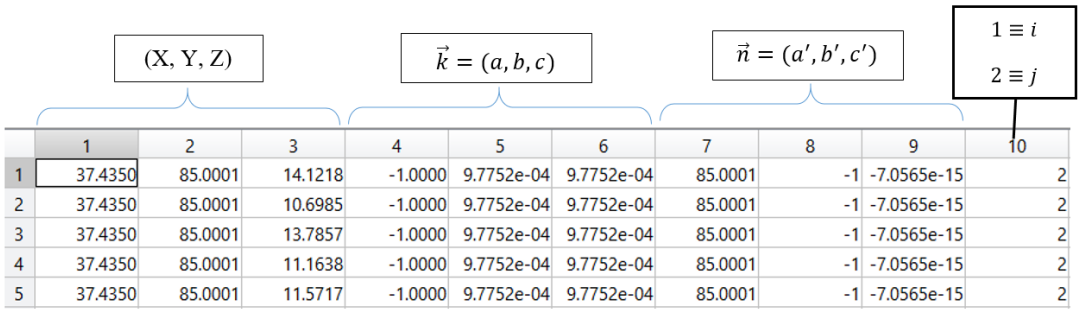


Figure IV-12: résultats de L'emplacement de l'addition sur la pièce de référence

La figure (VI.13) montre les données résultantes à envoyer au robot afin de rendre la pièce bien finie.



	1	2	3	4	5	6	7	8	9	10
1	37.4350	85.0001	14.1218	-1.0000	9.7752e-04	9.7752e-04	85.0001	-1	-7.0565e-15	2
2	37.4350	85.0001	10.6985	-1.0000	9.7752e-04	9.7752e-04	85.0001	-1	-7.0565e-15	2
3	37.4350	85.0001	13.7857	-1.0000	9.7752e-04	9.7752e-04	85.0001	-1	-7.0565e-15	2
4	37.4350	85.0001	11.1638	-1.0000	9.7752e-04	9.7752e-04	85.0001	-1	-7.0565e-15	2
5	37.4350	85.0001	11.5717	-1.0000	9.7752e-04	9.7752e-04	85.0001	-1	-7.0565e-15	2

Figure IV-13 : la matrice retourner par la fonction GET-REAL

Dans le langage RAPID l'orientation est un vecteur $q = [q_1 , q_2 , q_3 , q_4]$ tel que :

$$q_1 = \frac{\sqrt{x_1 + y_2 + z_3 + 1}}{2}$$

$$q_2 = \frac{\sqrt{x_1 - y_2 - z_3 + 1}}{2}$$

$$q_3 = \frac{\sqrt{y_2 - x_1 - z_3 + 1}}{2}$$

$$q_4 = \frac{\sqrt{z_3 - x_1 - y_2 + 1}}{2}$$

$\vec{\imath} = (x_1 , y_1, z_1)$, $\vec{\jmath} = (x_2 , y_2, z_2)$, $\vec{k} = (x_3 , y_3, z_3)$

Remarque :

Cette méthode de calcul des vecteurs d'orientation $(\vec{\boldsymbol{\imath}}, \vec{\boldsymbol{\jmath}}, \vec{\boldsymbol{k}})$ n'est pas toujours valable parce que l'orientation est l'un des grands problèmes de la recherche.

IV.4 L'ENVOI DE DONNÉES VERS LE ROBOT

Premièrement, on doit réarranger l'emplacement des points dans la matrice pour avoir une suite de points qui formera une trajectoire, et cela par le choix du point de départ et allant au celui le plus proche.

Suivi par la conversion des DATA numérique de type double (position) à une matrice de type STRUCT, pour qu'elle soit compatible avec le langage RAPID. Donc nous devons crier une fonction Convertir-Data (DATA).

Enfin, un programme (MATLAB/RAPID), pour la configuration de la communication (TCP/IP) server-client et la synchronisation de la transmission des données, est conçue.

On utilise la fonction Matlab (TCP-IP) pour la création de l'Object IP de même adresse IP de l'interface PC, la synchronisation est assurée par la création de la fonction « WHAIT -CHAR», et en RAIPD nous créions un SOCKET (server-client).

Les positions sont envoyées point par point, si le robot atteint la trajectoire à ce point, il demande une nouveau point, et l'opération de l'envoi et la réception des données continue jusqu'à recevoir une information que l'opération est terminé. La figure (VI.14) montre la structure du programme de l'échange des donnés entre Matlab et Robot-Studio.

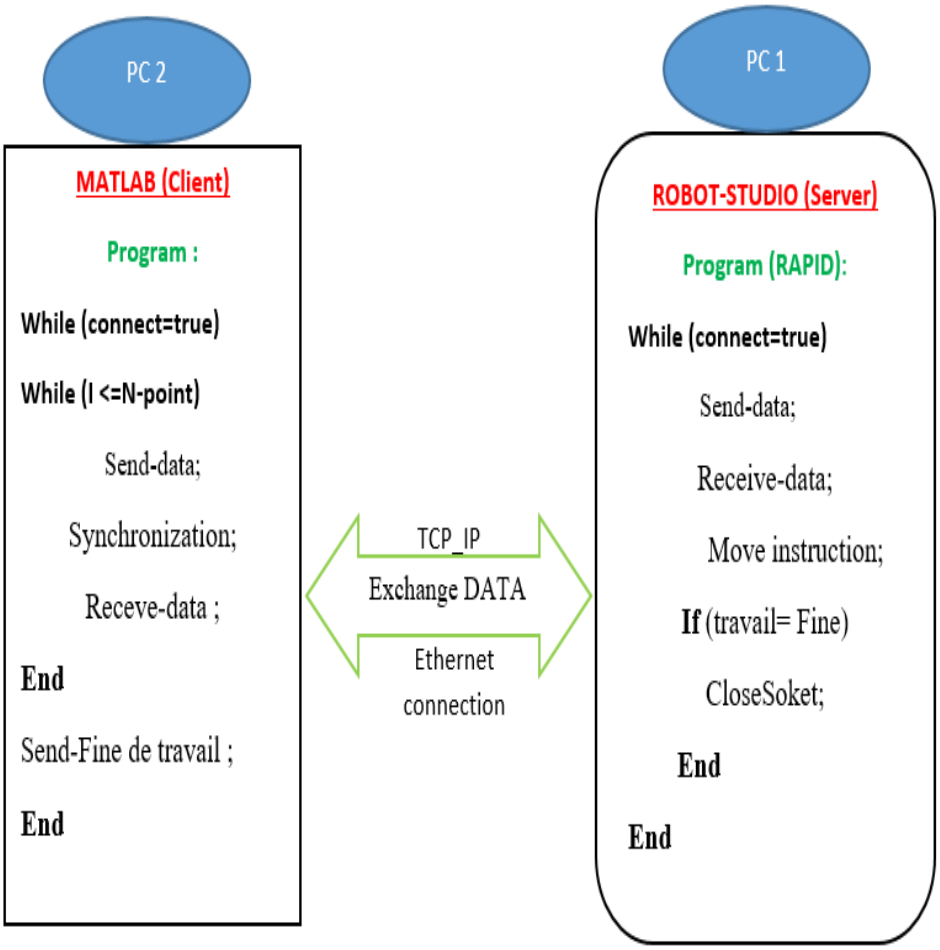


Figure IV-14: structure de programme de communication (Matlab/Robot-Studio)

IV.5 SIMULATION

La figure (IV-15) montre la réalisation d'une communication Ethernet entre PC1 et PC2

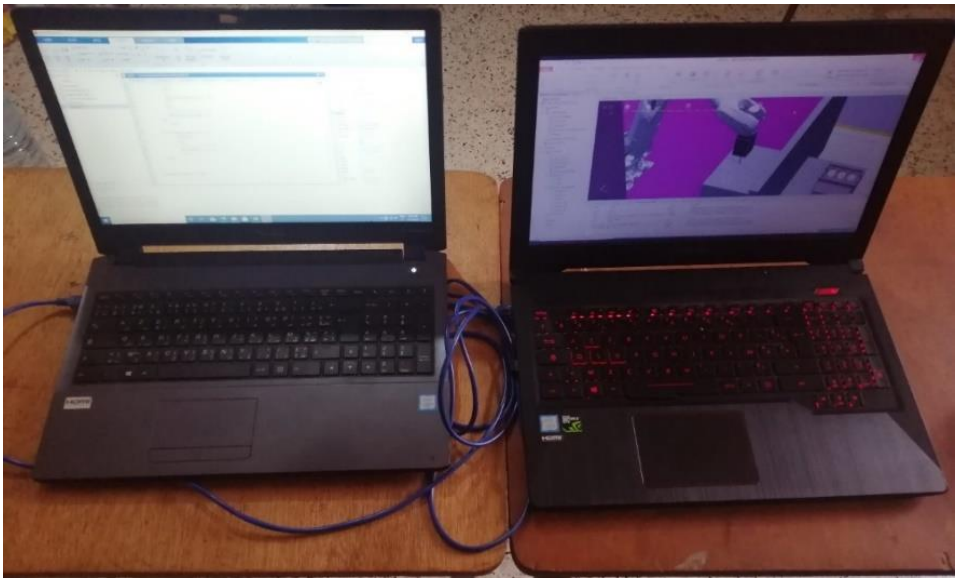


Figure IV-15: la réalisation d'une communication Ethernet entre PC1 et PC2

La simulation est faite avec la première pièce :

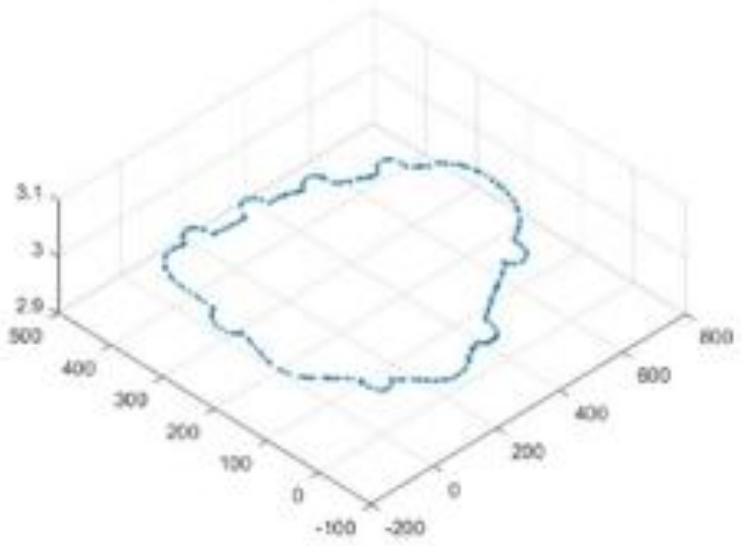


Figure IV-16: le contour de finition de la pièce 1 en mm.

La figure (IV-17) montre la trajectoire finale de (TCP) ROBOT envoyé par Matlab. Les coordonnées de trajectoire sont montrées au-dessous de la simulation.

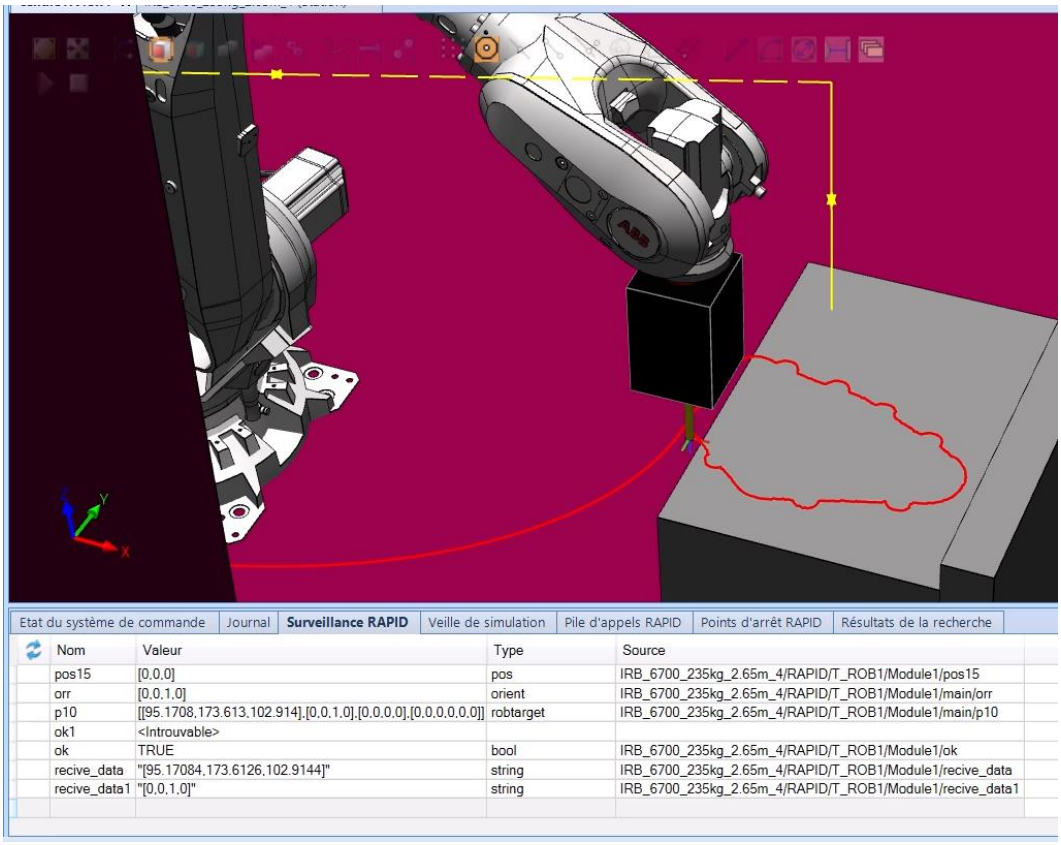


Figure IV-17: trajectoire finale résultante

CONCLUSION :

Ce chapitre est consacré à la présentation de la méthode de génération de la trajectoire suivi par l'effecteur du robot ABB-IRB 6700 pour la finition d'une pièce mécanique muni de bavures. La méthode est basée sur la segmentation de l'image, et afin de déterminer les additions indésirables, une méthode de comparaison, en exploitant les points de chaque segment de l'image avec celles des segments de l'image de référence, est conçue. Suivi par un filtrage des points lointains. Enfin, la méthode de régression linéaire est utilisée pour déterminer la finalité de la trajectoire à suivre.

CONCLUSION GENERALE

Le travail présenté dans ce mémoire est le fruit d'un stage effectué au sein du département Fonderie de la Société Nationale de Véhicules Industrielles (SNVI), sise à la zone industrielle de Rouiba. L'objectif de départ était d'apporter une amélioration au processus d'ébavurage de pièces mécaniques à base de la dernière génération du robot de type ABB-IRB 6700, de capacité de manutention de 235Kg et de portée de 2.65m. En effet, la pénibilité de la programmation de la trajectoire des bavures due au choix des points et la profondeur d'usinage rend la finition de ces pièces ne se réalise pas correctement qu'après plusieurs essais, ce qui engendre de pertes considérables en matière et en temps.

Pour répondre à ces objectifs, la première étape est de modéliser le robot ABB-IRB 6700. Dans la littérature le modèle géométrique directe et inverse de ce robot n'existe pas, pour cela nous avons utilisé le formalisme de Denavit-Hartenberg pour calculer les deux modèles. Les contraintes étaient de trouver la solution aux équations issues de la méthode de Paul pour résoudre le problème inverse. En fin, une validation est effectuée en introduisant des valeurs articulaires et calculer par la suite la position dans l'espace opérationnel en utilisant le MGD. Ces valeurs opérationnelles sont ensuite introduites dans le MGI pour calculer les valeurs articulaires. L'erreur était de l'ordre de zéro.

Une étude détaillée est ensuite effectuée sur la cellule robotisée afin de cerner le travail demandé. Pour ces fins, nous avons présenté le cycle de fonctionnement de système depuis la présence de la pièce sur la table tournante jusqu'à la fin de la tâche d'ébavurage par le robot et l'acquittement de la pièce de la cellule. Toutes ces opération sont contrôlé par un automate de type Siemens S1200. Nous avons conçu le programme de contrôle en utilisant le logiciel TIA Portal, ainsi que IHM qui sert à la supervision du système entier.

Afin de simplifier la tâche de programmation, nous avons modélisé l'environnement de la cellule robotisée sous le logiciel RobotStudio. Ce dernier est un puissant outil de simulation des robots, que la firme ABB a développé afin de résoudre des problèmes en robotique hors ligne, et l'utilisateur a toutes les garanties que la solution proposée fonctionnera en virtuel comme en réalité. Quelques simulations de suivi de trajectoires par l'effecteur du robot ont été effectuées sur des pièces mécaniques.

Enfin, et dont l'objectif d'améliorer le processus de finition de pièce mécanique, nous avons proposé une solution qui tient en compte la simplicité de sa mise en œuvre et qui élimine toutes les contrainte et les difficultés rencontrées par les opérateurs dans la solution actuelle. En effet, la solution est basée sur un traitement d'image de la pièce mécanique. La première étape est de construire le nuage des points indésirables par une comparaison avec une pièce de référence. Afin de bien situé la trajectoire des bavures, une segmentation de l'image est effectuée suivi par un filtrage de l'image afin d'éliminer les points distant de la trajectoire des bavures. Le résultat obtenu après ce traitement était un ensemble de points (Px, Py, Pz) et les angles d'orientations du repère lié à l'organe terminal du robot qui doivent être transmis au RobotStudio pour effectuer des simulations de suivi de trajectoire. Pour cette raison, nous avons développé un programme de communication entre Matlab et RobotStudio. La transmission est synchronisée par l'envoi des données pour chaque point de la trajectoire des bavures. Des

simulations sur RobotStudio ont été effectuées, et les résultats obtenus sont très satisfaisant, et la trajectoire des bavures a été bien suivie par l'effecteur du robot.

L'un des inconvénients les plus fréquent pour cette méthode est le fait d'être confronté à une pièce qui contient des trajectoires complexes, ainsi le niveau de complexité de la pièce peut engendrer un croisement de trajectoires ce qui provoque une diminution de la précision. Car la méthode proposée pour le calcul des vecteurs d'orientation n'est pas toujours valable, ce qui reste toujours un grand problème de la recherche, dans ce cas il faut élaborer un algorithme pour séparer les différentes trajectoires des bavures.

Le travail présenté dans ce mémoire a le potentiel de générer des travaux futurs qui permettront d'améliorer encore les résultats obtenus. On peut envisager :

- D'améliorer la solution obtenue pour l'ébavurage de n'importe quelle pièce mécanique.
- Introduire des algorithmes d'identification de contour dans la solution proposée.
- Introduire la vision industrielle et reconnaissance de forme pour générer la trajectoire des bavures.

Bibliographie

[1] ABB Robotics, Manuel du produit 6700;ID du document: 3HAC044266-004, Se-721 68 Västerås, 2013-10-15.

[2] ABB Robotics, Product specification Controller IRC5 with FlexPendant ;Document ID: 3HAC041344-001;Revision: C, Se-721 68 Västerås, 2013.

[3] S. E. Automation, Machine Safety: Functional Safety & Implementation of the New Machinery Directive, 2019.

[4] siemenc, SIMATIC s7 Automate programmable S7-1200;Manuel système;Numéro de référence du document: 6ES7298-8FA30-8CH0, NÜRNBERG ALLEMAGNE, 2014.

[5] T. Alexander, Unidrive M300 Quick Start Guide, Newtown, Powys. U: VP Technology, 29th May 2014.

[6] A. Bus, «Get Connected to Industrial Ethernet and Field Bus,» [En ligne]. Available: https://www.anybus.com.

[7] M. Bowne, «what is Profinet- Profinet explained,» [En ligne]. Available: https://us.profinet.com/profinet-explained/.

[8] «Kukatpally, Hyderabad DIFFERENCE BETWEEN DENAVIT -HARTENBERG (D-H) CLASSICAL AND MODIFIED CONVENTIONS FOR FORWARDKINEMATICS OF ROBOTS WITH CASE STUDY».

[9] «S. Hayati, R. Volpe, P. Backes, J. Balaram, R. Welch, R. Ivlev, G. Tharp, S. Peters, T. Ohm, R. Petras, The Rocky 7 Rover: A Mars Sciencecraft Prototype, Proceedings of the IEEE International Conference on Robotics and Automation 1997 (ICRA'97), Albu».

[10] «Technical reference manual RAPID Instructions, Functions and Data types RobotWare 5 Document id: 3HAC050917-001Revision H».

[12] Douglas Lanman Gabriel Taubin Build Your Own 3D Scanner: SIGGRAPH 2009 Course Notes.

[13] Mathworks. [En ligne]. Available: https://fr.mathworks.com.

ANNEXE

Rapid : le programme est très long donc on va illustrer les parties les plus importantes.

1-la table

```
MODULE module_tabl
PROC T_pos1()
SetDO tabl_pos2,0;
SetDO tabl_pos1,0;
SetDO tabl_pos1,1;
WaitTime 1.5;
!SetDO tabl_pos1,0;
ENDPROC
PROC T_pos2()
SetDO tabl_pos1,0;
SetDO tabl_pos2,0;
SetDO tabl_pos2,1;
WaitTime 1.5;
!SetDO tabl_pos2,0;
ENDPROC
ENDMODULE
```

2-porteur du pièce

```
MODULE open_close_prt
PROC open_Pr()
SetDO close_prt,0;
SetDO open_prt,1;
WaitTime 2;
ENDPROC
PROC close_Pr()
SetDO open_prt,0;
SetDO close_prt,1;
WaitTime 2;
ENDPROC
ENDMODULE
```

3-att-déatt-pièce

```
MODULE att_déatt_piece
PROC attache_piece()
MoveJ P_offs_piece,v1000,fine,tool0\WObj:=Wob_T_pos2;
MoveL P_piece,v1000,fine,tool0\WObj:=Wob_T_pos2;
SetDO déatt_piece,0;
SetDO close_prt,0;
SetDO open_prt,1;
setDo att_piece,1;
Movel P_offs_piece,v1000,fine,tool0\WObj:=Wob_T_pos2;
ENDPROC
PROC déattache_piece()
MoveJ P_offs_piece,v1000,fine,tool0\WObj:=Wob_T_pos2;
MoveL P_piece,v1000,fine,tool0\WObj:=Wob_T_pos2;
SetDO att_piece,0;
SetDO open_prt,0;
SetDO close_prt,1;
setDo déatt_piece,1;
```

```
Movel P_offs_piece,v1000,fine,tool0\WObj:=Wob_T_pos2;
endproc
ENDMODULE
```

3-Sous-path

```
PROC Path_120()
MoveJ Target_3870,v1000,z100,Tooldata_limeuse\WObj:=Wob_porteur;
MoveL Target_3520,v100,z100,Tooldata_limeuse\WObj:=Wob_porteur;
MoveL Target_3530,v100,z100,Tooldata_limeuse\WObj:=Wob_porteur;
MoveL Target_3540,v100,z100,Tooldata_limeuse\WObj:=Wob_porteur;
MoveL Target_3550,v100,z100,Tooldata_limeuse\WObj:=Wob_porteur;
MoveL Target_3560,v100,z100,Tooldata_limeuse\WObj:=Wob_porteur;
MoveL Target_3570,v100,z100,Tooldata_limeuse\WObj:=Wob_porteur;
MoveL Target_3580,v100,z100,Tooldata_limeuse\WObj:=Wob_porteur;
MoveL Target_3590,v100,z100,Tooldata_limeuse\WObj:=Wob_porteur;
MoveL Target_3600,v100,z100,Tooldata_limeuse\WObj:=Wob_porteur;
MoveJ Target_3610,v100,z100,Tooldata_limeuse\WObj:=Wob_porteur;
MoveL Target_3620,v100,z100,Tooldata_limeuse\WObj:=Wob_porteur;
MoveL Target_3630,v100,z100,Tooldata_limeuse\WObj:=Wob_porteur;
MoveL Target_3640,v100,z100,Tooldata_limeuse\WObj:=Wob_porteur;
MoveL Target_3650,v100,z100,Tooldata_limeuse\WObj:=Wob_porteur;
MoveL Target_3660,v100,z100,Tooldata_limeuse\WObj:=Wob_porteur;
MoveL Target_3670,v100,z100,Tooldata_limeuse\WObj:=Wob_porteur;
MoveL Target_3680,v100,z100,Tooldata_limeuse\WObj:=Wob_porteur;
MoveL Target_3690,v100,z100,Tooldata_limeuse\WObj:=Wob_porteur;
MoveL Target_3700,v100,z100,Tooldata_limeuse\WObj:=Wob_porteur;
MoveL Target_3710,v100,z100,Tooldata_limeuse\WObj:=Wob_porteur;
MoveL Target_3730,v100,z100,Tooldata_limeuse\WObj:=Wob_porteur;
MoveL Target_3720,v100,z100,Tooldata_limeuse\WObj:=Wob_porteur;
MoveL Target_3740,v100,z100,Tooldata_limeuse\WObj:=Wob_porteur;
MoveL Target_3750,v100,z100,Tooldata_limeuse\WObj:=Wob_porteur;
MoveL Target_3760,v100,z100,Tooldata_limeuse\WObj:=Wob_porteur;
MoveL Target_3770,v100,z100,Tooldata_limeuse\WObj:=Wob_porteur;
MoveL Target_3780,v100,z100,Tooldata_limeuse\WObj:=Wob_porteur;
MoveL Target_3790,v100,z100,Tooldata_limeuse\WObj:=Wob_porteur;
MoveL Target_3800,v100,z100,Tooldata_limeuse\WObj:=Wob_porteur;
MoveL Target_3810,v100,z100,Tooldata_limeuse\WObj:=Wob_porteur;
MoveL Target_3820,v100,z100,Tooldata_limeuse\WObj:=Wob_porteur;
MoveL Target_3830,v100,z100,Tooldata_limeuse\WObj:=Wob_porteur;
MoveL Target_3840,v100,z100,Tooldata_limeuse\WObj:=Wob_porteur;
MoveL Target_3850,v100,z100,Tooldata_limeuse\WObj:=Wob_porteur;
MoveJ Target_3870,v1000,z100,Tooldata_limeuse\WObj:=Wob_porteur;
ENDPROC
```

-Configuration()

```
SYSSIG_OUT:
-Status "EmStop" -Signal "Areet_Urg"
-Status "AutoOn" -Signal "Auto"
-Status "Error" -Signal "def_sys" -Arg2 "T_ROB1"
-Status "CycleOn" -Signal "OR_9"
SYSSIG_IN:
-Signal "Stop" -Action "Stop"
-Signal "EV_10" -Action "StartMain" -Arg1 "CONT"
INDUSTRIAL_NETWORK:
-Name "DeviceNet_Anybus" -Label "Anybus" -Address "16"
DEVICENET_INTERNAL_ANYBUS_DEVICE:
-Name "Automate" -VendorName "HMS" -ProductName "Anybus " -OutputSize 32\
```

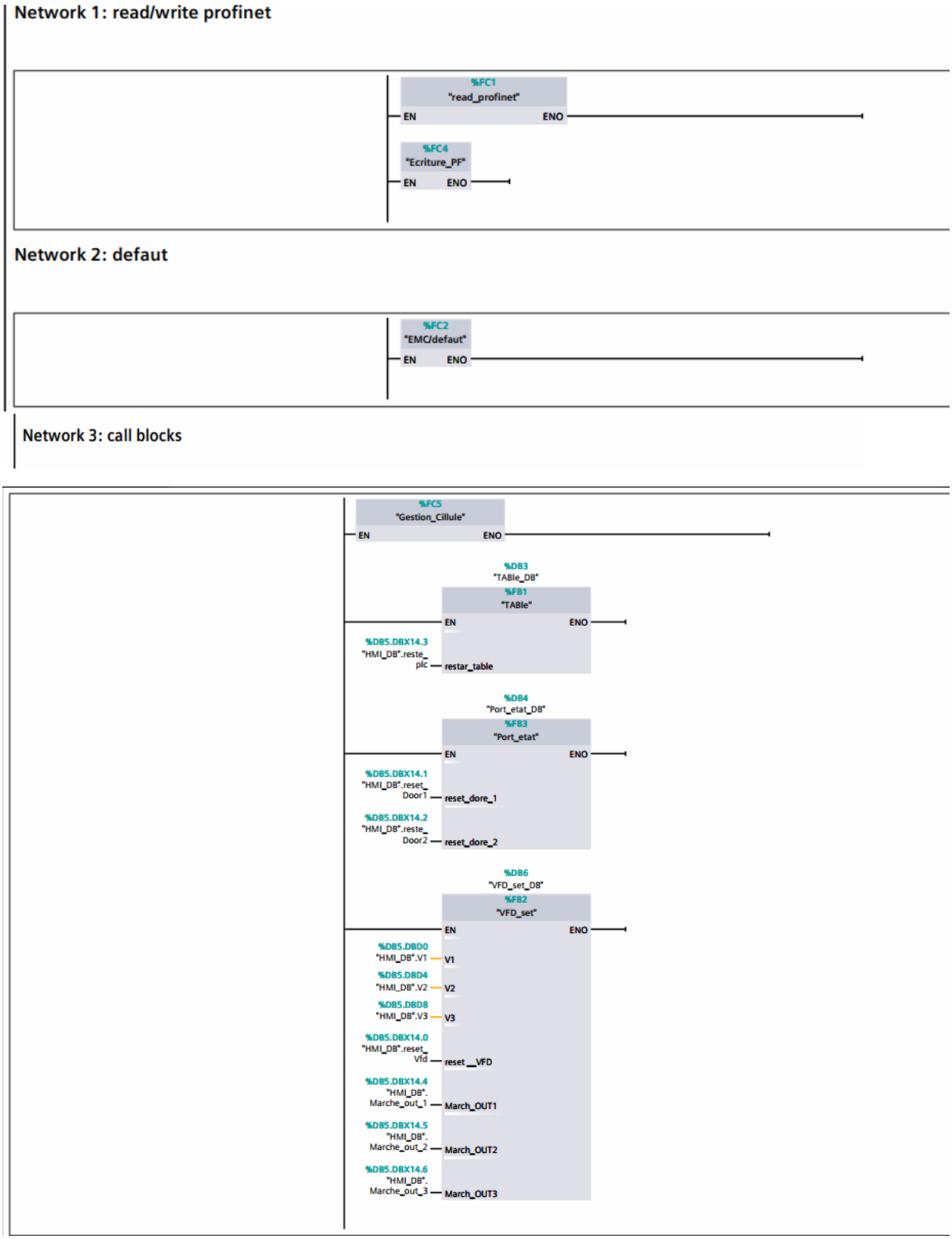
-InputSize 32
EIO_SIGNAL:
-Name "N_piece" -SignalType "GI" -Device "Automate"\
-Label "N° de la piece qui doit excuté" -DeviceMap "0-15"\
-Category "safety" -Access "All"
-Name "EV_1" -SignalType "DI" -Device "Automate" -Label "Auto_d'accé A"\
-DeviceMap "17" -Access "All"
-Name "EV_2" -SignalType "DI" -Device "Automate" -Label "Auto_d'accé B"\
-DeviceMap "18" -Access "All"
-Name "EV_3" -SignalType "DI" -Device "Automate"\
-Label "Auto_d'accé Outille 1" -DeviceMap "19" -Access "All"
-Name "EV_4" -SignalType "DI" -Device "Automate"\
-Label "Auto_d'accé Outille 2" -DeviceMap "20" -Access "All"
-Name "EV_5" -SignalType "DI" -Device "Automate"\
-Label "Auto_d'accé Outille3" -DeviceMap "21" -Access "All"
-Name "EV_6" -SignalType "DI" -Device "Automate"\
-Label "fin de prise recu" -DeviceMap "22" -Access "All"
-Name "EV_7" -SignalType "DI" -Device "Automate"\
-Label "fin de depose recu" -DeviceMap "23" -Access "All"
-Name "EV_8" -SignalType "DI" -Device "Automate"\
-Label "fin de travaille recu" -DeviceMap "24" -Access "All"
-Name "EV_9" -SignalType "DI" -Device "Automate" -Label "reset d'erreur"\
-DeviceMap "25" -Access "All"
-Name "def_sys" -SignalType "DO" -Device "Automate" -Label "def_sys"\
-DeviceMap "16" -Access "All"
-Name "N_cycle" -SignalType "GO" -Device "Automate" -Label "N° _cycle"\
-DeviceMap "0-15" -Category "safety" -Access "All"
-Name "OR_1" -SignalType "DO" -Device "Automate" -Label "Demande M_Out1"\
-DeviceMap "19" -Access "All"
-Name "OR_2" -SignalType "DO" -Device "Automate" -Label "Demande M_Out2"\
-DeviceMap "20" -Access "All"
-Name "OR_3" -SignalType "DO" -Device "Automate" -Label "Demande M_Out3"\
-DeviceMap "21" -Access "All"
-Name "OR_4" -SignalType "DO" -Device "Automate" -Label "Demande Pos A"\
-DeviceMap "22" -Access "All"
-Name "OR_5" -SignalType "DO" -Device "Automate" -Label "Demande Pos B"\
-DeviceMap "23" -Access "All"
-Name "OR_6" -SignalType "DO" -Device "Automate" -Label "Fin de depose"\
-DeviceMap "24" -Access "All"
-Name "OR_7" -SignalType "DO" -Device "Automate" -Label "Fin de travaille"\
-DeviceMap "25" -Access "All"
-Name "OR_8" -SignalType "DO" -Device "Automate" -Label "Fin d'Outille"\
-DeviceMap "26" -Access "All"
-Name "OR_9" -SignalType "DO" -Device "Automate"\
-Label "Robot en excution" -DeviceMap "27" -Access "All"
-Name "Stop" -SignalType "DI" -Device "Automate" -Label "Stop robot"\
-DeviceMap "16" -Access "All"
-Name "EV_10" -SignalType "DI" -Device "Automate" -Label "excut_Main"\
-DeviceMap "26" -Access "All"
-Name "Auto" -SignalType "DO" -Device "Automate" -Label "Demande M_Out1"\
-DeviceMap "17" -Access "All"
-Name "Areet_Urg" -SignalType "DO" -Device "Automate"\
-Label "Demande M_Out1" -DeviceMap "18" -Access "All"
-Name "tabl_pos1" -SignalType "DO"
-Name "tabl_pos2" -SignalType "DO"
-Name "att_piece" -SignalType "DO"
-Name "déatt_piece" -SignalType "DO"

-Name "close_prt" -SignalType "DO"
-Name "open_prt" -SignalType "DO"

exemple d’appel des procéder on programme principale:

```
MODULE Module1
proc main()
attache_piece;
papion;
Path_40;
Path_50;
Path_60;
Path_70;
Path_80;
Path_90;
Path_110;
Path_120;
déattache_piece;
MoveJ M_home,v1000,fine,tool0\WObj:=wobj0;
ENDPROC
ENDMODULE
```

TIA-PORTAL


Network 1: read/write profinet
%FC1
"read_profinet"
EN
ENO
%FC4
"Ecriture_PF"
EN
ENO
Network 2: defaut
%FC2
"EMC/defaut"
EN
ENO
Network 3: call blocks
%FC5
"Gestion_Cillule"
EN
ENO
%DB3
"TABle_DB"
%FB1
"TABle"
EN
ENO
%DB5.DBX14.3
"HMI_DB".reste_plc
restar_table
%DB4
"Port_etat_DB"
%FB3
"Port_etat"
EN
ENO
%DB5.DBX14.1
"HMI_DB".reset_Door1
reset_dore_1
%DB5.DBX14.2
"HMI_DB".reste_Door2
reset_dore_2
%DB6
"VFD_set_DB"
%FB2
"VFD_set"
EN
ENO
%DB5.DBD0
"HMI_DB".V1
V1
%DB5.DBD4
"HMI_DB".V2
V2
%DB5.DBD8
"HMI_DB".V3
V3
%DB5.DBX14.0
"HMI_DB".reset_Vfd
reset__VFD
%DB5.DBX14.4
"HMI_DB".Marche_out_1
March_OUT1
%DB5.DBX14.5
"HMI_DB".Marche_out_2
March_OUT2
%DB5.DBX14.6
"HMI_DB".Marche_out_3
March_OUT3

ملخص

يهدف العمل المقدم في هذه المذكرة إلى تحسين عملية التشطيب أو إزالة الزوائد الغير المرغوبة للأجزاء الميكانيكية باستخدام أحدث جيل من الروبوتات ABB-IRB 6700، وكانت الخطوة الأولى هي حساب النموذج الهندسي المباشر والعكسي للروبوت. استخدمنا لتصميم محيط وبيئة الخلية الروبوتية، والذي يوفر مساحة افتراضية ضخمة لاقتراح حلول الية. ويتم التحكم بألية الخلية عن طريق جهاز ألي من نوع Siemens S7-1200، وتمت برمجت ألبة التحكم والإشراف بواسطة برنامج .TIA-PORTAL فيما يتعلق بتحسين عملية التشطيب، قمنا باقتراح حلا يعتمد على معالجة الصور للأجزاء الميكانيكية وذلك بتجزئة الصورة ومقارنتها مع الصورة الأصلية ومن ثم تصفيتها وتنظيمها. تم إنشاء اتصال Ethernetبين Matlab وRobotStudio من أجل محاكات تتبع مسار الزوائد المراد تشطيبها، والنتائج تظهر نجاعة الحل المقترح.

الكلمات المفتاحية : ABB -IRB 6700, RobotStudio ,تحليل الصور ,التشطيب ,Siemens S7-1200

Résumé:

Le travail présenté dans ce mémoire a pour objectif l'amélioration de l'opération d'ébavurage de pièces mécaniques en utilisant la dernière génération du robot ABB-IRB 6700. La première étape était de calculer le modèle géométrique direct et inverse du robot. Afin de modéliser l'environnement de la cellule robotisé, nous avons utilisé le logiciel RobotStudio, qui offre un énorme espace virtuel pour proposer des solutions robotique. La partie automatisme de la cellule est gérée par un automate de type Siemens S7-1200, nous avons conçu les programmes de contrôle et de supervision sous le logiciel TIA-PORTAL. Quant à l'amélioration de la tâche d'ébavurage, nous avons proposé une solution basée sur un traitement d'image de la pièce mécanique, qui consiste à la segmentation de l'image puis le filtrage des points distants de la trajectoire des bavures. Une connexion Ethernet a été conçue entre Matlab et RobotStudio afin de simuler le suivi de trajectoire des bavures. Les résultats montre l'efficacité de la solution proposée.

Mots clé : Robot ABB-IRB 6700, RobotStudio, ébavurage, Traitement d'image, API Siemens S7-1200.

Abstract:

The work presented in this thesis aims to improve the deburring operation of mechanical pieces using the latest generation of the ABB-IRB 6700 robot. The first step is to calculate the forward and inverse kinematic model of the robot. In order to model the environment of the robotic cell, we have used the RobotStudio software, which offers an enormous virtual space to propose robotic solutions. The automation part of the cell is managed by a Siemens S7-1200, we have designed the control and supervision programs using the TIA-PORTAL software. For the improvement of the deburring task, we proposed a solution based on an image processing of the mechanical pieces, which consists in the segmentation of the image followed by the filtering of all points distant from the trajectory of burrs. An Ethernet connection was designed between Matlab and RobotStudio in order to simulate burr trajectory tracking. The proposed solution shows the effectiveness of the proposed solution.
Keywords: ABB-IRB 6700 robot, RobotStudio, deburring, Image processing, Siemens S7-1200 PLC