2018

Encadrants : U. NGUEVEU Sandra
BRIAND Cyril

Auteurs : GUILLOUX Lorris
SILVA Florent
MESKINE Walid
NGOMA MOUKO Thierry

[Projet Long 2018]

Commande et étude d'un réseau de transport d'une chaine de production robotisée présent sur le site de l'AIP

1 Remerciements

Dans un premier temps, nous souhaitions remercier Madame U. NGUEVEU Sandra ainsi que Monsieur BRIAND Cyril pour nous avoir permis de réaliser ce projet et pour toute l'aide, le soutien et le suivi qu'ils nous ont procurer.

Dans un second temps, nous souhaitions également remercier les équipes de l'AIP-PRIMECA pour l'accueil qu'ils nous ont réservé et l'intérêt qu'ils portaient à notre projet. Et plus particulièrement Madame FUGIER Francine pour le temps qu'elle nous a accordé sur le plan logistique et Monsieur CANZONERI Thierry pour son aide précieuse sur le plan informatique.

Finalement, nous tenions à remercier Monsieur BLANC-ROUCHOSSÉ Jean-Baptiste pour l'aide qu'il a pu nous apporter concernant la compréhension dans son ensemble du code informatique et de la relation entre les différents programmes.

Table des matières

1 Remerciements 1
2 Introduction 3
3 Présentation de l'organisme d'accueil 4
3.1 Le LAAS-CNRS 4
3.2 L'AIP-PRIMECA 5
4 Présentation de la cellule flexible 7
4.1 Le réseau de transport MONTRAC 7
4.2 Capteurs et Actionneurs 8
5 Objectifs et organisation du travail 10
5.1 Objectifs 10
5.2 Organisation 11
6 Montée en compétences dans les principaux outils du projet 12
6.1 Programmation ROS (sous Ubuntu) 12
6.2 Logiciel de simulation VREP 13
7 Accélération de la simulation 15
7.1 Accélération du point de vue hardware 15
7.2 Accélération du point de vue soft via V-Rep 16
8 Modifications des programmes existants pour répondre au cahier des charges 18
8.1 Portabilité du programme 19
8.2 Test de la simulation et compréhension des programmes 19
8.3 Modification de l'interface utilisateur 19
8.4 Modification de la création des produits 20
8.5 Changement de stratégie pour la création des produits 21
8.6 Mise en place d'un fichier de vérification de bon fonctionnement « fichier log » 21
9 Mise en place du sujet de TER 22
10 Améliorations possibles 25
10.1 Du point de vue de l'interface utilisateur 25
10.2 Fichier Log 25
10.3 Au niveau de la simulation 25
11 Conclusion 26
Bibliographie 27

2 Introduction

Dans le cadre de notre formation, en 3ème année d'école d'ingénieur à l'ENSEEIHT en option CDISC (Commande, Diagnostique et Informatique des Systèmes Critiques), nous avions l'opportunité de réaliser un Projet Long d'une durée de 6 semaines au sein de l'AIP-PRIMECA.

L'objectif de ce projet long était de rendre fonctionnel un réseau de transport industriel tout d'abord en simulation puis pouvoir appliquer les plans de productions préalablement testés au système réel. Ce projet est également l'opportunité de pouvoir se rapprocher du monde de l'entreprise et de développer de nouvelles compétences en Linux/ROS et VREP. Nous nous sommes particulièrement intéressés à la partie Simulation qui permettra aux étudiants de réaliser la totalité de leur travail sur une interface réaliste avant de pouvoir accéder à la maquette.

Ce travail fait suite aux projets long effectués en 2016 [4] et 2017 [3] par d'anciens étudiants ENSEEIHT ainsi qu'au TER puis au stage réalisés par des étudiants de l'Université Paul Sabatier.

Dans ce rapport, nous présenterons dans un premier temps l'AIP-PRIMECA ainsi que le LAAS, environnement dans lesquels nous étions implantés durant 6 semaines. Ensuite, nous présenterons les outils et systèmes sur lesquels nous avons travaillé pour faire notre simulation. De plus, nous détaillerons l'évolution du projet en expliquant les différentes étapes réalisées et nous dresserons un bilan à la fois technique et personnel de cette expérience. Pour finir, nous discuterons des évolutions futures envisageables et des améliorations que nous n'avons pas eu le temps de mettre en place.

3 Présentation de l'organisme d'accueil

3.1 Le LAAS-CNRS

Le Laboratoire d'analyse et d'architecture des systèmes (LAAS-CNRS [1]) est une unité propre du CNRS rattachée à l'Institut des sciences de l'ingénierie et des systèmes (INSIS) et à l'Institut des sciences de l'information et de leurs interactions (INS2I).

Figure 1: Etablissement du LAAS-CNRS

Les recherches menées au LAAS-CNRS visent à une compréhension fondamentale des systèmes complexes tout en considérant l'usage qui peut en découler. A l'inverse, de nombreuses problématiques sociétales ou industrielles, par exemple dans les domaines de l'aéronautique, de l'espace, de la santé, de l'énergie ou des réseaux de communication soulèvent des questions fondamentales qui nourrissent à leur tour l'inspiration des chercheurs.

Ainsi, à la fois défricheur de problématiques émergentes et promoteur de solutions intégrées, le laboratoire a identifié 4 axes stratégiques fondés sur les quatre champs disciplinaires qui constituent la marque de fabrique du laboratoire depuis sa création (informatique, robotique, automatique et micro et nano systèmes) :

- Adream : Architectures dynamiques reconfigurables pour systèmes embarqués autonomes mobiles

- Alive : Analyses des interactions avec le vivant et l'environnement

- Synergy : Systèmes pour une gestion intelligente de l'énergie

- Espace

Au sein de ces disciplines, 8 départements scientifiques définissent les orientations des prochaines années et coordonnent les activités des 22 équipes de recherche.

Ces recherches sont appliquées dans beaucoup de domaines d'applications comme l'aéronautique et l'espace, l'agriculture, la défense, énergie électrique, l'environnement et bien d'autres.

3.2 L'AIP-PRIMECA

Réparti sur tout le territoire et composé de 9 centres régionaux, le réseau AIP PRIMECA [2] est un réseau académique de chercheurs et d'enseignants chercheurs associé à des moyens technologiques de haut niveau pour la mise en commun de moyens techniques.

Figure 2: Répartition des pôles AIP-PRIMECA en France

Il est le résultat de la fusion entre :

- les A.I.P. (Ateliers Inter-établissements de Productique), créés en 1984 et utilisées comme support expérimental de formations approfondies dans le domaine de la Productique.

- PRIMECA (Pôles de Ressources Informatiques pour la MECAnique), créés en 1991 dans le but de promouvoir l'utilisation des outils informatiques dans la conception des produits mécaniques.

Les établissements toulousains (Université Paul Sabatier, INP Toulouse, INSA Toulouse, LAAS-CNRS) ont développé dès 1983 une politique de site afin de renforcer la formation pratique dans certains domaines nécessitant des moyens lourds et couteux, en phase avec la réalité industrielle. Cette politique de mutualisation des ressources et des compétences s'est traduit pas la création de trois ateliers interuniversitaires dont l'AIPPRIMECA, dans les domaines de la Mécanique et de la Productique. On peut retrouver une photo montrant l'établissement dans lequel nous avons évolué la plus grand partie du temps.

Figure 3: Etablissement de l'AIP-PRIMECA

4 Présentation de la cellule flexible

4.1 Le réseau de transport MONTRAC

L'AIP PRIMECA de Toulouse met à disposition de nombreux moyens dont la cellule flexible de production robotisée MONTRAC. Elle est constituée d'un réseau de transport monorail, appelé MONTRAC, sur lequel circulent des navettes. Ces dernières sont alimentées par les rails et sont donc en mouvement dès que la cellule est alimentée. Elles sont aussi autonomes et intelligentes : en effet, le seul moyen de les contrôler est de commander les actionneurs placés sur les rails via des automates.

Afin d'empêcher toute collision, chaque navette possède un capteur de proximité frontal qui permet de la stopper lorsqu'elle est trop proche d'une autre. La figure ci-dessous présente un schéma de la cellule :

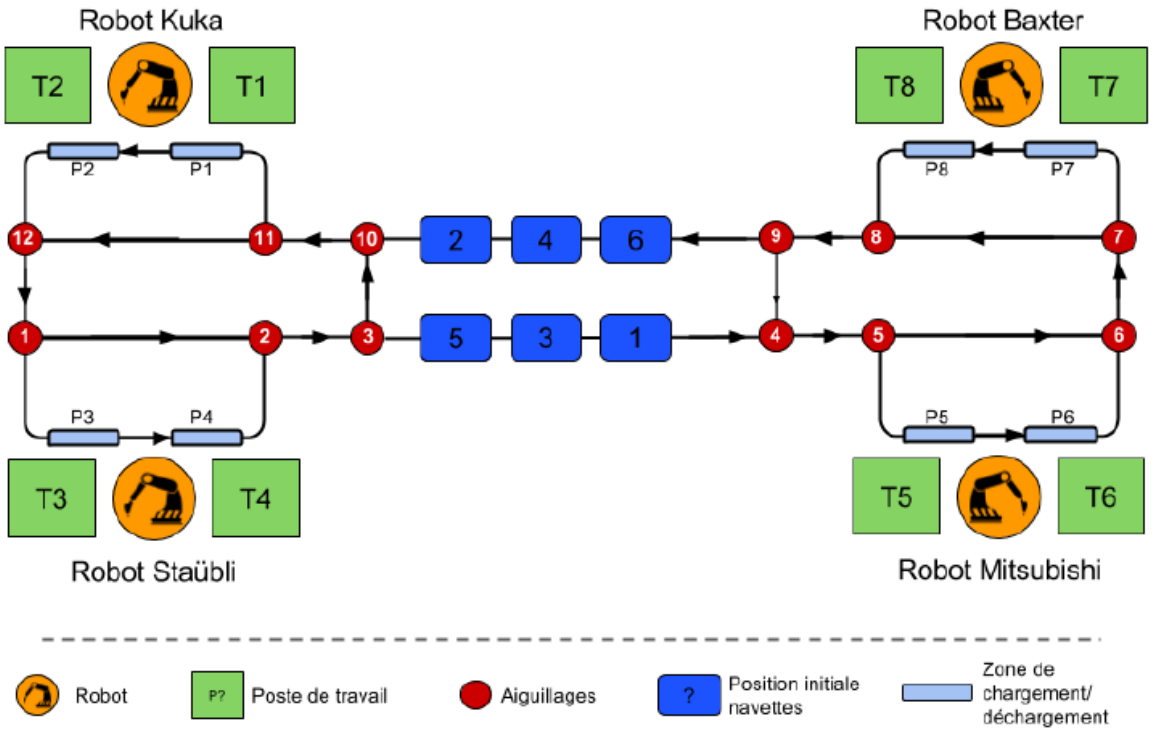


Figure 4: Schéma de la cellule flexible

Comme nous pouvons le voir, la cellule est composée de 4 zones de travail (chacune séparée en 2 zones de chargement/déchargement représentée en bleu clair et 2 postes de travail en vert) sur lesquelles 4 robots réalisent des opérations sur les produits transportés par les navettes. Ces zones sont desservies par des monorails en aluminium sur lesquels les navettes circulent, tout en respectant le sens de circulation indiqué par les flèches noires. Ce sens unique de circulation est imposé par l'alimentation latérale des rails et permet d'éviter

les risques de collisions frontales. Le routage est réalisé grâce aux 12 aiguillages présents (A1 à A12 représentés par des cercles rouges sur la figure 4). Le tout est divisé en 5 zones (ici invisibles), contrôlées chacune par un automate programmable.

4.2 Capteurs et Actionneurs

Les capteurs permettent de connaître la position des navettes, des aiguillages et des ergots. Les ergots servent à bloquer les navettes au niveau des zones de chargement/déchargement, ce qui permet aux robots de manipuler les produits positionnés sur les navettes, sans risque de les déplacer accidentellement. Les actionneurs permettent de commander les points d'arrêts des navettes, les aiguillages et la position des ergots. La liste des capteurs et actionneurs est donnée ci-dessous, ainsi qu'un schéma de leur répartition dans la cellule :

Tableau 1: Liste des actionneurs

RxG	Positionner l'aiguillage x à gauche
RxD	Positionner l'aiguillage x à droite
Dx	Déverrouiller l'aiguillage x
Vx	Verrouiller l'aiguillage x
STx	STx = 0 arrête la navette au niveau de l'actionneur (= 1 libère la navette)
PIx	Blocage/Déblocage des navettes sur la zone de chargement

Tableau 2: Liste des capteurs

CPx	Capteur de position, vaut 1 quand une navette est sur le capteur
PSx	Capteur de stop situé en face d'un actionneur STx pouvant arrêter la navette
CPIx	Vaut 1 quand l'ergot PI est sorti
DxD	Vaut 1 quand l'aiguillage est à droite
DxG	Vaut 1 quand l'aiguillage est à gauche

Nous pouvons retrouver ci-dessous un schéma de la cellule flexible mettant en évidence les différents capteurs et actionneurs cités précédemment :

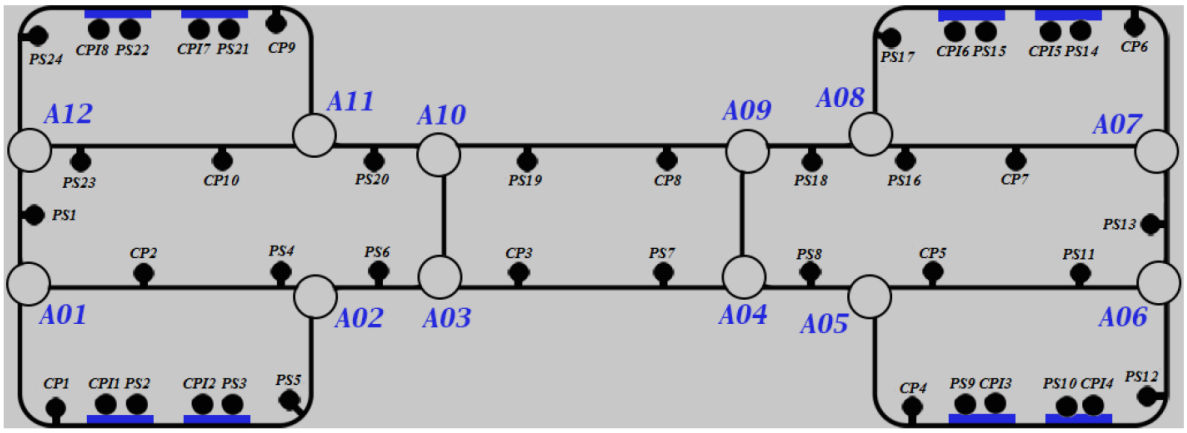


Figure 5: Schéma des capteurs et actionneurs

Les actionneurs STOP/START (STx), non visibles ici, permettent de bloquer les navettes et sont placés avant les aiguillages et au niveau des postes. Les actionneurs PIx font sortir l'ergot, permettant de stabiliser la navette au niveau des postes de travail. Il est pertinent de préciser que les ergots n'arrêtent pas physiquement la navette mais font office d'objet de détection magnétique pour la navette.

De plus, à chaque aiguillage sont associés deux capteurs DxD et DxG qui permettent de détecter sa position (respectivement droite ou gauche).

5 Objectifs et organisation du travail

5.1 Objectifs

A l'heure actuelle, l'utilité de cette maquette est de pouvoir développer des Travaux Pratiques destinés à des étudiants de cycle supérieur issus de différentes formations de Toulouse, en particulier pour l'ENSEEIHT qui souhaiterait remplacer une ancienne maquette (TER atelier flexible) qui ne fonctionne plus. Ces TP devraient permettre de former les étudiants à la commande de haut niveau de systèmes de production en respectant un ordonnancement préétabli et les gammes de production. Elle permettra aussi de se confronter et de se familiariser avec les notions de ressources partagées au sein d'un réseau de pétri complexe.

Ce projet s'inscrit dans la problématique de l'usine du futur, dans le sens où l'ensemble des différents capteurs présents sur la cellule permettent une communication constante entre les machines et les humains. En effet, il y a maintenant plusieurs années que l'on commence à parler d'usine du futur, ou « usine 4.0 », capable de faire communiquer les machines entre elles. Son principe est "qu'à chaque maillon des chaînes de production et d'approvisionnement, les outils et postes de travail communiquent en permanence grâce à Internet et aux réseaux virtuels. Machines, systèmes et produits échangent de l'information, entre eux ainsi qu'avec l'extérieur" [5].

Dans notre cas, il y a une communication constante entre l'utilisateur et la cellule, mais également une communication constante entre chaque navette, les rails et les robots. Il devient alors beaucoup plus aisé de localiser la source d'une panne et de rediriger la production sur d'autres machines afin d'éviter l'arrêt total de la production durant le temps de manutention de la machine en panne [6]. Nous produisons ainsi un suivi en temps réel de l'avancement des différentes tâches de la production et de l'acheminement, ce qui permet d'avoir une gestion et une planification de la production plus performante.

Plus particulièrement, nous avions au début de ce projet 3 grands objectifs à réaliser :

- Le premier objectif était de pouvoir accélérer la simulation qui était très lente et qui empêcherai donc les étudiants de travailler dans de bonnes conditions (et freinerai également leur avancement sur des créneaux de travaux pratiques limités). Ce travail a été réalisé sur le logiciel VREP et sera expliquer plus en détail par la suite.

- Le second objectif était de réaliser un sujet de TER à partir du programme existant afin de fournir un livrable complet et prêt à l'utilisation par les élèves de l'ENSEEIHT. Nous devions donc penser à un sujet complet permettant d'illustrer toutes les problématiques voulues liées au programme pédagogique en place dans la filière GEA. D'un point de vue

pédagogique, l'objectif est que les étudiants se forment aux problématiques de gestion de ressources partagées rencontrés sur les chaines de production. La modélisation de ce problème se fera à l'aide de réseaux de Petri. Afin que celui-ci ne soit pas trop compliqué à mettre en place, l'idée est que les étudiants aient accès à des fonctions de haut niveau pour coder la couche haute du programme.

- Le dernier objectif se rattache grandement avec le précédent, le but étant de fournir un code fonctionnel et adapté au problématiques voulues pour le TER. Cette partie fut de loin la plus complexe et la plus chronophage étant donnée la taille et la difficulté importante du projet.

5.2 Organisation

Nous avons basé l'organisation de notre projet sur la méthode Agile. Cette méthode consiste à livrer régulièrement des fonctionnalités à forte valeur ajoutée ainsi qu'une application fonctionnelle finale.

Chaque semaine, nous établissions une répartition des tâches et les objectifs à atteindre à la fin de la semaine. En effet, à chaque semaine était organisée une réunion durant laquelle nous présentions l'avancement du projet et les futures évolutions à faire. Nos encadrants nous aidaient en commentant notre travail afin de mettre en avant les corrections et/ou améliorations fonctionnelles à réaliser pour la semaine suivante. De plus, ces réunions nous permettaient aussi d'éclaircir certains points, en particulier la trame du futur TER à préparer pour les prochains étudiants. Au vu de l'avancée du projet, nous avons multiplié ce nombre de réunion à 2 par semaine pour procurer un meilleur suivi et pouvoir changer de directives plus facilement.

Par ailleurs, chaque réunion menait à la rédaction d'un compte-rendu afin de pouvoir conserver une trace écrite de ce qui avait été dit et de pouvoir envoyer, un bilan et un ordre du jour à nos encadrant.

Du fait que ce projet soit déjà lancé depuis plusieurs années, une plateforme de simulation et un environnement commande sur maquette réelle avaient déjà été réalisés. Ainsi, nous avons commencé à lire tous les livrables mis à notre disposition. Mais pour comprendre le travail effectué, nous avons dû passer par une phase de prise en main de ROS et de V-REP. En effet, ces deux outils nous étaient alors inconnus. Nous avons donc suivi de nombreux tutoriels afin de comprendre leurs fonctionnements et donc les codes de nos camarades pour commencer à remplir nos objectifs.

Pour ce qui est de la gestion des tâches au sein du groupe, nous nous sommes la plupart du temps répartis en deux groupes de 2 travaillant en parallèle. Ces groupes n'étaient pas fixes

et dépendaient des compétences et envies de chacun. L'ensemble de ces tâches seront détaillées par la suite dans ce rapport.

6 Montée en compétences dans les principaux outils du projet

6.1 Programmation ROS (sous Ubuntu)

Comme dit précédemment, pour la réalisation du projet, nous avons hérité des choix d'outils utilisés par les groupes précédents.

Le premier est le middleware ROS (Robot Operating System), une plateforme de développement logiciel qui fournit des bibliothèques et des outils pour aider les développeurs de logiciels à créer des applications robotiques. Pouvant fonctionner sur un ou plusieurs ordinateurs, il procure de nombreuses fonctionnalités telles que l'abstraction du matériel, le contrôle des périphériques de bas niveau, la transmission de messages entre les processus et la gestion des packages installés.

De manière simple, ROS permet de créer des sous-programmes appelés nœuds ou *nodes* en anglais, qui peuvent communiquer entre eux à l'aide de messages synchrones ou asynchrones. ROS offre une architecture souple de communication interprocessus et inter-machine. Dans le cas de messages asynchrones, les processus ROS *nodes*, peuvent communiquer avec d'autres via des topics.

La connexion entre les *nodes* est gérée par un master et suit le processus suivant :

- Un premier *node* avertit le master qu'il a une donnée à partager
- Un deuxième *node* avertit le master qu'il souhaite avoir accès à une donnée
- Une connexion entre les deux *nodes* est créée
- Le premier *node* peut envoyer des données au second

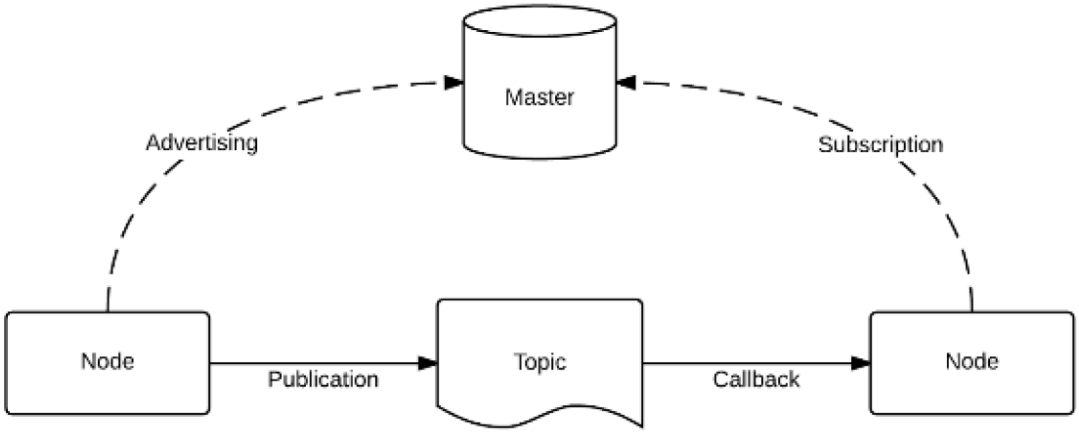


Figure 6: Diagramme de fonctionnement de ROS

Un *node* qui publie des données est appelé un *publisher*, et, un *node* qui souscrit à des données est appelé un *subscriber*. Un *node* peut être à la fois *publisher* et *subscriber*. Les messages envoyés sur les topics sont, pour la plupart, standardisés, ce qui rend le système extrêmement flexible.

Pour les messages synchrones, que l'on appelle services, les nœuds ont seulement accès aux services offerts par chacun d'entre eux, par l'intermédiaire de clients. Sans entrer dans le détail, les entrées et sorties de chaque service sont définies au niveau des nœuds où ils sont instanciés, et elles sont connues des nœuds clients correspondants. Il est important de remarquer que ROS permet une communication inter-machine, des *nodes* s'exécutant sur des machines distinctes, mais ayant connaissance du même master peuvent communiquer de manière transparente pour l'utilisateur. Ceci représente une des grandes forces de ROS.

De plus, ROS est sous licence open source, ce qui permet d'avoir accès à une grande communauté et de disposer de nombreux tutoriels. ROS est aujourd'hui, officiellement supporté par plus de 75 robots. La grande souplesse de ROS lui permet d'être déployé sur des robots très différents (robot mobile, bras industriel, multicoptère) et qui évoluent dans des milieux variés (terrestre, aérien, marin et sous-marin).

6.2 Logiciel de simulation VREP

Le second outil que nous avons utilisé est V-REP, un simulateur dédié à la simulation de scènes composées de robots et développé par la société Coppelia Robotics. L'intérêt de ce simulateur est qu'il est compatible avec ROS. En d'autres termes, il est vu par nos nœuds ROS comme un nœud spécifique, avec lequel nous pouvons communiquer.

Il permet la simulation de la cellule MONTRAC, se situant à l'AIP. La scène a été initialement créée par le groupe précédent, et nous y avons ajouté de nombreux éléments au cours du développement de notre projet long. En plus des modèles 3D des navettes et des rails de la cellule, on retrouve aussi les modèles de robots KUKA que nous avons utilisé pour simuler notre atelier de fabrication. Nous avons également apporté quelques modifications au niveau de l'emplacement des capteurs (voir paragraphe 2.2). La simulation V-REP finale permet donc de faire évoluer les navettes sur les rails, tout en communiquant l'état des capteurs au reste de notre simulateur, grâce à son nœud ROS.

7 Accélération de la simulation

Le premier problème que nous avions à résoudre était donc la vitesse de simulation. En effet, la simulation réalisée par nos camarades était malgré eux très lente et les navettes mettaient du temps pour se déplacer entre les différents postes. Cela pourrait sans doute empêcher les étudiants de travailler dans de bonnes conditions, mais aussi perturber leur avancée et progression vu le temps restreint dont ils disposent. Il fallait donc trouver un moyen d'augmenter la vitesse de simulation pour rendre les tests plus fluides.

7.1 Accélération du point de vue hardware

Tout d'abord, nous avons lancé la simulation sur plusieurs machines différentes pour évaluer la vitesse sur chaque poste. Nous avons remarqué que cette dernière varie légèrement : Plus un ordinateur est performant, plus la vitesse de simulation est grande. Le problème est donc lié au temps de calcul et cette piste ne menait à rien puisque les ressources dont on dispose sont limitées. Il faut savoir qu'une machine assez puissante (type AIP) est nécessaire pour faire tourner la simulation avec une vitesse décente.

Nous avons également tenté avec succès de mettre en place l'environnement de simulation sur d'autres machines sous forme de machine virtuelles Linux. Or il faut savoir que les machines virtuelle tournent essentiellement sur le CPU de l'ordinateur et non la carte graphique, les performances observées sont donc assez médiocres sur machines virtuelle. Des machines native Linux sont donc à privilégier.

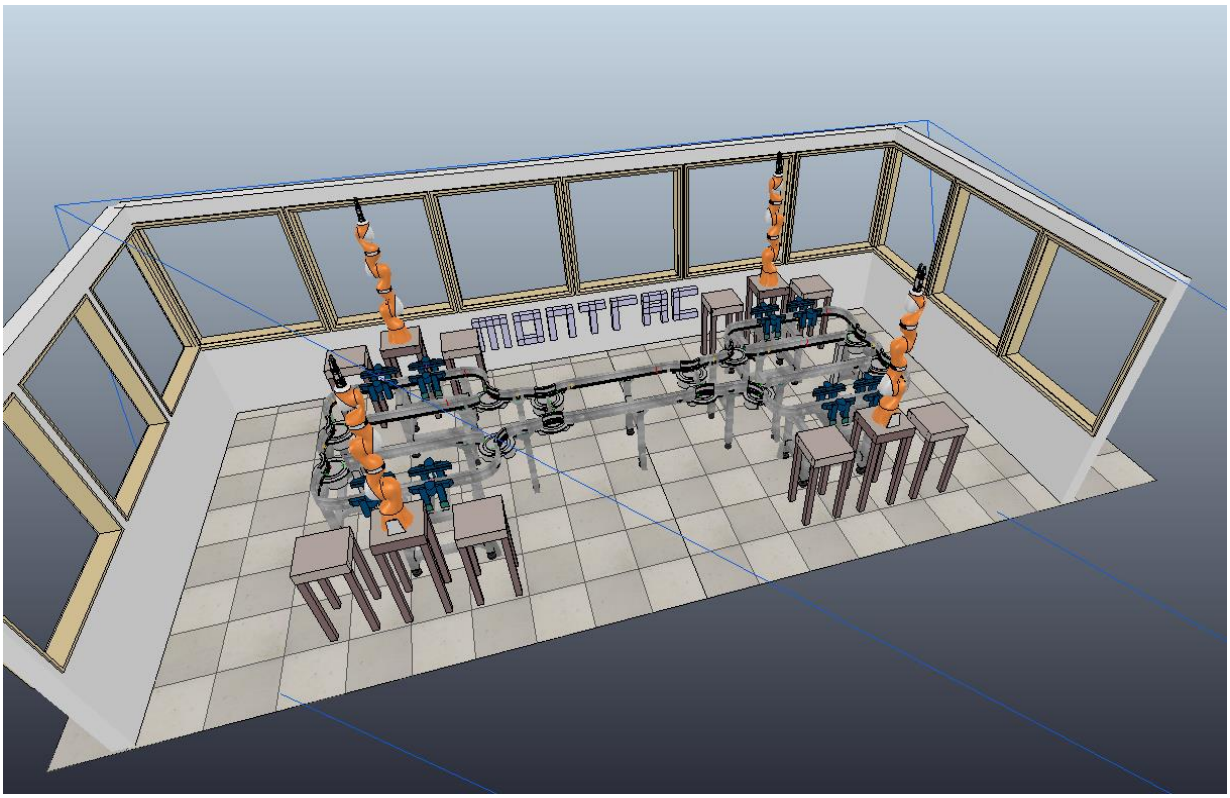


Figure 7 - Simulation 3D sous V-rep

7.2 Accélération du point de vue soft via V-Rep

Ensuite, nous nous sommes intéressés au logiciel V-REP et aux paramètres de simulation. Le tout premier paramètre à ajuster était donc le pas de calcul qui était fixé à 35ms. On augmentait alors peu à peu le pas de calcul et on remarquait que la vitesse de simulation augmentait aussi.

Cependant, à partir de 80ms nous avons remarqué un mauvais fonctionnement de la simulation et les scénarios n'arrivaient pas à bout suite aux accidents rencontrés. Par exemple, la navette transportant le produit ne s'arrête pas devant le poste:

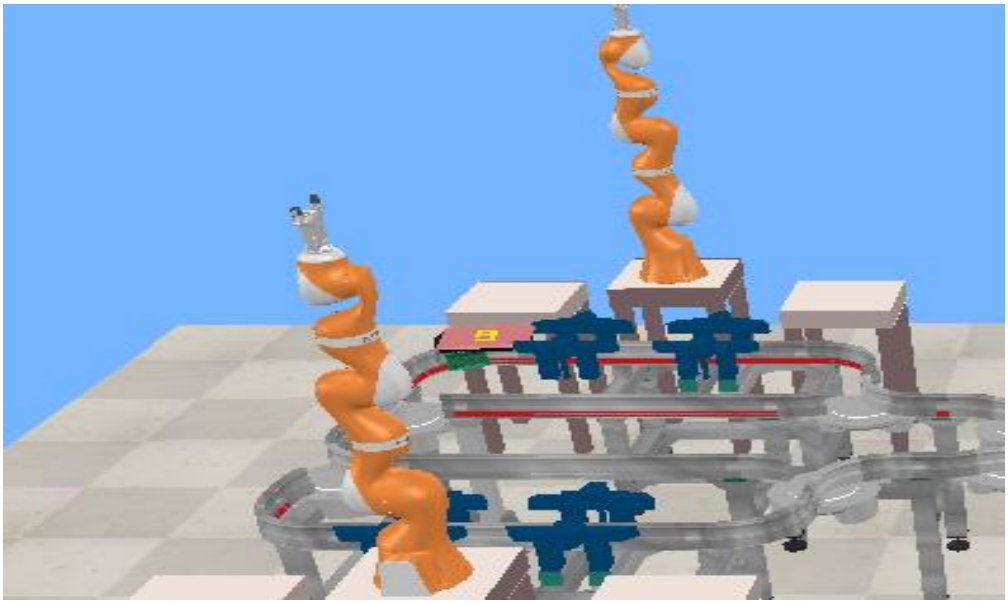


Figure 8 - Situation anormale suite à une augmentation trop importante de la vitesse de simulation

Nous avons donc décidé de se limiter à un pas de calcul de 65ms pour la suite, aucun dysfonctionnement n'a été reporté et les navettes suivaient bien le plan.

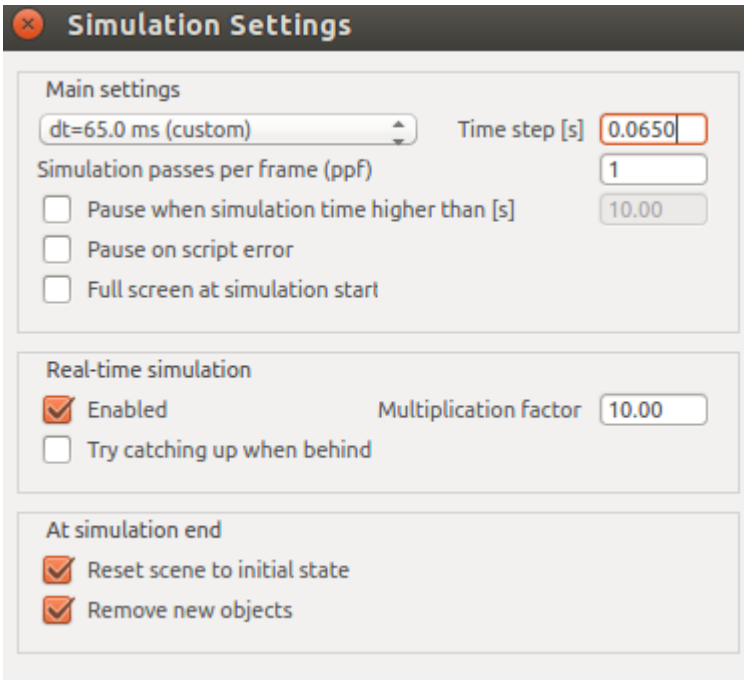


Figure 9 - Paramètres de simulation optimaux

Le logiciel V-rep offrait aussi différents modèles de calcul (Bullet, Newton, ODE et Vortex) avec différentes vitesses (Very accurate → Very Fast). Il fallait donc exploiter cette piste pour choisir la méthode qui offrait les meilleurs résultats.

La plupart des méthodes n'étaient pas adaptées à notre simulation et présentaient même un crash des fois, comme est le cas pour la méthode Vortex en Very Fast:

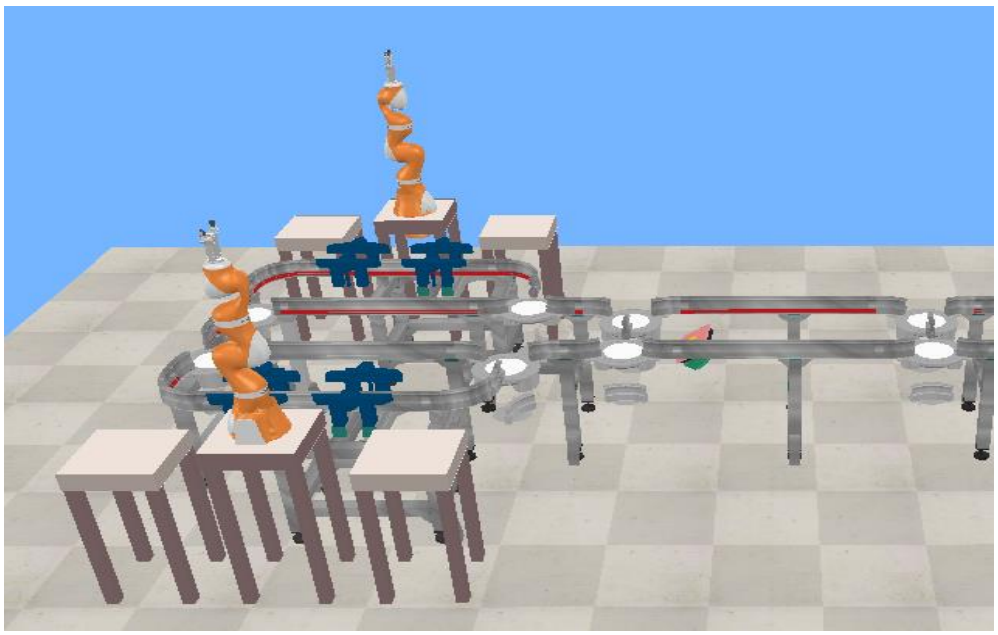
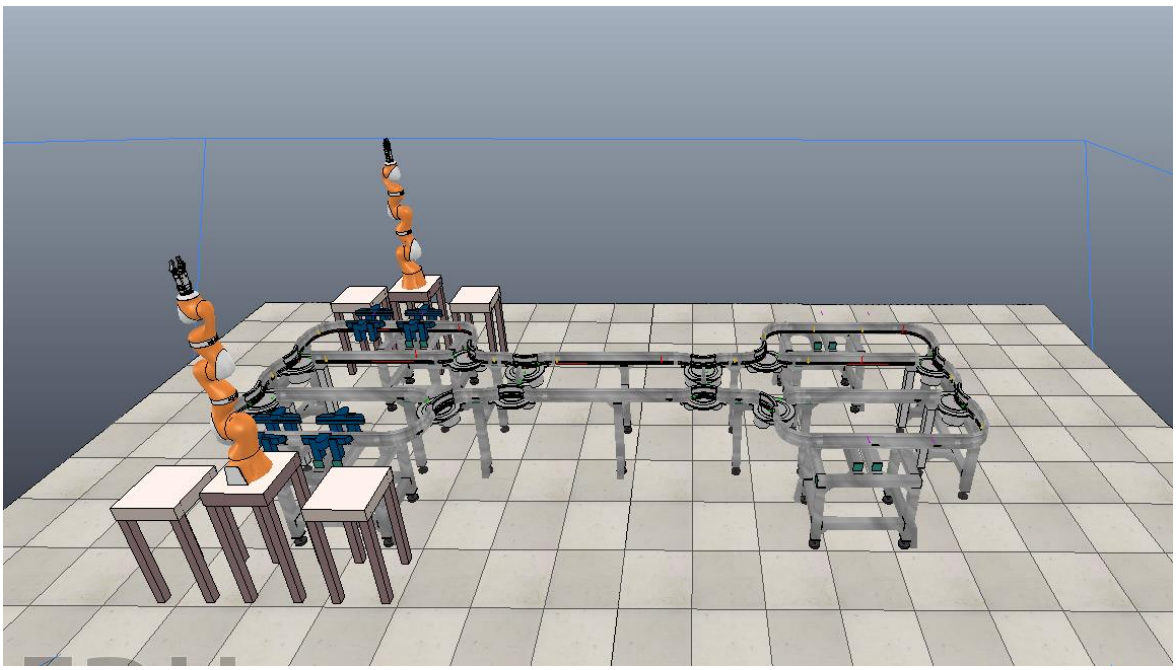


Figure 10 - Dysfonctionnement des modèles V-rep en méthode Vortex

Le seul mode qui fonctionnait bien était le mode Bullet que nous avons décidé de garder pour la suite en Fast.

Même après tous ces changements, la simulation nous paraissait encore lente. Il fallait donc exploiter une autre piste qui est celle les robots. En effet ces derniers requièrent beaucoup de calculs et de mémoire par conséquent, ce qui faisait ralentir la simulation. On voyait donc une nette différence en retirant les deux robots qui n'étaient pas utilisés de notre modèle puisque les navettes ne parcouraient que les postes 1, 2, 3 et 4. Nous avons aussi enlevé tous les détails graphiques qui n'affectaient pas la simulation:



Suite à ça, nous remarquons que la vitesse de la simulation est désormais acceptable.

8 Modifications des programmes existants pour répondre au cahier des charges

Un des principaux axes du projet long était la modification du programme en vue de mettre en place le TER pour les futurs étudiants.

Pour cela nous avons dû comprendre le programme, dans sa globalité. Premièrement nous avons dû mettre en place l'environnement de simulation. Malheureusement le programme permettant d'initialiser l'environnement sur une nouvelle machine fourni par les anciens élèves ne fonctionnait pas. Nous avons cependant pu récupérer les fichiers déjà compilés des élèves de l'année dernière sur une des machines et à partir de ceux-ci nous avons créé un nouvel environnement qui pourra par la suite être directement utilisé par de futurs étudiants. Après avoir récupéré les fichiers, nous nous sommes rendus compte que la recompilation en fonctionnait toujours pas, ceci était dû aux fichiers de compilation de packages ROS (CmakeFiles et CmakeLists) qui ne possédaient pas les bons chemins d'accès pour la compilation, après avoir changé ces chemins la compilation fonctionnait de nouveau.

8.1 Portabilité du programme

La portabilité du programme est une chose essentielle. En effet, le projet sera amené à être déplacé sur plusieurs machines lorsque les étudiants l'utiliserons pour le TER. Nous avons donc dans un premier temps concentré nos efforts sur cet aspect-là. Finalement, nous avons réussi à rendre le projet portable et re-compilable et nous avons mis en place un nouveau tutoriel d'initialisation de l'environnement. Ce tutoriel pourra être utilisé par les encadrants du TER ou les élèves mêmes pour mettre en place leur environnement. Le fichier tutoriel est le fichier nommé « Mise en place de l'environnement de simulation ». Ce tutoriel à bien entendu été testé et approuvé par nos soins sur une machine virtuelle Linux.

8.2 Test de la simulation et compréhension des programmes

Une fois l'environnement et sa mise en route établis nous avons pu commencer à faire tourner la simulation et à essayer d'appréhender le fonctionnement des programmes. Le projet étant assez conséquent, nous avons mis un certain temps pour comprendre les principaux programmes. Nous avons finalement fait appel à un élève de l'année dernière pour nous débloquer et pour répondre à nos questions, nous avons grâce à lui acquis une meilleure connaissance des programmes. Connaissance assez détaillée pour pouvoir entamer les modifications.

Certains choix ont étés pris quand a la modification désirée du programme. Certains de ces choix ont pu être mis en œuvre mais d'autre n'ont pas pu être mis e œuvre, du moins cette année.

8.3 Modification de l'interface utilisateur

Premièrement nous nous sommes penchés vers l'interface utilisateur. Nous avons modifié cette interface pour la rendre plus compréhensible auprès des étudiants. Nous avons tout d'abord rajouté une interface TER symbolisée par un bouton « TER » cela permet de lancer directement le réseau de pétri que les étudiants créerons eux même à partir d'un squelette fourni avec l'archive du projet. Nous avons également ajouté un cadre permettant d'afficher les différents produits à créer. L'interface se présente maintenant comme ceci. La partie accélération est prise en compte ici.

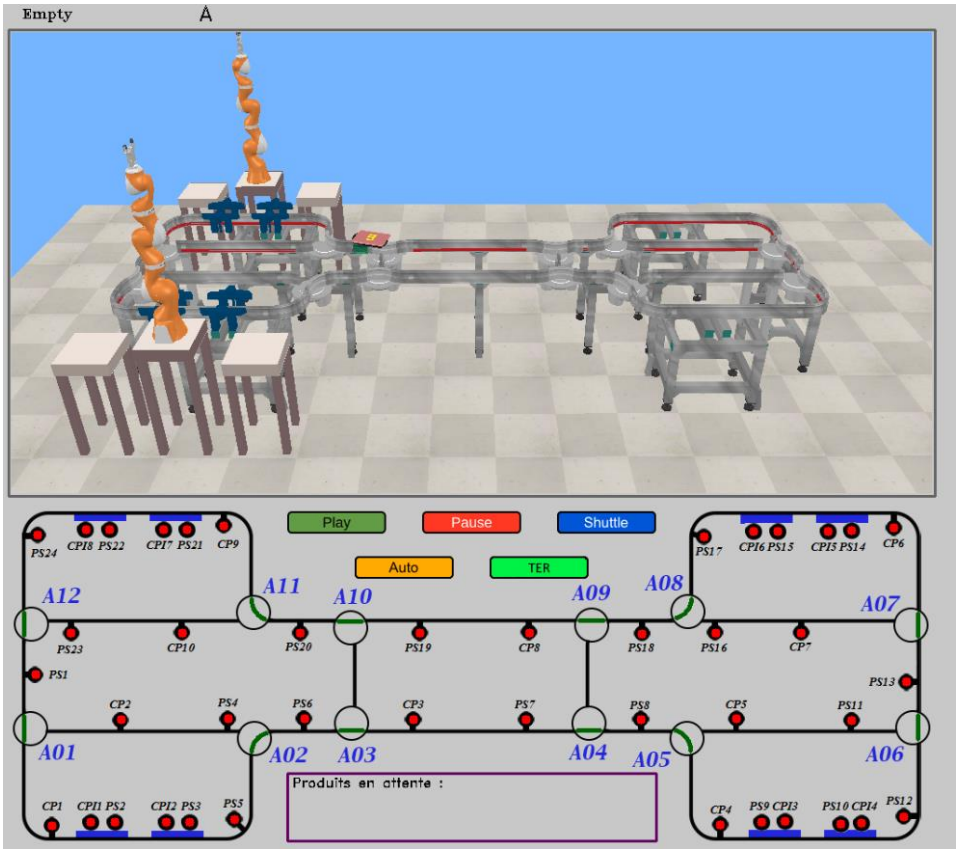


Figure 11 - Simulation finale

8.4 Modification de la création des produits

Un des sous objectifs de modification des programmes était d'essayer de séparer la création des produits avec la création des navettes. En effet, dans le programme réalisé par les étudiants des années précédentes, la création d'un nouveau produit est forcément liée à une navette. Nous voulions changer ce mode d'initialisation pour pouvoir proposer un problème de ressource dans le sujet de TER (pouvoir déclarer moins de navettes que de produits pour créer des conflits comme vu dans la partie sujet de TER). Nous avons donc dans un premier temps réussi à créer des navettes vides en réutilisant le principe d'initialisation des années précédentes. Dans un second temps, nous voulions donc créer des produits séparément et donc directement sur les plateformes de traitement. Sur cet axe de développement, nous sommes tombés sur un problème dû à la manière avec laquelle ils avaient codé leur programme les années précédentes. En effet, pour faciliter le travail d'équipe, ils avaient utilisé une méthode de programmation modulaire permettant de travailler à plusieurs sur le même code en parallèle. Cependant, pour réaliser le changement en question, il nous fallait toucher au cœur même du programme et donc aux communications entre les différents blocs. Ce qui n'était pas possible et impliquait donc de recommencer le projet avec un code plus orienté sur la consultation d'une "librairie" principale pour que tous les sous-programmes utilisent les mêmes données en temps réel. Etant donné le temps restreint qui nous était alloué, nous avons donc abandonné cet axe de développement pour explorer une nouvelle solution.

8.5 Changement de stratégie pour la création des produits

Vu que nous ne pouvions pas séparer la création d'un produit et d'une navette nous avons décidé d'aborder le problème sous un autre angle. Faute de temps nous avons choisi la solution qui nous semblait la plus facile à mettre en œuvre. Autant sur le plan programmation que pour les futurs étudiants qui utiliserons la simulation. Nous avons décidé d'implanter une nouvelle fonction de haut niveau qui permettra aux étudiants de supprimer une navette, que ce soit dans le Scheduler mais également dans la simulation. Cette fonction appelée « Destroy_Shuttle » et qui prend comme argument le Handle de la navette. La navette était déjà détruite en sortie de production est nous nous sommes inspiré de cette fonction pour créer la nôtre et nous l'avons rendue plus portable grâce à la création d'un topic dédié à la destruction de la navette directement depuis le réseau de pétri. Ce topic est appelé comme sa fonction éponyme « /commande_navette/Destroy_Shuttle ». Après avoir testé cette fonction nous avons choisi de l'utiliser comme partie intégrante du sujet, à savoir que les produit seront créés sur des navettes qui devront obligatoirement être détruite à partir du moment où l'objet se situe dans la première étape de son processus de fabrication.

Typiquement avec un objet de type A qui devra poursuivre le processus de fabrication « 1-3-2-4 », un navette contenant un produit A non usiné sera amené au poste 1 puis sera détruire après que le produit ait été récupéré. Pour le reste de plan de fabrication du produit d'autre navette (crées vides) seront amenés pour récupérer les produits ainsi.

8.6 Mise en place d'un fichier de vérification de bon fonctionnement « fichier log »

Un des points importants du projet que nous avons abordé en début de ce Projet Long était la mise en place d'un fichier « log file ». Ce fichier log a pour but de permettre à l'enseignant de contrôler le travail des étudiants, et également de les évaluer puisque au final ce sera uniquement ce fichier qui sera utilisé pour vérifier que leur programme fonctionne bien. Ce fichier sera traité ensuite automatiquement par un programme tiers qui vérifiera que tout a été réalisé correctement et qu'il n'y a pas d'erreur d'inversion de gamme de fabrication, d'erreur de priorités ou d'incohérences au niveau du réseau de pétri. Pour l'évaluation le professeur pourra récupérer les différents fichiers log, faire tourner l'algorithme dessus et vérifier que tout fonctionne.

Cependant le cahier des charges détaillé de ce programme de test n'étant pas encore détaillé, seule la mise en place du fichier log a été faite et nous donnerons par la suite quelques pistes pour pouvoir mettre en place ce programme.

Le fichier log entrera les différents temps d'arrivée des produits et des navettes ainsi que l'état d'avancement des produits. Vu que le cahier des charges pour le programme de test n'a pas été défini, le processus de fabrication du fichier sera donc amené à changer par la suite.

9 Mise en place du sujet de TER

Comme présenté précédemment, le sujet de TER fait partie des objectifs majeurs de notre projet. Sa rédaction s'est faite en se basant sur le programme pédagogique de la filière GEA. En effet, il a fallu penser à un sujet complet afin que les étudiants puissent mettre en application toutes leurs connaissances des réseaux de pétri, du partage de ressources et de la programmation en langage C++.

Le sujet de TER est structuré de façon à simplifier le plus possible la compréhension du système par les étudiants. Nous l'avons structuré de la façon suivante :

- **Introduction** : Une brève introduction afin de présenter les objectifs du TER, préciser les notions qui seront abordées et situer le contexte de l'atelier.

- **Présentation de la plateforme** : Comme son nom l'indique, il s'agit de la présentation de la cellule. Elle est identique dans le fond à celle qui a été faite précédemment dans ce rapport (cf Présentation de la cellule flexible).

- **Le cahier des charges** : Il s'agit de la partie la plus importante du sujet. En effet, c'est le cahier des charges qui permet de définir les attentes du TER. Dans ce dernier, on oriente les étudiants sur comment ils devront mener leur projet et quels outils ils utiliseront.

L'objectif principal est de réussir à assurer le traitement de plusieurs produits selon leur gamme de production avec un nombre limité de ressources : les robots et les navettes.

A l'aide d'un fichier de configuration "ProductConfiguration" qui sera donné aux étudiants, ils devront définir l'ordre et le séquencement du traitement de plusieurs produits. Ce fichier est assez important dans la mesure où la programmation de toutes les fonctions qui ont permis de réaliser cet atelier a été faite à partir de ce dernier. De plus, il contient les informations permettant d'utiliser correctement les fonctions de haut niveau disponibles. Ainsi comprendre ce fichier est primordial pour les étudiants.

Un autre fichier dédié à la programmation du réseau de pétri en C++ sera également fourni aux étudiants. Pour faciliter la programmation du réseau de pétri et faire correspondre la logique du code et celle des fonctions de haut niveau, l'initialisation a déjà été faite comme on le voit sur la figure ci-dessous :

```
int main(int argc, char **argv)
{

//initialisation du noeud ros et création d'un handle associé au noeud
        ros::init(argc, argv, "commande");
        ros::NodeHandle noeud;

//création et initialisation des objets

        Commande cmd(noeud,argv[0]);//Objet de la classe Commande permettant d'utiliser les fonctions de haut niveau

        Robots Robots(noeud);//Objet de la classe Robot permettant d'utiliser les fonctions des robots

        ros::Rate loop_rate(25); //fréquence de la boucle ros

// Initialisation de variables //

        cmd.Initialisation();

        int M[Nb_Place];//Matrice contenant l'ensemble des places du rdp
        int Nb_Place_T1,Nb_Place_T2,Nb_Place_T3,Nb_Place_T4;//Nombre de places par poste

        for(int i=0;i<Nb_Place;i++) M[i]=0;//Initialisation de toutes les places : Il n'y a aucun jeton

        //On réserve aux Robots, les places 50 et 250
        M[50]=1;//Robot 1 libre
        M[250]=1;//Robot 2 libre

        bool modif=1;

        while (ros::ok())
        {
////////////////////C'est à Vous/////////////////////////////////////////////////
```

Figure 12 - Initialisation dans le squelette du réseau de pétri fourni au étudiants

La réalisation physique de la plateforme n'étant pas encore opérationnelle, le travail se fera uniquement en simulation. Cela représente une information importante puisque les étudiants pourront tester plus souvent leur programme. Cependant, compte tenu de la vitesse de simulation qui est relativement faible, il faut bien regarder le programme avant de le simuler.

Enfin, nous avons mis en place un fichier Log qui détaille tous les événements qui se sont produits lors de la simulation comme détaillé dans la partie précédente. Ainsi grâce à ce dernier, les étudiants pourront comparer leurs résultats à la configuration qu'ils avaient rentrée pour valider le réseau de pétri.

Pour faciliter la compréhension et la programmation du système, nous avons pensé proposer aux étudiants une approche simpliste en ne tenant pas compte des robots dans un premier temps. Le traitement de produits au niveau des postes pourra être remplacé par une temporisation de quelques secondes. Dans un second temps, les robots seront pris en compte avec leurs fonctions afin de simuler le système global.

- **La simulation :** Nous avons également présenté brièvement la partie opérative de la simulation pour permettre aux étudiants de l'utiliser. En effet, nous avons expliqué tous les fonctions de chaque bouton disponible notamment celles du bouton « mode » qui permet de sélectionner le mode de fonctionnement de la simulation.

- **Fonctions disponibles :** Pour programmer le réseau de pétri, nous avons mis en place des fonctions de haut niveau. La nécessité de ces fonctions a été abordée par les encadrants pour faciliter le travail des étudiants et se focaliser uniquement sur les connaissances de réseaux de pétri et de partage de ressources. Ces fonctions sont récapitulées dans le tableau suivant :

Objet	Fonctions
Robots	void EnvoyerRobot(int numRobot, int position)
	void EnvoyerAngles(int numRobot, int angle1..., int angle7)
	void DescendreBras(int numRobot)
	void MonterBras(int numRobot)
	void FermerPince(int numRobot)
	void OuvrirPince(int numRobot)
	void ControlerRobot(int Position, int numPosition, int bras, int pince)
	int RobotInitialise(int numRobot)
	int RobotEnPosition(int numRobot)
	int BrasEnPosition(int numRobot)
	int PinceEnPosition(int numRobot)
	int TraitementFini(int numTache)
Produits	void PiecePrise (int numPoste)
	void PieceDeposee(int numPoste)
	int ProduitSurNavette(int handle)
Navettes	int NouvelleNavette()
	void ReinitialiserNouvelleNavette
	int NavetteStoppeeVide(int numPoste)
	int NavetteStoppee(int numPoste)
	void NavettePartie(int numPoste)
	void DefinirDestination(int handle, int destination)
	int NavetteDisponible()
	void DestroyShuttle(int numNavette)

Figure 13 - Tableau des fonctions de haut niveau disponibles

Le détail concernant ces fonctions se trouve avec la documentation jointe.

10 Améliorations possibles

10.1 Du point de vue de l'interface utilisateur

Sur la fin du projet nous étions entrains de modifier l'espace dédiés aux produits, nous souhaiterions pouvoir suivre l'état d'avancement des produits plutôt que simplement voir les produits qui sont à fabriquer comme c'est le cas-là. Cela permettrait de se rendre mieux compte de l'état d'évolution du système plus particulièrement lorsqu'il y a plusieurs produits en même temps.

On pourrait également ajouter sur la partie droite de l'interface (et donc agrandir le cadre) l'avancement du fichier log en direct, ou alors via directement un ROS_INFO pour suivre l'état générale du réseau de pétri, en plus du terminal contenant l'état des places.

10.2 Fichier Log

Pour ce qui est du fichier Log, il faut donc créer un programme simple dans un langage au choix (Python est le choix qui nous semble être le mieux car plus simple d'implémentation). Le programme lira les données directement sur le fichier log et les interprétera en fonctions du cahier des charges souhaité. Il vérifiera par exemple que tous les produits ont bien étés crées avec la bonne gamme de fabrication, qu'il n'y a pas eu de conflit de ressources etc. Ainsi des pénalités pour être attribués pour chaque critère du cahier des charges non respecté. Le nombre total de pénalité déterminera la note de l'élève, un score de 0 voudra dire que le réseau de pétri satisfait toutes les attentes du cahier des charges.

10.3 Au niveau de la simulation

Nous avons fait au mieux avec nos connaissances pour accélérer la simulation mais de toute évidence nous aurions besoin d'un expert de V-Rep pour nous donner les clés pour optimiser la simulation. Nous avons fait un annonce durant notre projet mais nous n'avons malheureusement pas eu de réponse à notre requête, nous avons donc fait au mieux.

Pour ce qui est des programmes en eux-mêmes, il serait bien de pouvoir ajouter au niveau de la simulation toute la partie de droite. En effet jusqu'à présent la partie de droite est condamnée par l'aiguillage n°3. Cet aiguillage permet justement d'accéder à la partie de droite et pour l'instant l'aiguillage n°3 sert de sortie pour les produits (en détruisant la navette). Ce qu'il serait préférable serait de rapatrier le programmes de destruction de la navette dans les programmes du nœud Shuttle. Une fois cela fait il faudrait créer tous les programmes concernant les aiguillages de droite ainsi que les postes et les robots, de sorte à ce que la simulation soit utilisée dans sa globalité. Cela permettrait de mettre en place des gammes de fabrication plus complexes mais également permettrait un partage de ressources encore plus

complexe à mettre en place puisque seulement deux rails permettent de transiter entre la partie gauche et la partie droite de la simulation.

Un ménage dans les fonctions serait également à faire puisque beaucoup de fonctions présentent des doublons qui utilisent des ressources mémoires inutiles, l'élément le plus flagrant est la gestion de la « map » des produits et des navettes qui devrait plutôt se faire dans un nœud dédié qui communiquerait les information au différents nœuds en n'envoyant uniquement les information nécessaires pour chaque nœud.

11 Conclusion

Pour conclure, nous sommes heureux d'avoir eu l'opportunité de travailler sur ce projet très intéressant sur le plan technique mais également sur le plan pédagogique au vu de sa connexité avec le cursus CDISC de l'ENSEEIHT.

Nous avons également apprécié l'opportunité de pouvoir travailler sur une maquette commune à plusieurs laboratoires et grandes écoles de a région Toulousaine du fait de la qualité du matériel mais également à travers l'importance accordé à notre travail et à celui de nos prédécesseurs.

Nous avons réussi au bout de ces 6 semaines à livrer un projet fonctionnel et assez proche des attentes de nos encadrants. Il reste néanmoins quelques tâches (citées dans la partie précédente) que nous n'avons pas pu réaliser complètement ou à améliorer au vu du temps imparti. Cependant, nous avons pu les communiquer clairement avec les personnes rattachées à ce projet pour que celui-ci aboutisse au plus vite.

Finalement, le projet long que nous avons eu la chance de réaliser à l'AIP-PRIMECA nous a beaucoup appris. Dans le sens technique bien évidemment ou nous avons consolidé nôtre pratique de Linux mais également découvert de nouveaux langages et logiciels comme ROS et VREP. Ceux-ci pourront nous être très utiles dans le futur notamment par leur large utilisation dans le domaine professionnel en robotique.

De plus, cette expérience nous a également permis de se rapprocher encore plus près du monde professionnel de l'ingénieur et de mieux se préparer à notre futur. De nombreuses situations rencontrées comme l'arrivée dans une nouvelle équipe, des problèmes complexes auxquels nous avons été confrontés nous ont permis de mieux appréhender ce qu'est le "travail d'ingénieur".

Bibliographie

[1] AIP-PRIMECA [En ligne]
http://www.aip-primeca.net/

[2] LAAS-CNRS [En ligne]
https://www.laas.fr/

[3] M. Jean-Baptiste Blanc-Rouchossé, Mlle Claire Delage, M. Enrique Maldonado, M. Maxime Maurin, Mlle Aurélie Quintana, Mlle Céline Tomé *« Commande et Simulation d'un réseau de transport d'un système de production robotisé »*

[4] ANTONIUTTI Emilie, BERTIN Thibault, DEMMER Simon, LE BIHAN Clément Commande et Simulation d'un réseau de transport d'un système de production Rapport de Projet Long, GEA CDISC. Toulouse : ENSEEIHT, 2016. *https://github.com/ClementLeBihan/CelluleFlexible/blob/master/Livrables/Rapport_Projet_Long*

[5] Vinci Energies [En ligne] L'usine du future sera autonome
http://www.vinci-energies.com/cest-deja-demain/pour-une-industrie-intelligente/lusine-du-futur-sera-autonome/

[6] M. Grunow, H.-O. Günther, M. Lehmann Strategies for dispatching AGVs at automated seaport container terminals. Pages 587-610, Oct 2006

[7]