Réf : PFE EIB-5

Autoliv Electronics France

Ecole Nationale Supérieur d'Electronique et ses Applications (ENSEA)
Option troisième année - Electronique Instrumentation et Biosciences (EIB)

Universidade de Brasília (UnB)
Faculdade de Tecnologia (FT)
Departamento de Engenharia Elétrica (ENE)

Auto calibration de la position du robot CEM et implémentation d'un joystick

Lígia Campos Magalhães

Tuteur de stage : Vincent NICORA
Professeur conseillère : Sophie OLIJNYK

Rapport de stage de Projet de Fin d'Etude
Trabalho de Conclusão de Curso em engenharia elétrica
Stage réalisé au sein d'Autoliv du 2 février 2015 au 31 juillet 2015

Cergy Pontoise, France
2015

Lígia Campos Magalhães
Auto calibration de la position du robot CEM et implémentation d'un joystick

Ce rapport de stage de 3ème année de l'Ecole Nationale Supérieure de l'Electronique et ses Applications, a été réalisé dans le cadre d'un projet de fin d'étude du 2 février au 31 juillet 2015 à Autoliv Electronic SAS. Ce rapport sera aussi présenté au département d'ingénierie électrique à l'Université de Brasilia, comme une exigence partielle pour obtenir le diplôme d'ingénieurs en génie électrique.

Coordonnées du service d'accueil :

AUTOLIV ELETRONICS
Bat « LE CRYNIE »
21/23 RUE DU PETIT ALBI Parc SILIC
95801 CERGY PONTOISE CEDEX

Cergy Pontoise, France
2015

Remerciements

Tout d’abord, je remercie Dieu qui m’a donné le don de la vie, la capacité d’apprendre et la volonté de suivre mes objectifs. Ensuite je tiens à remercier mes parents qui m’ont encouragé et épaulé pour réaliser ce stage.

Je remercie toute l'équipe pédagogique et administrative responsable de ma formation à l’UnB (Universidade de Brasilia) et à l’ENSEA, ainsi que les différents intervenants extérieurs pour les connaissances et les apports personnels qu’ils m'ont permis d'acquérir. Je remercie aussi l’ENSEA pour m’avoir donné l’opportunité d’étudier en France pendant deux années et permettre de renforcer mes connaissances au cours de mes études d’ingénieur.

J’aimerai remercier vivement mon maitre de stage, Vincent NICORA, pour sa qualité de travail, ses exigences et son accueil. Grâce aussi à sa confiance, j'ai pu accomplir convenablement les tâches demandées pour la réalisation du projet. Il fut d'une aide précieuse dans les moments les plus délicats. Je remercie également Francis SOPPELSA pour m’avoir acuilli au sein de son service « Product Validation » pendant la durée de mon stage.

J'adresse mes remerciements à ma professeur cosseillère, Sophie OLIJNYK, pour son travail, son écoute et sa disponibilité.

Je remercie tous les employés d’Autoliv Electronics de Cergy pour le temps passé ensemble, les conseils, et la qualité des échanges et de partage. Je tiens à remercier mes amis stagiaires avec qui j’ai pu partager les problématiques de nos projets respectifs et l’aide qu’ils m’ont apportée.

Résumé

Ce rapport présente le travail réalisé au cours de mon PFE de six mois au sein de l'entreprise *Autoliv Electronics SAS* dans le service *Product Validation*, où se trouve le laboratoire d'essais de compatibilité électromagnétique accrédité COFRAC. Pour la réalisation de l'essai immunité aux émetteurs portables des calculateurs d'airbags, le service simule sur un banc de test, un émetteur portable avec une antenne fixé au bout d'un bras robotisé *Adept*. L'objectif de ce stage était de faciliter la manipulation de ce robot par l'utilisateur. Lors de la première partie du stage, j'ai développé une fonction qui fait l'auto calibration du repère de ce robot pour simplifier la création du fichier de positionnement de l'antenne. Le logiciel du robot (V+) peut changer son repère avec trois points. Pour que le robot trouve ces trois points, une méthode de reconnaissance d'un point dans l'espace a été utilisée. Parmi les méthodes étudiées, celle retenue a été d'utiliser une dalle tactile avec un dessin d'un repère et un bouton poussoir. La transformation du repère est faite à partir des formules algébriques. Les langages V+ et LabVIEW ont été utilisé. Dans la deuxième partie du stage j'ai ajouté une nouvelle fonctionnalité au banc de test qui permet à l'utilisateur de bouger facilement le robot avec un joystick. Pour cela, j'ai utilisé un Arduino. Enfin, j'ai testé les deux fonctions ajoutées, elles sont en accord avec le cahier des charges.

Mots-clés : Autoliv, tests CEM, matrice de rotation, repère, V+, Arduino, joystick, Adept, robot.

Sommaire

Table des figures

1. Introduction

Ce rapport montre tout le travail effectué au cours d'un stage de six mois au sein de l'entreprise *Autoliv Electronics SAS*, dans le parc d'activités de Cergy Saint-Christophe. L'objectif de ce stage était de faciliter la manipulation d'un robot par l'utilisateur. Dans la première partie du stage j'ai développé une fonction pour faciliter la création du fichier de positionnement de l'antenne. L'auto calibration du repère du robot a été mise en place pour cette demande. Dans la deuxième partie du stage j'ai ajouté une nouvelle fonctionnalité au robot qui permet à l'uilisateur de bouger librement le robot avec un joystick.

Le rapport contient une présentation de la société *Autoliv* et de sa division *Autoliv Electronic*, ainsi qu'une présentation du service d'accueil : *Product Validation*. Il présente aussi le contexte du stage et les besoins attendu. Nous verrons comment se déroulent les tests CEM au sein du service dans lequel l'outil a été développé. Puis nous parlerons du banc de test *HANDY TRANSMITTER* constitué d'un robot équipé d'une antenne fixée à l'extrémité de son bras. Les besoins du stage visent à améliorer ce banc de test. Le cahier des charges et le diagramme de Gantt du stage seront présentés.

Le travail effectué pour améliorer les systèmes existants sera présenté en détails dans la suite du document. Le travail inclut l'étude des méthodes, l'outil développés, les tests réalisés et aussi l'analyse du résultat trouvé. En conclusion, un bilan personnel et technique sur le stage et ses résultats seront mis en évidence, et aussi des suggestions pour la suite du développement d'outil.

2. Présentation de l'entreprise

Le stage auquel ce rapport se réfère a été fait au sein de l'entreprise *Autoliv*. L'entreprise sera présentée ainsi que le service d'accueil avec l'objectif de définir le contexte du travail développé.

2.1. Historique de l'entreprise

En 1953, la société *Autoliv* a été fondée en Suède comme service de repaire de voiture par Lennart Lindblad sous le nom de *Lindblads Autoservice AB*. Elle était un leader dans le domaine de la ceinture de sécurité depuis 1956. En 1968, le nom de la société est modifié pour *Autoliv AB*. En 1975, *Autoliv* est acquis par *Gränges Weda AB*, qui développait des ceintures rétractiles.

En 1980, *Autoliv* devient une filiale au sein d'*Electrolux* quand ce groupe acquiert *Gränges AB*, et la production d'airbag est commencée. En 1984, le nom de la société est modifié pour *Electrolux Autoliv AB*.

Au cours des années 1980 et 1990, *Autoliv* acquiert des actions et le contrôle de nombreuses sociétés spécialisées dans l'équipement automobile. En 1994, *Electrolux* vend ses actions dans *Autoliv* par une offre publique et les actions sont inscrites à la *Bourse de Stockholm*. Le nom de l'entreprise est modifié pour *Autoliv AB*.

Autoliv AB a été l'une des plus grandes sociétés de sécurité automobile dans l'Europe grâce à la technologie prise des entreprise acquises et aux investissements dans la recherche et développement. En 1997, *Autoliv AB* fusionne avec *Morton ASP* (*Automotive Safety Products*), le premier fabricant d'airbag en Amérique du Nord et en Asie.

Autoliv Inc. est formé à partir de cette fusion.

2.2. Autoliv Inc. aujourd'hui

Depuis sa création *Autoliv Inc.* est le leader mondial de la sécurité automobile. Le groupe *Autoliv* invente et fabrique des systèmes innovants et performants, il est le premier fabricant mondial d'airbags et de ceintures de sécurité. La société *Autoliv* est aujourd'hui implantée dans 31 pays, représentés en bleu foncé dans la *figure 1*, avec près de 31 000 collaborateurs.

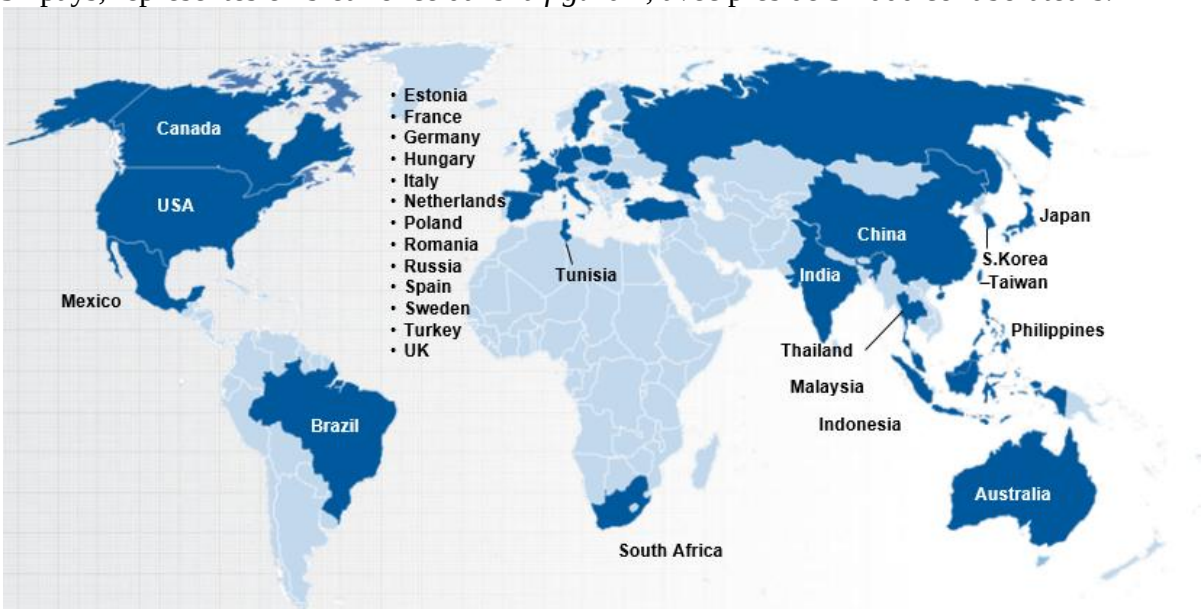


figure 1. Autoliv dans le monde

La vision de la société est de réduire significativement les accidents de la circulation, ainsi que le nombre de morts et de blessés. Sa mission est de créer, fabriquer et vendre des systèmes de sécurité automobile de pointe. Sauver des vies est la valeur principale de la société. Parmi

les valeurs de la société on retrouve aussi les clients, les employés, l'innovation, l'éthique et la culture.

Autoliv a atteint des niveaux de performance et de qualité de premier plan. L'étape suivante pour rester au sommet est le *Q5*, la recherche de la qualité dans ses 5 dimensions : le client, la croissance, le produit, le fournisseur et le comportement. La qualité de chaque dimension est également importante pour le bon fonctionnement de l'entreprise.

Autoliv est une société qui n'arrête pas de se développer. En 2013, ses principales innovations ont été les suivantes :

figure 2. Systèmes de télématique

figure 3. Vision Nocturne

figure 4. Ceinture rétractile avec un contrôle électronique

figure 5. Airbag sur la ceinture

Le premier airbag au monde pour la protection des piétons (figure 6) a été développé chez *Autoliv.*

figure 6. Airbag piétons

La division Électronique du groupe fournit les dispositifs électroniques, tels que les capteurs et unités de commande, et se développe principalement dans le domaine de la sécurité active et passive de la voiture. On peut citer notamment :

- les capteurs et calculateurs de détection de crash et de commande d'airbags
- les électroniques d'enrouleurs électriques
- les caméras et calculateurs pour les systèmes de vision diurne (alerte de franchissement de ligne, alerte collision…) et nocturne (caméra infra-rouge, détection de piétons)
- les radars de détection d'obstacle (pré-crash, freinage automatique,…)

Autoliv Electronics France est située à Saint-Etienne du Rouvray pour son site de production (environ 500 personnes) et sur le site de Cergy-Pontoise pour le centre de Recherche-Développement (environ 170 personnes).

Le rayonnement du site de Cergy est principalement européen pour les projets d'applications et mondial pour la *R&D* de base. Les principaux produits de ce centre de Recherche-Développement sont les suivants :

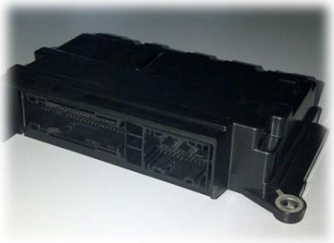


figure 7. Contrôleur d'airbag (ECU)

figure 8. Contrôleur de ceintures intégré

figure 9. Contrôleur d'airbag partie intérieur

figure 10. Capteur de pression

figure 11. Capteurs d'accélération

Les principaux clients sont : Renault, PSA, Volvo, JLR, Ford, GM, Fiat, Audi, Daimler, BMW.

Tous les produits de ce centre ont les certifications et prix remportés suivants : *Spice Level 3 Software*, ISO TS 16949, ISO 14001, ISO 17025

2.3.Présentation du service d'accueil

Le travail présenté dans ce rapport a été fait au sein de l'entreprise *Autoliv Electronics SAS* à Cergy-Pontoise, où est situé son principal laboratoire de test, *Product Validation Laboratoires*, qui est accrédité COFRAC car il satisfait les exigences de la norme NF EN ISO/CEI 17025 : 2005 (Accréditation N° 1-2077).

Le laboratoire est créé en 2004, et depuis 2009 il remplit les exigences de la norme ISO/CEI 17025 : 2005. Le laboratoire travail pour tous les sites d'*Autoliv* et pour les

compagnies externes. Ce service est dirigé par Francis SOLPPESA et l'organigramme du pôle de travail est le suivant :

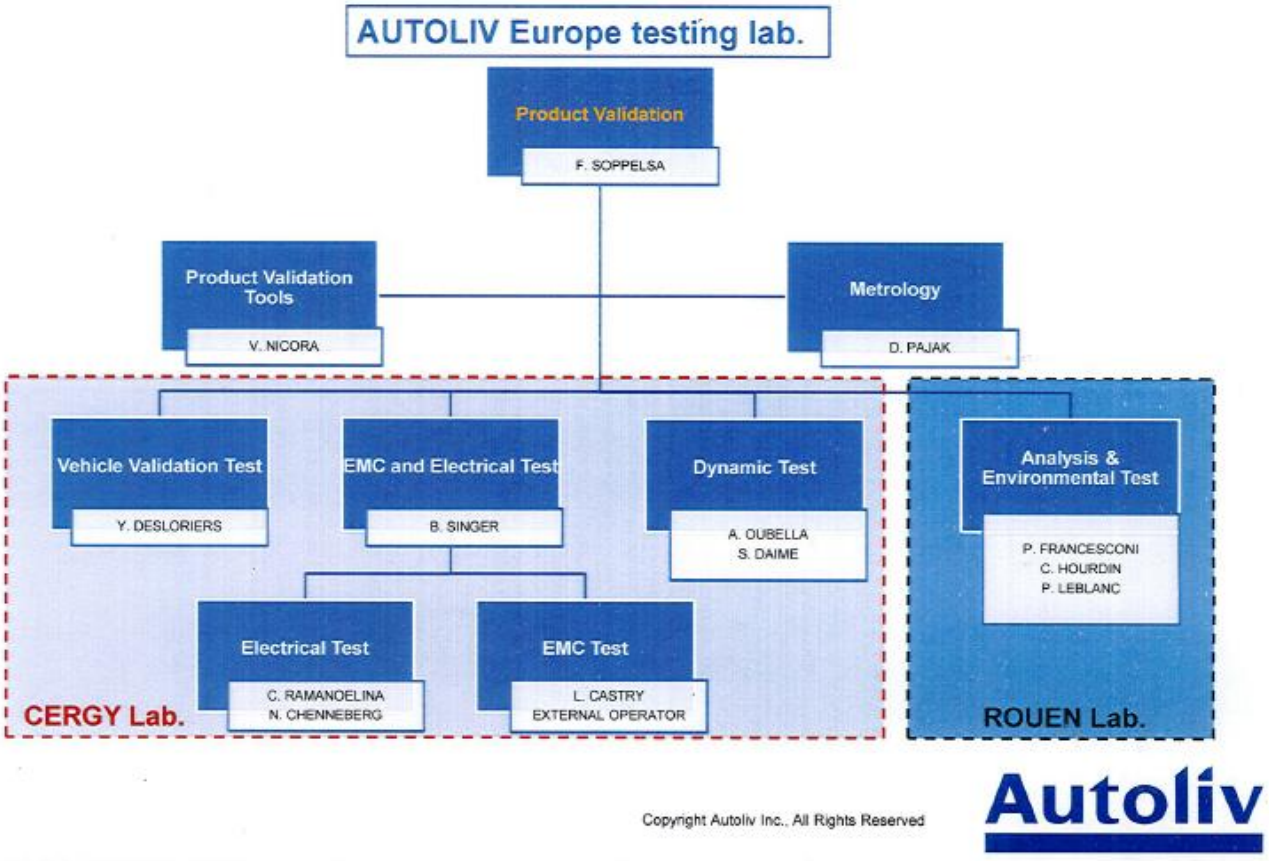


figure 12. Organigramme service d'accueil.

Parmi les principales actions de ce service on retrouve :

- La planification et l'exécution des essais de validation et qualification sur la table durant le développement du produit, avec la rédaction d'un rapport qui contient l'analyse du résultat et proposition des solutions de problème.
- L'analyse des contraintes d'environnement des produits et rédaction des plans de tests en accord avec la demande du client.
- L'assistance aux équipes de conception dans la définition de nouveaux produits.
- Et aussi l'assistance aux clients lors de différentes phases : l'implantation, la validation et la certification de compatibilité électromagnétique du véhicule.

Au sein du service *Product Validation Laboratoires* on peut trouver le service *Product & Validation Tools* où sont développés des outils qui permettent de faciliter et automatiser des tests de différents produits de l'entreprise. Ce service est dirigé par Vincent NICORA, mon tuteur de stage.

Parmi les outils utilisés, nous avons deux robots automatisés : un robot pour les tests d'immunité aux émetteurs portables et un autre robot pour les tests de résistance à des décharges électrostatiques. On retrouve, entre autres, aussi un banc qui permet de réaliser des profils et des coupures sur les alimentations des boîtiers d'airbag, un banc de calibration de la centrale inertielle.

3. Présentation de sujet du stage

Le but de mon stage est de répondre aux besoins du laboratoire des essais de compatibilité électromagnétique (CEM). Dans cette partie du rapport le contexte du stage et les besoins auxquels le stage répond seront présentés.

3.1. Compatibilité électromagnétique (CEM)

Les bruits électromagnétiques et radioélectriques sont le résultat de tous les courants électriques induisant une multitude de champs et signaux parasites. Une bonne compatibilité électromagnétique d'un appareil décrit un état de *bon voisinage électromagnétique* avec à la fois, un niveau limité d'émissions non désirées provenant de l'appareil, ainsi qu'une immunité de cet appareil contre les perturbations provenant des autres équipements, ou de l'environnement.

Les émissions sont liées à la génération de l'énergie électromagnétique par une source dans l'appareil, les contre-mesures devraient être prises afin de réduire cette production et d'éviter la fuite de toutes les énergies non désirées dans l'environnement externe. L'immunité se réfère au bon fonctionnement de l'équipement électrique, dénommée la victime, en présence de perturbations électromagnétiques imprévues.

La compatibilité électromagnétique est la capacité de deux appareils, électriques ou électroniques, à fonctionner dans un environnement donné, chacun générant des perturbations électromagnétiques différentes, et à respecter les normes acceptables de fonctionnement. Les tests CEM permettent de vérifier que les ondes électromagnétiques de l'environnement extérieur n'influent pas sur le fonctionnement des produits développés, ou au contraire, que les produits développés ne perturbent pas l'environnement extérieur.

3.1.1. Les essais CEM

Les diverses réglementations imposent un niveau de compatibilité électromagnétique à respecter. Les méthodes d'évaluation des perturbations, ainsi que les limites de niveau de perturbation à ne pas dépasser ou à supporter dans un environnement donné sont établies par les réglementations.

Le laboratoire d'essais CEM d'*Autoliv* localisé dans le service *Product Validation* à Cergy est accrédité par *Cofrac* selon le périmètre suivant :

- Essais de compatibilité électromagnétique en émission (27-1).
- Essais de compatibilité électromagnétique en immunité (107).

Avec les limitations pour les essais :

- Matériels dont l'encombrement est compatible avec les dimensions du plan de masse.
- Courant consommé inférieur ou égal à 100 A (monophasé, triphasé ou courant continu).

Le service teste donc les appareils afin d'être sûr qu'ils respectent bien ces réglementations, que ce soit en immunité ou en émission. Pour cela, il dispose de trois chambres anéchoïque, une chambre Faraday, deux robots et 3 bancs d'essais pour les tests électriques.

Les principaux essais sont :

- Mesure des émissions rayonnées, mesure en cage à 1 mètre ; mesure des émissions rayonnées en champ magnétique, mesure à 7 cm, ou à 5 cm de chaque face testée.
- Mesure des émissions conduites, mesure directe en tension et en courant.
- Immunité aux perturbations conduites mode commun (méthode BCI).
- Immunité aux perturbations électriques par rayonnement d'énergie électromagnétique en bande étroite.
- Immunité aux perturbations en champ magnétique par rayonnement

- Immunité aux émetteurs portables
- Immunité aux surtensions transitoires sur les lignes d'alimentation et sur les lignes de signaux.
- Immunité aux décharges électrostatiques, appliquer des décharges électrostatiques aux différentes parties de l'appareil en essai.

3.1.2. Le test fait par l'outil développé

L'outil développé dans le stage est le banc de test *HANDY TRANSMITTER*, avec lequel est fait le test d'immunité aux émetteurs portables. Le banc de test simule un émetteur portable qui émet normalement des perturbations électromagnétiques générées par le *GSM, wifi, Bluetooth*. Le test d'immunité est indispensable pour vérifier le bon fonctionnement des différentes parties électronique de la voiture.

Le test consiste à émettre des ondes électromagnétiques plus ou moins proches du produit à valider. Une antenne simule les émetteurs portables. Différentes antennes peuvent être utilisées afin d'envoyer des signaux de différentes puissances à différentes fréquences afin de simuler les différents rayonnements électromagnétiques que peut subir un calculateur d'airbag dans une voiture (rayonnement d'un téléphone portable, d'un talkiewalkie d'un équipement wifi….). Les essais sont faits dans une chambre anéchoïque qui absorbe l'écho et simule les conditions de « champ libre ».

3.2. Outil Existant

L'outil existant est le banc de test *HANDY TRANSMITTER* :

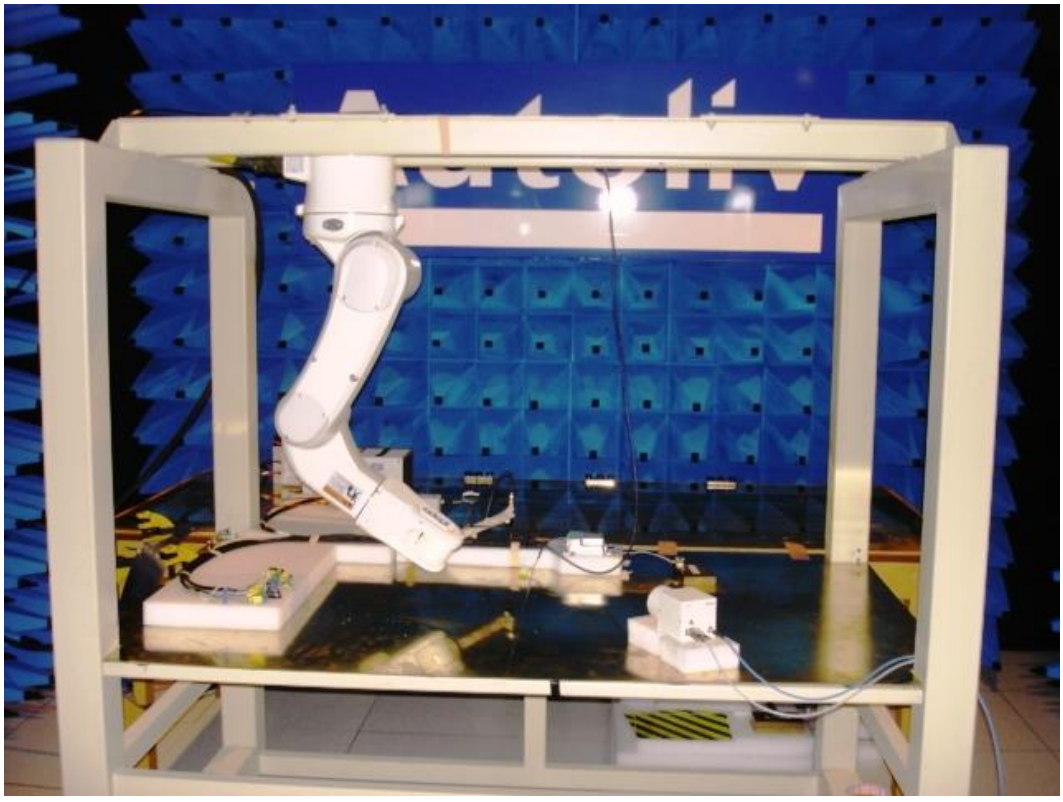


figure 13. *HANDY TRANSMITTER*

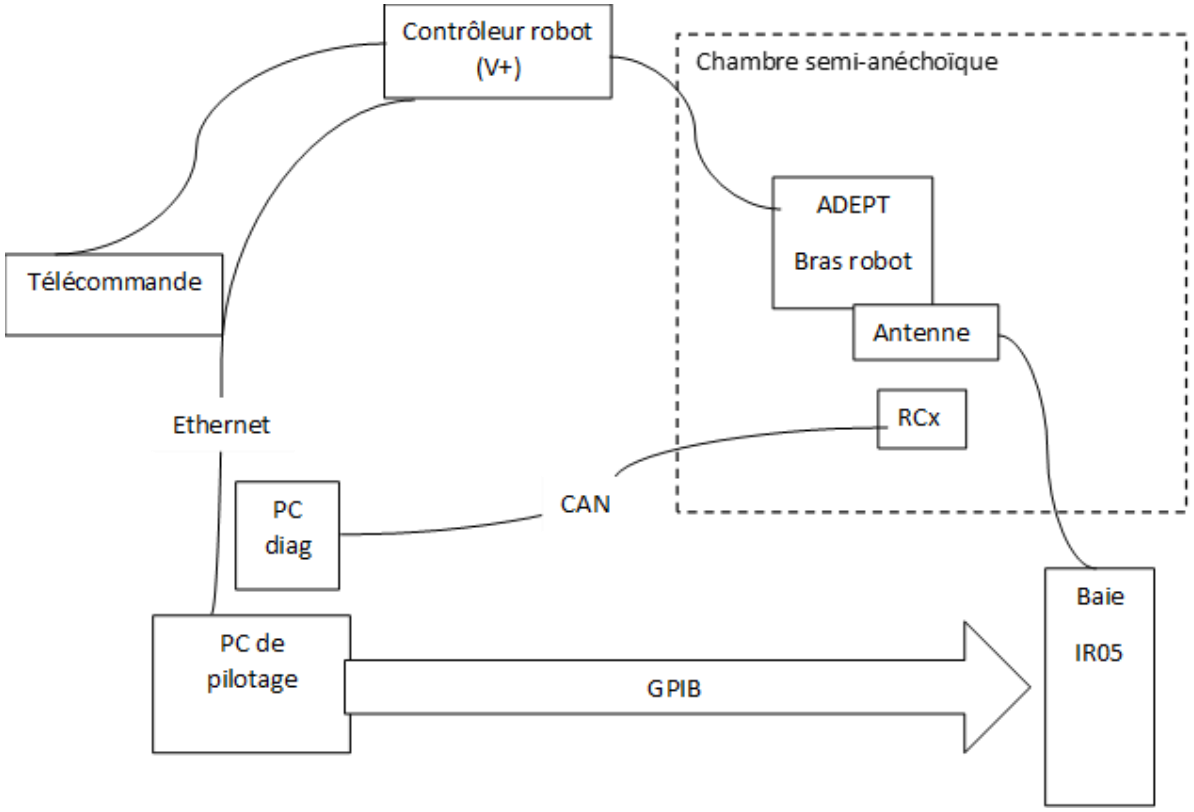


figure 14. Organigramme général d'outil

L'outil existant est composé de :

- Robot Adept, utilisé pour déplacer une antenne autour du produit
- Chambre anéchoïque
- Contrôleur airbag (RCx), ou autre produit à valider
- Baie IR05 (décrite ultérieurement)
- Antenne
- Ordinateur diagnostique.
- Ordinateur de pilotage qui contrôle la baie IR05 et le Robot Adept.

L'utilisateur fait le test CEM via l'*ordinateur diagnostique* et l'*ordinateur de pilotage*. L'*ordinateur diagnostique* contient le software diagnostic du produit automobile acheté par *Autoliv* et l'*ordinateur de pilotage* contient le soft *LabVIEW* développé par les employés d'*Autoliv*.

L'*ordinateur diagnostique* est souvent changé puisque chaque produit et chaque spécification du client exige des essais différents. La liaison entre le produit et l'*ordinateur diagnostique* est souvent fait via bus *CAN* (*Controller Area Network*), mais d'autres types de communication peuvent être utilisés.

L'*ordinateur de pilotage* communique avec l'utilisateur via *LabVIEW*. La communication entre la *baie IR05* et cet ordinateur est faite via *GPIB* (*General Purpose Interface Bus*), un bus de communication numérique à courte distance. La communication entre cet ordinateur et le contrôleur du robot *Adept* est faite via Ethernet. L'utilisateur peut contrôler le robot *Adept* et la *baie IR05* via *LabVIEW*. Le contrôleur du robot *Adept* utilise des programmes codés en *V+*, sa mémoire peut être changée par les programmes *AdeptWindows* et *AdeptACE*. L'utilisateur peut contrôler également le robot via la télécommande A*dept* : « le pendant ».

Le robot *Adept* est équipé d'une antenne fixée à l'extrémité de son bras. Cette antenne, qui simule les émetteurs portables, émet du champ électromagnétique produit par la *BAIE IR05*. Le robot *Adept*, l'antenne et le produit à tester sont placés dans la chambre anéchoïque.

La partie suivante traitera plus en détails la *baie IR05*, le robot *Adep*t et le programme *LabVIEW* développé par les employés d'*Autoliv*.

3.2.1. Baie de génération de fréquence de puissance

La baie de génération de fréquence e puissance, aussi appelée : *baie IR05*, est composée de :

- Un générateur qui génère des perturbations entre 26MHz et 2.7GHz avec une modulation CW, AM ou PM, en fonction des cahiers des charges constructeur.
- Un wattmètre qui mesure la puissance incidente et réfléchie, aidant à calculer le rapport d'ondes stationnaires (ROS) de l'antenne
- Un switch qui change d'amplificateur est utilisé
- Deux amplificateurs qui amplifient les ondes de façon à avoir une puissance suffisamment grande pour être mesurée.

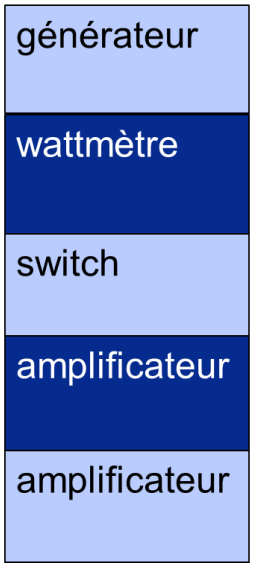


figure 15. Baie IR05

Le rapport d'ondes stationnaires (ROS) exprime la qualité de l'adaptation de l'antenne, à une ligne de transmission, coaxiale ou bifilaire. Ils sont calculés à partir de la puissance incidente et réfléchie de l'antenne mesurée par le wattmètre.

La communication entre l'utilisateur et la baie est fait via *LabVIEW* et GPIB. Le sujet de stage n'inclut aucune modification dans cette partie d'outil.

3.2.2. Le Robot Adept

Le robot *Adept Viper s850* ™ est utilisé pour déplacer l'antenne autour du produit à valider. Il polarise et positionne l'antenne en accord selon le besoin de chaque essai. Le mouvement du robot est orienté par le contrôleur de ce dernier. L'utilisateur peut modifier la position du robot en envoyant une commande par le programme *LabVIEW* ou en utilisant la télécommande du robot.


figure 16. Robot Adept Viper s850 ™

figure 17. Contrôleur du robot

figure 18. Télécommande du robot

Le modèle du robot utilisé est l'*Adept Viper s850* ™ à 6 axes.

Le contrôleur du robot *Adept SmartController CX* est équipé d'une carte *CompactFlash* ™ (CF) amovible. La CF, livrée avec tous les systèmes, est configurée en usine et installée par *Adept*, elle stocke le système d'exploitation V +, *AIM Software*, les programmes d'application, les fichiers de données, et les licences d'*Adept*. C'est dans la partie *programmes d'application* que l'on retrouve les programmes créés par *Autoliv* (codé en *V+*).

Le contrôleur du robot a 6 taches indépendantes. Le système utilisé par l'application des essais CEM a les différentes tâches suivantes :

- La tache 0 est utilisée pour toute la commande de mouvement du robot.
- Dans la tache 1, le programme qui monitore le mouvement du robot par rapport au tableau de positions envoyé par *LabVIEW*, est lancé.
- La tache 2 est dédiée à contrôler les arrêts d'urgences du robot et à allumer le robot.
- La tache 3 contient le programme qui écoute la communication Ethernet et prend la décision pour chaque commande reçue.
- La tache 4 est utilisée pour envoyer les messages du robot via Ethernet.
- La tache 5 est utilisée pour monitorer la télécommande du robot, quand il y a besoin.
- Dans la tache 6 j'ai rajouté une fonction pour gérer la communication en série du joystick.

Le positionnement du robot est spécifié par les valeurs des angles de chaque articulation de celui-ci ou par la position du bout de l'outil du robot. Dans le cas du test CEM l'outil est une antenne. On utilise des antennes de tailles différentes.

figure 19. Antennes patch et sleeves

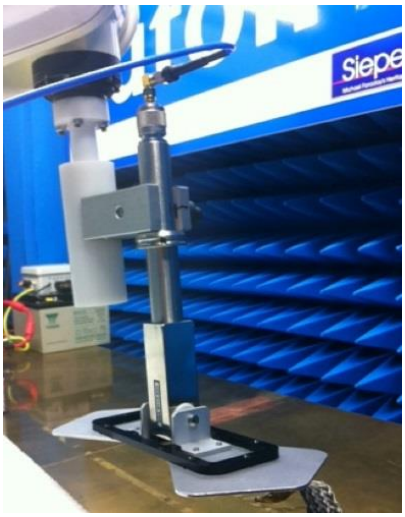


figure 20. Antenne Schwarzbeck

Le positionnement du robot sera expliqué en détail dans la section *4. Position dans le robot*.

3.2.3. Programme LabVIEW

Le programme *LabVIEW* développé par *Autoliv* permet la communication de l'utilisateur avec le contrôleur du robot *Adept* et la baie *IR05*. Il peut choisir d'utiliser le robot, la baie *IR05* ou les deux simultanément.

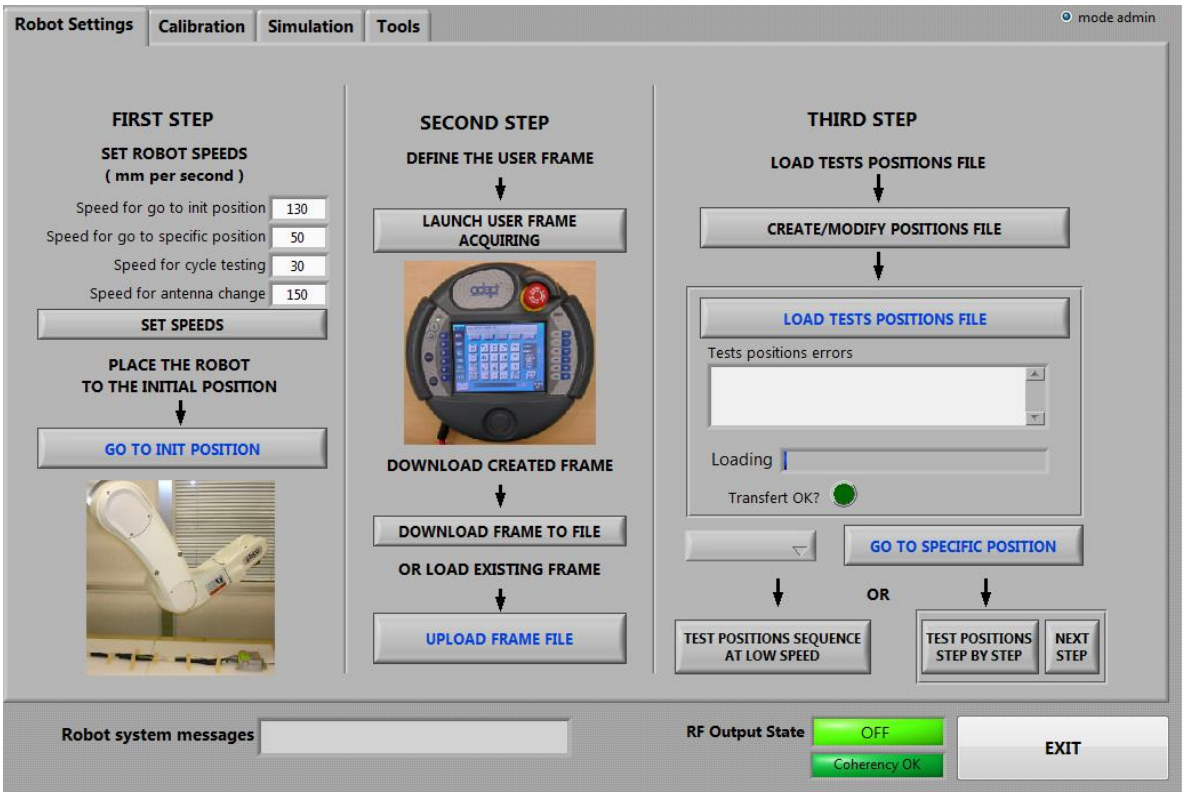


figure 21. Software LabVIEW existant

La configuration du robot est faite dans l'onglet *Robot Settings*. L'utilisateur a la possibilité de choisir la vitesse de travail et de demander au robot de se positionner dans un *point de départ* prédéterminé dans le programme du robot. Le programme *LabVIEW* ne permet pas la modification de ce *point de départ*.

L'utilisateur doit choisir le *repère utilisateur*. Il peut le créer avec la télécommande ou choisir un déjà existant. Ce dernier est un fichier *.fra* où sont sauvegardés les points nécessaires pour créer le repère dans le robot par contre celui créé par la télécommande pourra être sauvegardé par l'utilisateur.

Avec le repère bien défini l'utilisateur peut créer et télécharger le *fichier de positions* par rapport à ce repère. Le *fichier de positions* est un fichier *.txt* qui contient les positionnements de l'outil du robot pour les essais CEM. L'utilisateur peut tester chaque position pour savoir si elle est bien en envoyant le robot à cette position. Il peut aussi tester tout le chemin de l'antenne en mode continue.

Avant de commencer le test de compatibilité électromagnétique, l'utilisateur doit avoir des *fichiers de calibration* des antennes existantes. Il y a un fichier par antenne. Ce fichier contient l'atténuation du système pour chaque fréquence et chaque puissance utilisées (l'atténuation est calculée grâce à la mesure du rapport d'ondes stationnaires (ROS)). La calibration doit être sauvegardée par l'utilisateur dans une archive *.cal*.

Les essais CEM sont faits dans l'onglet *Simulation*. Avant les essais, l'utilisateur doit choisir les fréquences et modulation du signal à rayonner. Pour cela, il peut télécharger les archives *.cfg*. Il a l'option de sauvegarder les configurations des signaux à rayonner pour pouvoir les utiliser après.

Pour choisir quelle antenne utiliser, l'utilisateur a l'option de limiter le ROS des signaux désirés, puis demander au programme *LabVIEW* de trouver parmi les calibrations d'antennes sauvegardées l'antenne la plus adéquate pour chaque signal.

La simulation des émetteurs portable peut être faite en mode continue, ou en mode pas à pas. Dans le mode continu, l'antenne émet des champs électromagnétiques dans tout le chemin choisi dans le *fichier de position*. Dans le mode pas à pas, l'antenne émet seulement quand elle est placée dans une position citée dans le *fichier de position* et l'utilisateur devra choisir le temps pendant lequel l'antenne émet le champ. Le déplacement de l'antenne ainsi que l'émission du champ électromagnétique sont faits automatiquement par le programme *LabVIEW*.

Dans l'onglet *Tools* se trouve trois outils qui permettent :

- La création automatique d'un *fichier de configuration* du signal à émettre *.cfg*. L'utilisateur fournit les fréquences minimum, maximum, le pas de la fréquence, les puissances minimum, maximum, le pas correspondant à la puissance et la modulation du signal. Avec ces informations, l'outil crée une archive qui contient tous les signaux désirés.
- Vérification que *LabVIEW* est capable de communiquer avec le programme qui diagnostique le produit. Normalement l'*ordinateur de pilotage* et l'*ordinateur diagnostique* ne communiquent pas mutuellement.
- Manipulation du générateur de fréquence manuellement.

3.3.Les besoins d'outil

Le banc de test *HANDY TRANSMITTER* fonctionne bien, mais parfois, le robot n'utilise pas le bon repère de travail, ce qui engendre des déplacements aux mauvaises positions. De plus, le déplacement du robot via la télécommande est commandé par des boutons, et sa vitesse maximale est trop basse ce qui gêne le confort de l'utilisateur.

Dans l'outil existant (avant le stage), le robot utilise toujours le même fichier du repère pour le deplacement. Si le support du produit à tester ou la table s'incline, il faut que l'utilisateur mesure l'inclinaison et corrige tous les points du fichier de positions sinon l'antenne peut se cogner contre le produit et se casser.

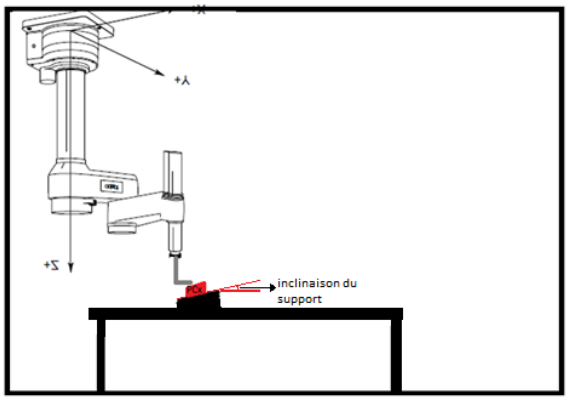


figure 22. Exemple d'inclinaison du support du produit à tester

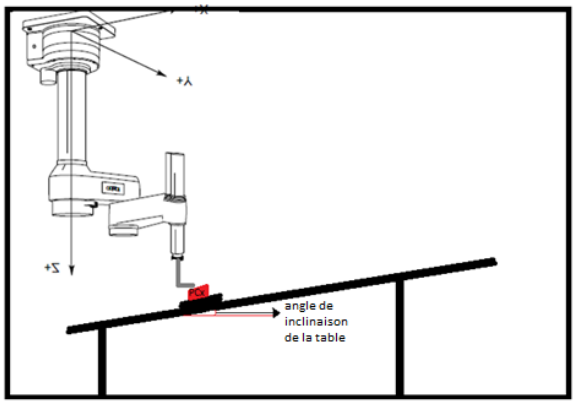


figure 23. Exemple d'inclinaison de la table

L'objectif de ce stage est de faciliter la manipulation d'un robot par l'utilisateur. Dans la première partie, l'objectif est de faciliter la création du fichier qui donne le positionnement de l'antenne au robot pour chaque test et aussi de refaire le repère de travail du robot. A la fin de cette partie, le repère sera le plan du contrôleur d'airbag à tester. J'ai conçu une fonction qui

permet de refaire un nouveau *repère utilisateur* à chaque début de test, en réduisant le nombre d'interventions de l'utilisateur le plus possible.

Dans la deuxième partie, j'ai ajouté une nouvelle fonctionnalité au robot celle de pouvoir bouger librement avec un joystick. L'implémentation de ce joystick était en grande partie logicielle. J'ai choisi l'*Arduino* pour gérer les informations du joystick et des boutons poussoirs, ainsi que pour envoyer ces données au contrôleur du robot via un port serial. De plus, j'ai conçu une carte pour placer et connecter les boutons poussoirs à l'*Arduino*.

3.3.1. Cahier des charges

On a défini le cahier des charges pendant une réunion avec tous les utilisateurs du banc de test. Les exigences pour chaque partie sont décrites dans la suite du rapport.

3.3.1.1. Auto calibration du robot CEM

Il faut changer le mode de fonctionnement des essais CEM afin de créer, d'une façon plus simple et intuitive, des fichiers contenant des points. Pour chaque produit à tester et pour chaque positionnement, il faudra que l'utilisateur crée un nouveau repère. Le nouveau repère doit être fait à chaque fois que l'on allume le robot et que l'on fixé l'antenne.

Il faut veiller à ne pas détériorer le fil de l'antenne lors des différentes phases de calibration. L'antenne la plus utilisée est la *Schwarzbeck*. La priorité est de faire fonctionner le système avec celle-ci. Néanmoins il faut que les méthodes soient utilisables avec les autres types d'antennes.

figure 24. Antenne Schwarzbeck

Pour l'auto calibration, il faut permettre à l'utilisateur de mettre le capteur dans l'antenne puis de l'enlever avant de commencer le test.

Lorsque le programme d'auto calibration sera prêt, il faudra immobiliser l'antenne et faire un nouveau support. Il faut modifier le point de départ du robot CEM afin que le support d'antenne soit perpendiculaire au plan de masse.

Le développement de l'auto calibration se fera en trois étapes :

- Détermination d'un repère perpendiculaire au plan de masse
- Détermination d'un repère non perpendiculaire au plan de masse
- Après calibration incliner l'antenne en fonction du repère créé

Le programme d'auto calibration permettra de créer le repère avec différentes orientations.

3.3.1.2. Implémentation d'un joystick

L'objectif du joystick est de commander les mouvements du robot d'un mode plus simple et intuitif que ce que fait la télécommande. Le joystick aura la fonction principale de bouger le robot en 3D. On n'a pas besoin de la fonction de mouvement de chaque articulation qui est dans la télécommande.

Il faut avoir des boutons pour choisir le mouvement exclusif sur un axe et ainsi l'utilisateur peut bouger le robot plus facilement suivant une droite. On peut choisir un joystick 3D qui a un potentiomètre pour l'axe Z comme pour les axes X et Y. Si on choisit un joystick 2D il faut avoir un bouton pour bouger le robot sur l'axe Z.

La vitesse doit augmenter progressivement si le joystick est dans la valeur maximale. Il faut avoir un potentiomètre pour commander la vitesse maximale à la main. On peut ajouter une fonction dans le joystick pour jouer avec l'orientation de l'antenne.

L'idéal est la connexion du joystick directement dans le contrôleur du robot. Pour faire ça, on aura besoin d'un microcontrôleur. Dans un premier temps, on va simuler un joystick dans *LabVIEW*, ensuite on connectera un joystick à l*'ordinateur de pilotage*. Lorsque ces fonctionnalités seront validées, on peut développer une carte pour que le joystick soit connecté directement au contrôleur du robot.

3.3.2. Diagramme de Gantt

Le diagramme de Gantt établie est le suivant :

Diagramme de Gantt général par mois				date final :	31/07/2015	
	février	mars	avril	mai	juin	juillet
étude du projet existant	■					
faire un programme qui construit un nouveau repère avec la télécommande du robot.		■				
Rendre l'auto calibration du robot automatisée, avec une dalle tactile.			■	■		
implémenter un joystick afin de bouger le robot en 3D.					■	■

Au cours des premiers mois, une étude du contexte du stage est réalisé et la définition du cahier des charges, ainsi que le diagramme de Gantt. Le mois de mars sera dédié à mettre en place un programme de création de repère par LabVIEW avec la télécommande, et à choisir une méthode automatique. La mise en place de la méthode automatique sera faite pendant avril et mai.

L'implémentation du joystick sera développée après la première partie du stage. Les mois de juin et juillet seront dédiés au joystick.

Diagramme de Gantt détaillé février et mars par semaine									
	02/02	09/02	16/02	23/02	02/03	09/03	16/03	23/03	30/03
Etudier les programmes existant									
Faire le cahier des charges									
Faire un programme qui fait le repère à partir de la touche des points									
Fabriquer ou acheter le capteur									
Implémenter le programme									
Valider le programme									

Diagramme de Gantt détaillé avril par semaine									
	06/04	13/04	20/04	27/04	04/05	11/05	18/05	25/05	01/06
Faire le programme qui construit le repère automatiquement									
Implémenter le programme dans le logiciel *LabVIEW* et *V+*									
Faire un nouveau support pour l'antenne *Schwarzbeck*									
Valider le programme pour le repère perpendiculaire à la table									
Faire les ajustements pour valider le programme pour repère non perpendiculaire au plan de masse									
Faire les ajustements pour qu'après calibration l'antenne soit inclinée en fonction du repère.									

Diagramme de Gantt détaillé juin et juillet par semaine					date final :		31/07/2015	
	08/06	15/06	22/06	29/06	06/07	13/07	20/07	27/07
Simuler dans *LabVIEW* le joystick								
Faire le programme *V+* pour recevoir de joystick simulé dans *LabVIEW*								
Choisir un joystick								
Implémenter le joystick dans *LabVIEW*								
Valider le programme								
Modifier le mode de faire les fichiers de positions								
Faire tous les ajustements nécessaire dans *LabVIEW* pour que le programme soit plus simple et avec tous fonctionnalités								
Faire la documentation d'instruction de l'utilisateur du programme *LabVIEW*								

4. Analyse du fonctionnement du robot : comment le positionner

La connaissance du logiciel du robot (V+) est fondamentale pour atteindre l'objectif principal du stage. En effet, le but est de faire bouger le robot selon les positions désirées par l'utilisateur. Pour cela, j'ai analysé, minutieusement, le fonctionnement du robot par rapport aux commandes de positionnement.

Cette partie du rapport décrit les types de positions dans le langage *V+*, comment ils sont associés à l'espace dans lequel se déplace l'unité mécanique, et aussi, comment le *repère utilisateur* est créé dans le contrôleur du robot.

4.1. Définition de l'outil du robot

L'outil en bout de bras du robot est l'antenne. Si les dimensions de l'outil sont connues, la commande *TOOL* peut être utilisée pour le décrire. La dimension *null* de l'outil équivaut à mettre son centre à la surface de la bride du montage et ses axes de coordonnées parallèle à celui de la dernière articulation du robot.

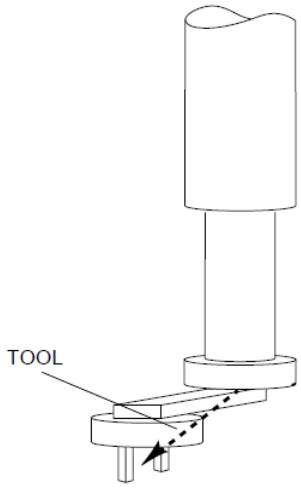


figure 25. Définition d'outil

Si l'outil, par exemple, a des doigts t à 50 mm en dessous de la bride et de 100 mm à la direction x, et qu'on veut changer le réglage de l'outil pour compenser les déportées, il faut écrire la commande :

TOOL TRANS (100, 0, -50)

4.2. Types de position

Une position peut être spécifiée en utilisant soit

- Une *variable de transformation*
- Un *point de précision*, en spécifiant l'état dans lequel devraient être les axes du robot lorsqu'une certaine position est atteinte.

Les positions utilisées dans l'outil développé sont les points du *fichier de points*, les positions du *repère utilisateur* et le *point de départ* du robot. Ce dernier est un *point de précision* pré déterminé dans le programme du robot (*#depart1*), les autres sont des *variables de transformation.*

4.2.1. Variable de transformation

La variable de position *Adept* de base (la *variable de transformation*) est un ensemble de six composants, chacune identifiant individuellement une position dans l'espace cartésien et l'orientation d'outil en bout de bras à cette position.

« TRANS(X, Y, Z, lacet, basculement, roulis) »

Les trois premiers composants d'une *variable de transformation* sont les valeurs des points sur les axes X, Y, Z. Les trois autres composants spécifient l'orientation de l'outil en bout de bras ou ce qui peut être considéré comme étant le repère de référence local (TOOL).

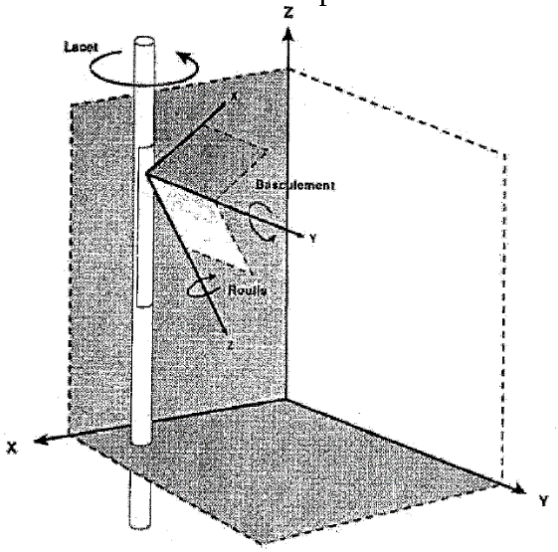


figure 26. Lacet, basculement et roulis

Ces trois autres composants sont le *lacet*, le *basculement* et le *roulis.* On entend par *roulis* la rotation autour de l'axe Z de ce repère de référence local (TOOL). On entend par *basculement* le mouvement de déviation de haut en bas, autour de l'axe Y du repère de référence local (TOOL). On entend par *lacet* la rotation du repère de référence local parallèle à l'axe Z du repère de référence principal (WORLD). Cette rotation ne s'effectue pas autour de l'axe Z du repère de référence principal mais est centrée à l'origine du repère de référence local (TOOL).

Les *variables de transformations* sont des positions relatives au système de coordonnées de référence, appelé repère cartésien (WORLD). La variable de transformation peut être définie relativement à une autre variable de transformation ou à un autre repère, le *repère utilisateur*. La création de ce repère dans le robot sera expliqué dans la section *4.3. Repère utilisateur*.

4.2.2. Point de précision

La variable de précision comporte six valeurs car le robot utilisé est de 6 axes. Chaque valeur correspond à l'angle de l'axe du robot.

La méthode la plus directe pour créer une variable de position consiste à placer l'unité mécanique sur la position à enregistrer comme variable de position, et d'entrer la commande du système d'exploitation suivante *HERE nom_pos*. Où *nom_pos* est le nom de la variable de la position. Le nom d'un *point de précision* doit commencer par *#*.

Si des positions doivent être renseignées au robot ou si des données d'un autre type doivent être rendues permanentes, les données dans la *mémoire système* doivent être sauvegardées dans un fichier-disque avant d'exploiter le robot avec un autre programme ou désactiver le système (éteindre le contrôleur du robot).

Le *point de départ* déterminé avant le début du stage n'était pas correct car l'orientation de l'antenne dans cette position n'était pas parallèle au *repère utilisateur* sauvegardé. Pour le changer, j'ai créé un programme avec la commande *HERE #depart1* que j'appelle quand le robot est positionné parallèlement à la table. Le programme modifie la position dans la mémoire système de la *CompactFlash* du contrôleur du robot. Avant d'éteindre le contrôleur du robot, je sauvegarde la *mémoire système* dans un fichier-disque. Ainsi le *point de départ* est définitivement modifié dans le fichier-disque utilisé par l'outil *HANDY TRANSMITTER.*

4.3. Repère utilisateur

Le *fichier de position* téléchargé par l'utilisateur dans le programme *LabVIEW* contient des positions relatives au *repère utilisateur*. *LabVIEW* envoi chaque point au système d'exploitation du robot qui crée un tableau de *variable de transformation*. Les nouvelles *variables de transformation* sont définies relativement au *repère utilisateur* du robot.

Pour modifier ou créer le *repère utilisateur*, il faut utiliser l'instruction *FRAME*. Cette instruction crée un système de coordonnées en utilisant seulement les localisations de quatre positions, sans utiliser leurs orientations. Deux positions peuvent être identiques, par conséquent, pour déterminer un repère utilisateur il faut trois points.

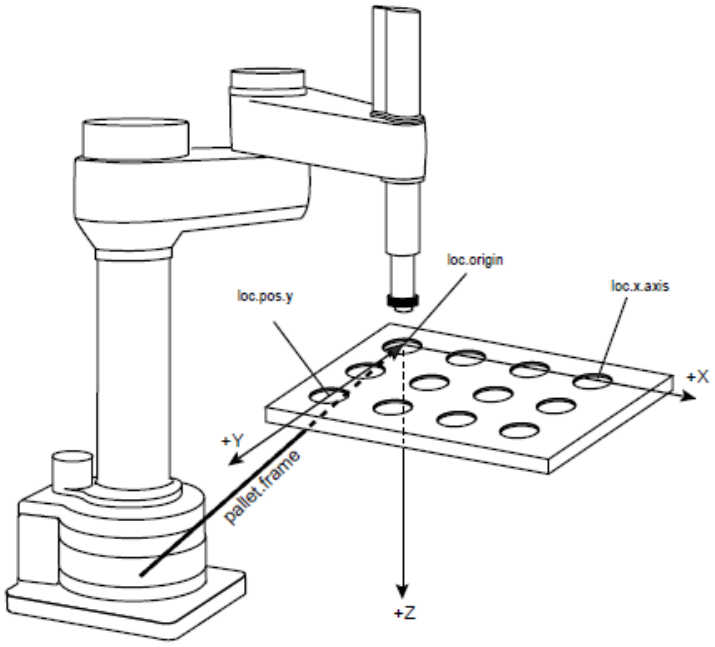


figure 27. Exemple de repère utilisateur

SET pallet.frame = FRAME (depart_x, dir_x, dir_y, origine)

Le traitement de l'instruction *FRAME* (de gauche à droite) implique la détermination des transformations d'orientation et de position du système de coordonnées comme suit :

- Une droite entre *depart_x* et *dir_x* détermine deux éléments : Cette droite sera parallèle au futur axe X du système de coordonnées. Elle pourrait devenir cet axe x. Le sens allant de *départ_x* vers *dir_x* définit la position positive de l'axe x
- Une droite tracée perpendiculairement à la droite ci-dessus et dirigée ver *dir_y* détermine trois éléments : Cette droite sera parallèle au futur axe y du système de coordonnées. Elle pourrait devenir cet axe Y. Le sens allant vers *dir_y* définit la direction positive de l'axe Y. La direction positive de l'axe Z est également connue. Utiliser l'axe X, l'axe Y et la règle de la main droite pour obtenir un repère direct. On connaît maintenant l'orientation du système de coordonnées.
- L'origine correspond aux valeurs nulles des axes X, Y et Z. On peut utiliser la valeur *depart_x* et ainsi on aura besoin de trois points pour déterminer le repère utilisateur. On connaît maintenant la position du système de coordonnées.

Si les coordonnées par rapport au *repère utilisateur* sont connues on peut déterminer les *variables de transformations* (positions du robot) par l'instruction *TRANS* :

SET position = pallet_frame : TRANS (X, Y, Z, lacet, basculement, roulis)

Avec *X, Y, Z, lacet, basculement, roulis* par rapport le *repère utilisateur pallet_frame*.

5. Réalisation première partie : Auto calibration du robot

Cette partie du rapport présente le travail développé pour faire l'auto calibration du robot CEM. Dans un premier temps, je parlerai des capteurs et méthodes trouvés pour permettre l'auto calibration. Ensuite on retrouve le système choisi et l'algorithme développé. Le test et les résultats seront présentés dans la section *7. Tests dans la chambre des essais CEM*.

5.1. Recherche de capteur et méthodes

L'intérêt de la première partie du stage est de créer un repère utilisateur dans le robot, où les positions de trois points sont nécessaires. Ces points sont appelés : *origine*, *dirx* et *diry*. Pour chaque système, il faut placer les capteurs sur ces points et mettre en place une méthode pour la détection des points. Les systèmes étudiés sont :

- Un système avec un palpeur (bouton poussoir) et un joystick.
- Le mélange de trois systèmes : ultrason, infrarouge et palpeur.
- Un système avec un laser, deux miroirs et quatre photodiodes.
- Un système avec une dalle tactile et un palpeur.

5.1.1. Système avec un palpeur et un joystick

Ce système est composé d'un joystick et d'un bouton poussoir fixé au bout de l'antenne. Les trois points du repère sont choisis par l'utilisateur. L'utilisateur doit placer le robot au-dessus d'un point du repère à l'aide d'un joystick, puis demander au robot de descendre vers le point. Le robot s'arrête lorsque le bouton poussoir est activé. L'utilisateur doit vérifier si le robot a touché correctement le point désiré et le confirmer au logiciel. Quand le robot touche le point correctement, ce point est défini dans le logiciel comme un point du repère. Dans le cas contraire, le logiciel refait les mêmes étapes pour le même point. Le cycle recommence pour définir les autres points du repère. Après la définition des trois points, le logiciel crée le *repère utilisateur*.

Lorsque l'utilisateur lance le programme, il suit l'organigramme suivant :

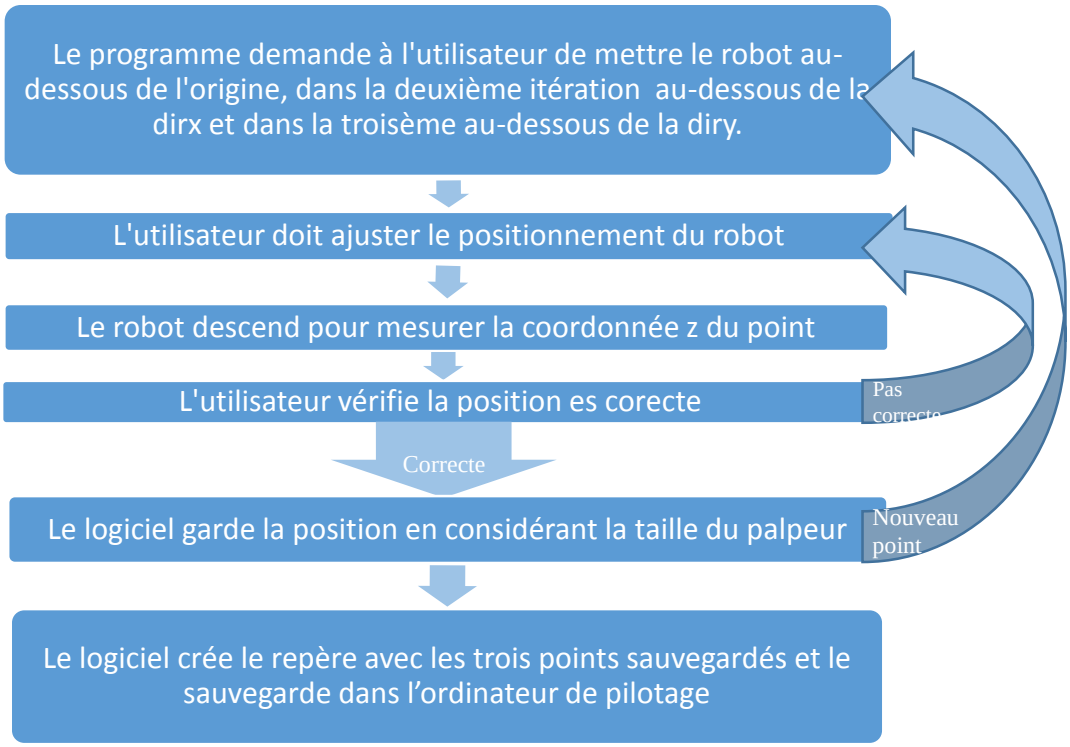


figure 28. Organigramme système avec un palpeur et joystick

Ce système a été implémenté pendant le mois de mars. Au lieu du joystick, j'ai utilisé la télécommande pour faire bouger le robot. L'inconvénient de cette méthode est la nécessité de les plusieurs interventions de l'utilisateur. Pour automatiser l'auto calibration du robot, j'ai étudié d'autres systèmes.

5.1.2. Le mélange de trois systèmes (ultrason, infrarouge et palpeur)

Ce système est composé de trois types de capteurs. Chaque capteur a un inconvénient qui empêche la création d'un repère automatiquement et/ou d'une manière suffisamment précise. Les trois capteurs sont :

- Ultrason : La précision pour trouver les points n'est pas suffisante (1 centimètre) mais l'angle de détection est grand (30 degrés). L'intervention de l'utilisateur est faite seulement au début pour placer le robot proche au premier point. Comme l'angle de détection est grand, deux capteurs ultrason avec une distance de 10 centimètres entre eux peuvent détecter un objet à 18 cm de hauteur du centre d'un capteur. Ainsi, après avoir trouvé un point du repère, le robot peut s'éloigner du capteur et le prochain capteur peut le détecter.
- Infrarouge : ce capteur est plus précis (3 mm à 20 mm de distance) et l'angle de détection est plus petit (10 degrés). Un système composé uniquement par ce capteur demande l'intervention de l'utilisateur à chaque point pour bouger le robot, mais l'utilisateur n'est pas obligé d'être précis.
- Bouton poussoir : précision dépend de la taille du palpeur. Pour faire un système qui utilise seulement le bouton poussoir comme capteur, l'utilisateur doit commander le robot pour arriver à chaque point avec une grande précision.

Pour cela, j'ai trouvé une solution faisant un système avec ces trois capteurs pour contourner leurs inconvénients. Ce système contient, dans chaque point du repère, un ultrason, une photorésistance et un bouton poussoir, et dans l'antenne, un photoémetteur et un bouton poussoir. Le point du repère est défini quand le bouton poussoir de l'antenne touche celui du point du repère. La précision du système sera la taille des palpeurs.

Au milieu du capteur ultrason, je place la photorésistance sur laquelle se trouve le bouton poussoir. Le photoémetteur aura aussi un bouton poussoir et est placé au bout de l'antenne.

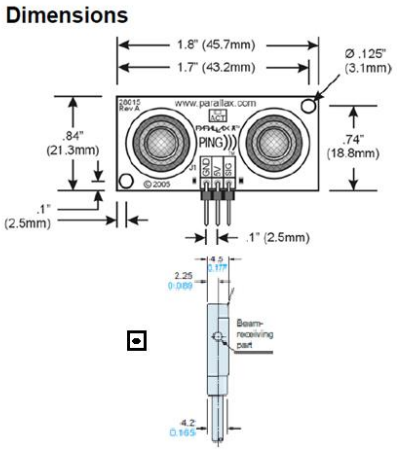


figure 29. Ultrason, photorésistance et bouton poussoir.

L'algorithme est représenté dans le schéma de la « figure 30 » et expliqué par la suite.

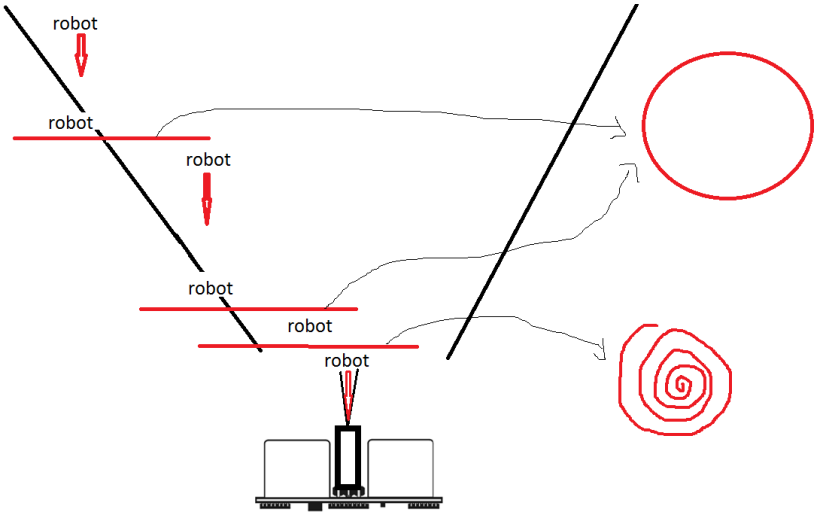


figure 30. Algorithme système avec trois capteurs

1) L'utilisateur place l'ensemble des trois capteurs, pour chacun des points du repère, sur le calculateur d'airbag.
2) L'utilisateur place le robot sur le premier capteur ultrason.
3) Le robot descend jusqu'au point où l'ultrason ne le détecte plus.
4) Le robot fait un cercle et le logiciel sauvegarde les distances de chaque position.
5) Le robot va à la position la plus proche du capteur et retourne à l'étape « 3) ». Ce processus se répète jusqu'à ce que la photorésistance détecte le robot, et passe à l'étape « 6) ».
6) Le robot suit une trajectoire quelconque pour trouver le centre de la photorésistance.
7) Le robot descend jusqu'au point où les deux palpeurs se rejoignent et cette position sera sauvegardée.
8) Le robot monte.
9) Le programme éteint les capteurs du point sauvegardé et allume ceux du prochain point à trouver. Puis, retourne à l'étape « 3) ».
10) Après avoir définit les trois points, le programme crée le repère dans le logiciel du robot.

Les inconvénients de cette méthode résident dans le fait que l'antenne réfléchit les ondes sonores aléatoirement, et par conséquent loin du capteur, et ainsi que le besoin de plusieurs capteurs.

5.1.3. Système avec laser

Parmi les méthodes que j'ai étudiées, on retrouve un système qui contient deux lasers sur deux piliers, deux barillets optiques motorisés et quatre photodiodes omnidirectionnelles. Les lasers scannent en permanence la table, à la mi-distance entre la base du robot et cette table. Pour cela, deux barillets (miroirs) optiques montés sur un axe Tilt motorisé sont utilisés. Sur chaque point du repère voulu, l'utilisateur place une photodiode omnidirectionnelle, et la dernière photodiode est localisée au bout de l'antenne.

figure 31. Barillets optiques motorisés

Les lasers vont scanner les photodiodes plusieurs fois par seconde. Au moment de la détection de la photodiode par le laser, un micro enregistre l'angle du barillet et la distance (deux dimensions). Par corrélation avec le second laser-barillet, le programme calcule la troisième dimension.

L'algorithme rapproche la photodiode de l'antenne vers la première photodiode sur le produit à tester. Lorsque les coordonnées calculées dans l'étape d'avant sont les mêmes, la photodiode de l'antenne touche celle du produit, puis on retient les coordonnées du robot qui deviennent le premier point du repère. Il faut répéter le processus pour les autres points du repère.

C'est un peu le même concept que le fonctionnement des imprimantes laser.

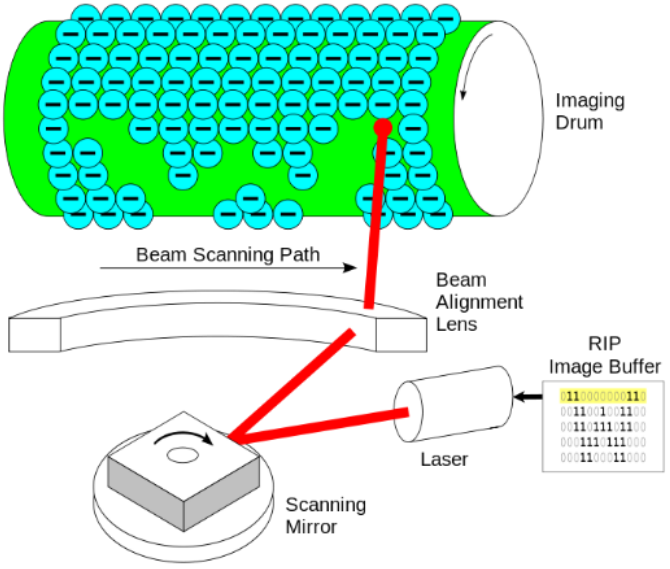


figure 32. Fonctionnement d'une imprimante à laser

Le développement des parties : acquisition de données et contrôle du moteur requièrent plus de temps que prévu pour le développement d'auto calibration. Ce système pouvait être choisi si je n'avais pas eu à implanter le joystick dans le robot.

5.1.4. Système avec une dalle tactile et un palpeur

Le système avec la dalle tactile consiste à placer celle-ci sur le produit à tester et à placer un palpeur au bout de l'antenne. J'ai dessiné un repère sur la dalle tactile pour que l'utilisateur puisse choisir où va être le *repère utilisateur*. Voici, les étapes prévues pour cela :

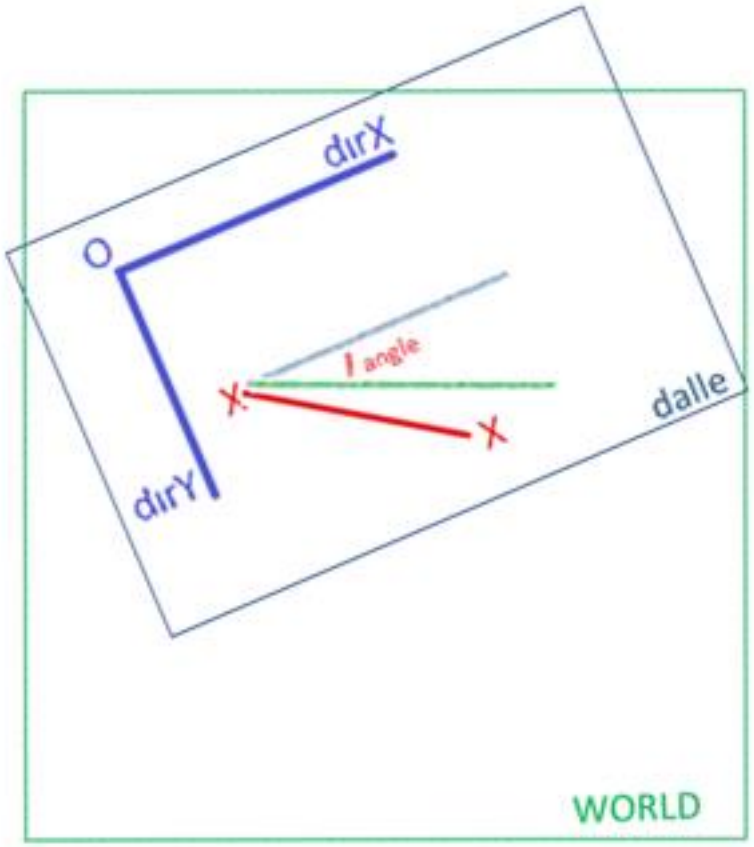


figure 33. Système avec une dalle tactile

1) L'utilisateur doit placer le robot au-dessus de la dalle et lancer l'auto calibration.
2) Le programme d'auto calibration fait descendre le robot jusqu'à ce qu'il touche un point sur la dalle.
3) Le robot se déplace et touche un autre point de la dalle.
4) Le logiciel calcule l'angle entre le repère de la dalle et celui du robot avec ces deux points.
5) Et ainsi le programme calcule la position du point du repère de la dalle par rapport au repère du robot et demande à ce dernier de toucher ce point.
6) Le logiciel crée le *repère utilisateur* avec les trois points calculés.

Dans le commerce, il existe des dalles tactiles qui font bouger la souris de l'ordinateur et j'ai pu récupérer la position de la souris grâce au langage LabVIEW ce qui permettra de détecter la position du robot sur la dalle.

5.2. Système choisi – partie hardware

J'ai choisi le système avec la dalle tactile et le bouton poussoir pour sa facilité d'acquisition des données avec *LabVIEW*. Le système développé pour créer le *repère utilisateur* contient une dalle tactile qui a été achetée avec son contrôleur, un bouton poussoir et ses supports.

Cette partie du rapport présente les capteurs que j'ai choisis et leurs principales caractéristiques.

5.2.1. Caractérisation de la dalle

La dalle tactile choisie est *Keytec Magictouch Add On Touch Screen*.

figure 34. Dalle tactile choisie

La dalle fait bouger la souris. La position de celle-ci par rapport à la dalle tactile dépend de la résolution de l'écran de l'*ordinateur de pilotage* et de la calibration de la dalle. Je fais la calibration grâce au programme installé sur l'*ordinateur de pilotage* à partir des drivers de la dalle tactile.

Le programme *LabVIEW* récupère les valeurs des coordonnées X et Y de la souris en pixels. La proportion entre l'axe X et l'axe Y est différente : un mouvement d'un centimètre vers l'axe X fait bouger la souris d'un numéro de pixels différent de celui du même mouvement vers l'axe Y.

Les bords de la dalle sont instables, chaque touche dans le même endroit proche du bord de la dalle fait bouger la souris dans une position différente. Pour savoir où est la zone stable et linéaire de la dalle, j'ai fait un essai de caractérisation de celle-ci.

Le processus de caractérisation est le suivant :

1) On positionne la dalle de sorte que son repère soit parallèle au repère du robot et le robot sur un *point 1* différent à chaque essai.
2) On programme un logiciel de telle sorte que lorsque le robot bouge en direction de l'axe des X, il récupère le facteur X. De façon analogue, il récupère le facteur Y.

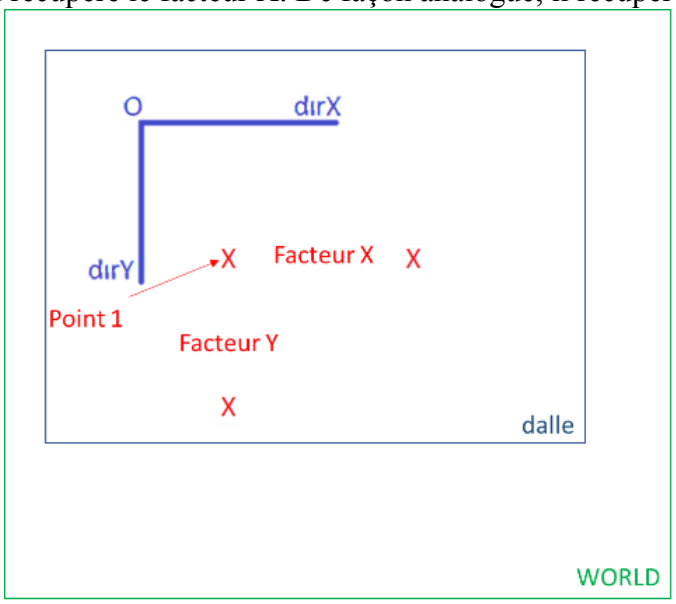


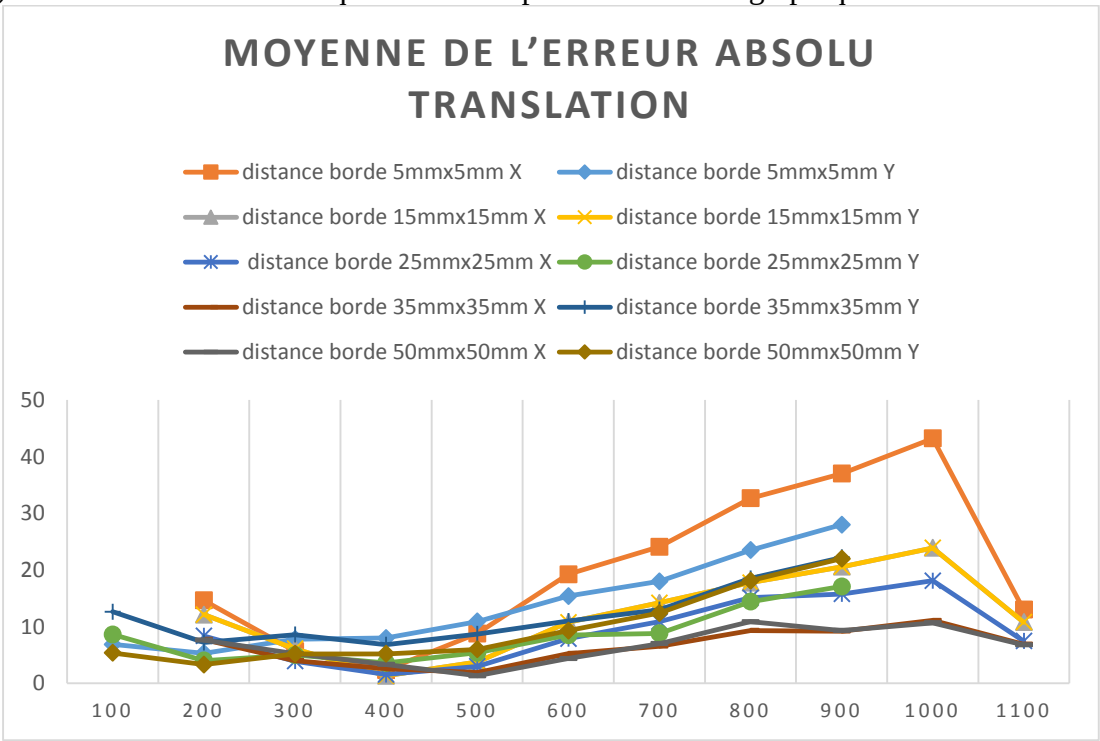
figure 35. Début de la méthode de caractérisation de la dalle tactile

3) Le logiciel calcule les coordonnées X et Y du robot pour qu'il touche les coordonnées X et Y de la dalle de la façon suivante :

$$X_{robot} = (X_{dalle} - X_{point1-dalle}) * (facteur\ X) + X_{point1-robot}$$
$$Y_{robot} = (Y_{dalle} - Y_{point1-dalle}) * (facteur\ Y) + Y_{point1-robot}$$

Avec les facteurs X et Y mesurés et les coordonnées du *point 1*. J'ai fait en sorte que le robot touche toute la dalle avec un écart de 100 pixels (vers 26 millimètre).

4) L'erreur absolue de chaque touche est présentée dans le graphique 1 :



Graphique 1. Caractérisation de la dalle tactile

Ces essais montrent que les positions stables et linéaires se trouvent à une distance de 35 millimètres des bords de la dalle.

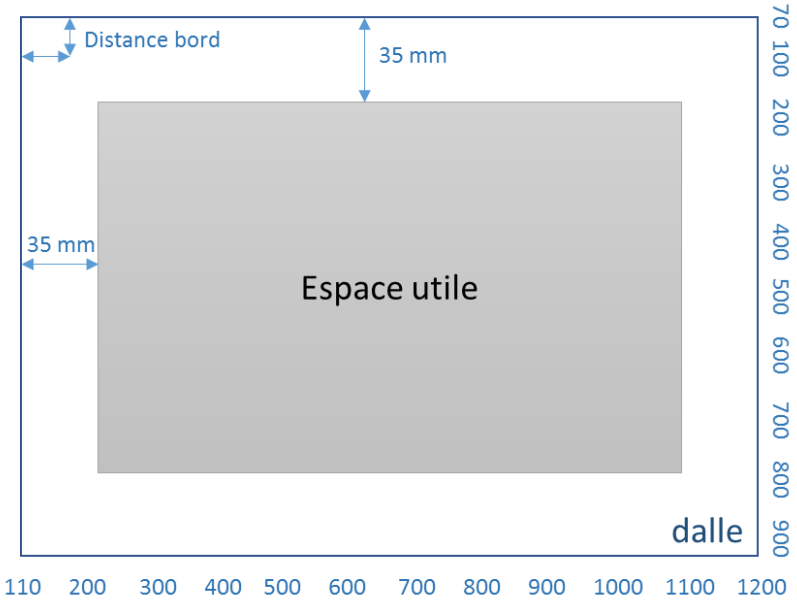


figure 36. Dalle caractérisé

5.2.2. Choix de bouton poussoir

Le bouton poussoir doit avoir un bout rond et rigide pour que le système soit plus précis. Il ne doit pas demander beaucoup de force pour ne pas casser la dalle ni l'antenne. Il doit être fixe selon les axes X et Y.

Je fixe le bouton poussoir à l'antenne dans le but de sécuriser l'antenne. Cette dernière ne doit pas s'approcher de plus de 5 millimètres du produit à tester.

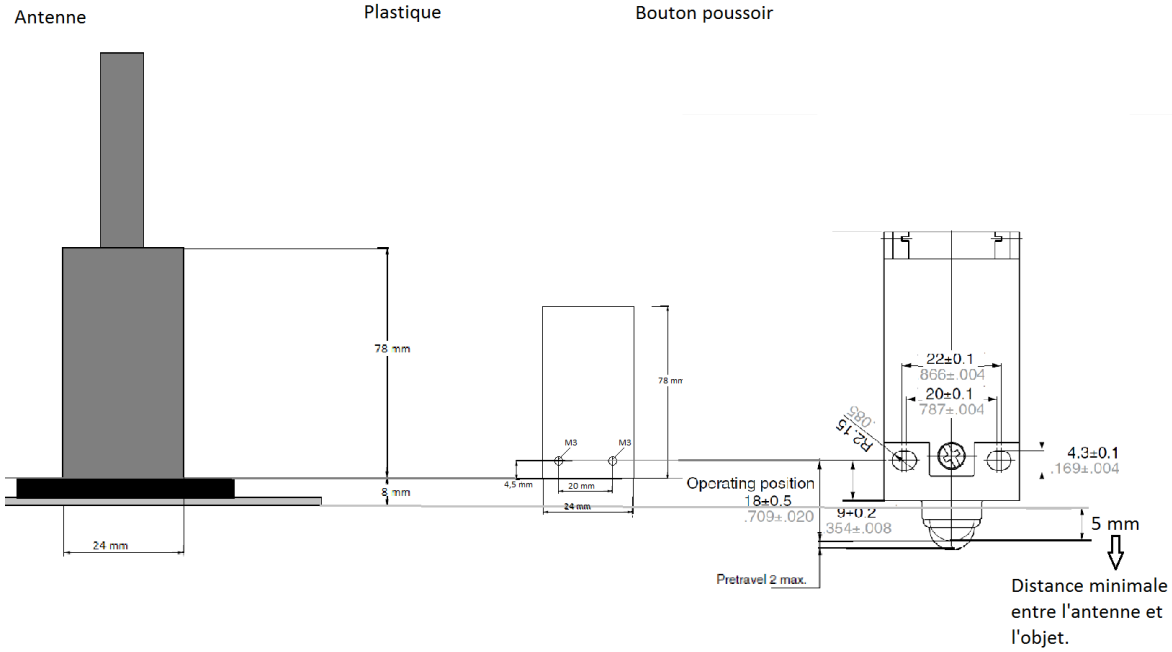


figure 37. Installation du bouton poussoir

5.3. Algorithme conçu pour l'auto calibration

L'algorithme que j'ai conçu pour l'auto calibration du robot CEM prend en compte toutes les particularités des capteurs choisis. J'ai programmé le logiciel pour calculer les facteurs X et Y de la dalle automatiquement.

La première partie de l'algorithme calcule le facteur X et Y et l'angle de rotation entre le repère *WORLD* du robot et le repère de la dalle. Les coordonnées du premier point où le robot a touché la dalle seront la référence pour cette première partie.

La deuxième partie de l'algorithme recherche les points du repère, en ayant le facteur X et Y fixes, ainsi que l'angle de rotation. Les coordonnées du dernier point touché sont utilisées comme référence pour la translation d'un repère par rapport à l'autre.

Pour trouver un point sur la dalle par rapport au repère du robot, j'ai calculé la matrice de rotation. En mathématiques, et plus précisément en algèbre linéaire, une matrice de rotation Q est une matrice orthogonale de déterminant 1, ce qui peut s'exprimer par les équations suivantes : $QtQ = I = QQt$ et $det\ Q = 1$, où Qt est la matrice transposée de Q, et I est la matrice identité.

$$Q = \begin{pmatrix} cos\,\theta & sin\,\theta \\ -sin\,\theta & cos\,\theta \end{pmatrix}$$

Expression de la matrice de rotation d'angle θ dans le plan muni d'un repère orthonormé direct.

Ces matrices sont exactement celles qui, dans un espace euclidien, représentent les isométries (vectorielles) directes. Ces dernières sont aussi appelées rotations vectorielles (d'où

le nom de *matrice de rotation*), parce qu'en dimension 2 et 3, elles correspondent respectivement aux rotations affines planes autour de l'origine et aux rotations affines dans l'espace autour d'un axe passant par l'origine.

La nouvelle fonction que j'ai conçue et ajoutée dans le logiciel *LabVIEW* pour faire l'auto calibration du robot CEM fait le processus suivant :

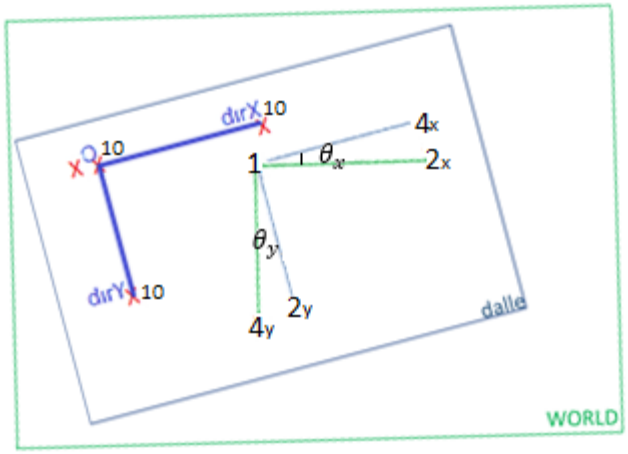


figure 38. Algorithme complet

1) Elle fait descendre le robot et récupère le point 1. $\left(\begin{bmatrix} X_{point1-dalle} \\ Y_{point1-dalle} \end{bmatrix}, \begin{bmatrix} X_{point1-robot} \\ Y_{point1-robot} \end{bmatrix}\right)$
2) Elle fait bouger le robot selon l'axe X du World, et calcule l'angle (θ_x^t) entre les repères et le facteur X (f_x^t).

$$\theta_x^t = \tan^{-1}\left(\frac{Y_{robot}^t - Y_{point1-robot}}{X_{robot}^t - X_{point1-robot}}\right) - \tan^{-1}\left(\frac{Y_{dalle}^t - Y_{point1-dalle}}{X_{dalle}^t - X_{point1-dalle}}\right)$$

$$f_x^t = \frac{\sqrt{(Y_{robot}^t - Y_{point1-robot})^2 + \left(X_{robot}^t - X_{point1-robot}\right)^2}}{\sqrt{(Y_{dalle}^t - Y_{point1-dalle})^2 + (X_{dalle}^t - X_{point1-dalle})^2}}$$

3) Elle calcule les coordonnées d'un point ayant la même valeur Y (dans le repère de la dalle) que celle du *point 1* puis projette le résultat dans le repère du robot. Pour être sûr que le facteur X est correct il faut que la différence entre les coordonnées Y des points soit zéro.

$$\begin{bmatrix} X_{robot}^{t+1} \\ Y_{robot}^{t+1} \end{bmatrix} = \begin{pmatrix} \cos\theta_x^t & \sin\theta_x^t \\ -\sin\theta_x^t & \cos\theta_x^t \end{pmatrix} * \left(\begin{bmatrix} X_{dalle}^{t+1} \\ Y_{point1-dalle} \end{bmatrix} - \begin{bmatrix} X_{point1-dalle} \\ Y_{point1-dalle} \end{bmatrix}\right) * f_x^t + \begin{bmatrix} X_{point1-robot} \\ Y_{point1-robot} \end{bmatrix}$$

4) Elle envoi le robot à ce point pour récupérer ces coordonnées $\left(\begin{bmatrix} X_{dalle}^{t+1} \\ Y_{dalle}^{t+1} \end{bmatrix}, \begin{bmatrix} X_{robot}^{t+1} \\ Y_{robot}^{t+1} \end{bmatrix}\right)$.
5) Elle calcule θ_x^{t+1} et f_x^{t+1} , et vérifie si Y_{dalle}^{t+1} est égal à $Y_{point1-dalle}$, dans le cas négatif le logiciel retourne à l'étape 3 en utilisant les paramètres calculés juste avant.
6) Elle sauvegarde le facteur X (f_x) et l'angle(θ_x).
7) Le logiciel répète le processus pour calculer le facteur Y (f_y) et l'angle (θ_y), mais cette fois-ci, il fait bouger le robot suivant l'axe Y, et cherche un point avec la même valeur X que celle du *point 1*.
8) Le logiciel sauvegarde les facteurs X (f_x) et Y (f_y) et la moyenne des angles rencontrés

$$\theta = \frac{\theta_x + \theta_y}{2}$$

9) Avec les paramètres sauvegardés, le logiciel calcule la position du point du repère dans le repère du robot :

$$\begin{bmatrix} X_{ptrep-robot} \\ Y_{ptrep-erobot} \end{bmatrix} = \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix} * \left(\begin{pmatrix} f_x & 0 \\ 0 & f_y \end{pmatrix} * \left(\begin{bmatrix} X_{ptrep-dalle} \\ Y_{ptrep-dalle} \end{bmatrix} - \begin{bmatrix} X^t_{point1-dalle} \\ Y^t_{point1-dalle} \end{bmatrix} \right) \right) + \begin{bmatrix} X^t_{point1-robot} \\ Y^t_{point1-robot} \end{bmatrix}$$

Les coordonnées $\begin{bmatrix} X_{ptrep-dalle} \\ Y_{ptrep-dalle} \end{bmatrix}$ sont constantes dans le programme développé, mais pour chaque point du repère on a des coordonnées différentes.

10) Le logiciel envoi le robot au point calculé et récupère les coordonnées $\left(\begin{bmatrix} X^{t+1}_{point1-dalle} \\ Y^{t+1}_{point1-dalle} \end{bmatrix}, \begin{bmatrix} X^{t+1}_{point1-robot} \\ Y^{t+1}_{point1-robot} \end{bmatrix} \right)$.

11) Elle vérifie si $\begin{bmatrix} X^{t+1}_{point1-dalle} \\ Y^{t+1}_{point1-dalle} \end{bmatrix} = \begin{bmatrix} X_{ptrep-dalle} \\ Y_{ptrep-dalle} \end{bmatrix}$, dans le cas contraire le logiciel retourne à l'étape 9 en utilisant le nouveau *point 1* touché.

12) Le logiciel défini le point du repère et répète le processus pour les autres points de ce dernier.

13) Après avoir calculé et retrouvé les trois points constituant le repère, le logiciel en crée un *repère utilisateur*.

14) Elle affiche un message à l'utilisateur pour confirmer la création du repère.

J'ai implémenté cette fonction dans le logiciel LabVIEW et ajouté l'option de l'auto calibration dans l'onglet *Robot Settings* du programme existant. Lorsque l'utilisateur appui sur cette option, le logiciel lui demande de placer la dalle dans la position choisie qui sera considérée comme étant le *repère utilisateur* et de placer le robot au-dessous de la dalle. Après confirmation du positionnement, la fonction d'auto calibration est lancée. Si une erreur se produit dans la génération du programme, un message d'erreur est affiché.

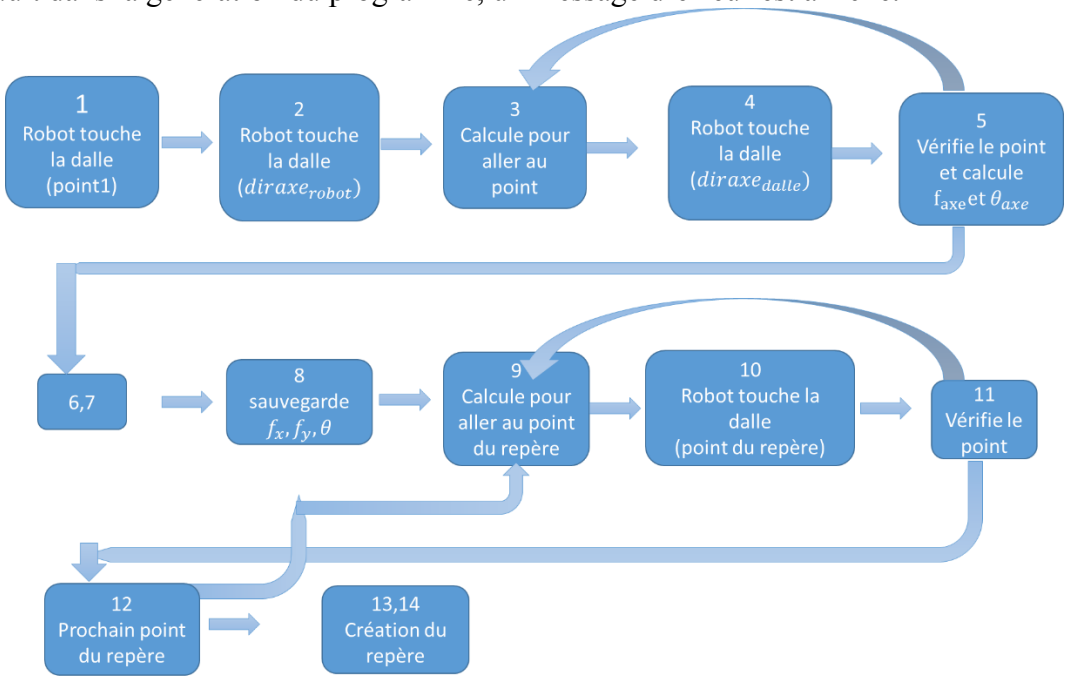


figure 39. Schéma de fonctionnement de la fonction

J'ai fait le test de cette fonction dans la chambre CEM, les résultats sont présentés dans la section *7.Tests dans la chambre des essais CEM.*

6. Réalisation deuxième partie : Implémentation d'un joystick

Cette partie du rapport présente le travail que j'ai fait pour l'implémentation d'un joystick. Dans un premier temps, je parlerai de l'essai programmé via LabVIEW, et j'expliquerai pourquoi cette solution n'a pas été retenue. Ensuite je présenterai le système que j'ai conçu et l'algorithme que j'ai développé. Les tests et les résultats seront présentés dans la section *7. Tests dans la chambre des essais CEM.*

6.1. Implémentation d'un joystick via LabVIEW

La première étape de l'implémentation du joystick était de le simuler sur LabVIEW et de mettre en place un programme dans le contrôleur du robot pour recevoir et traiter les données. La deuxième étape était de connecter un joystick à l'*ordinateur de pilotage* avec un port USB et récupérer ses données via LabVIEW.

La communication entre LabVIEW et le robot est faite par Ethernet. Pour le faire bouger par le joystick, il faut envoyer la valeur de celui-ci au contrôleur plus d'une fois par seconde, sinon l'utilisateur n'aura pas de control précis du robot.

Le programme du robot développé par *Autoliv* n'a pas été conçu pour recevoir plusieurs chaines de communication via Ethernet dans un écart de temps trop petit.

La communication entre le joystick et le robot doit être modifiée, pour cela, j'avais deux options :

- changer la communication Ethernet en optimisant le programme conçu par *Autoliv*
- mettre en place la communication sérial entre le joystick et le robot.

Connecter le robot directement au joystick était prévu dans le cahier des charges comme un extra, dépendant de l'avancement du stage. On a décidé alors que la meilleure option était de passer directement à l'étape suivante et ne pas continuer à développer sur LabVIEW.

6.2. Partie Hardware

Le temps restant pour la fin du stage n'était pas suffisant pour mettre en place un microcontrôleur complexe.

Arduino est un circuit imprimé en matériel libre sur lequel se trouve un microcontrôleur qui peut être programmé pour analyser et produire des signaux électriques. J'ai eu une expérience avec la carte *Arduino* avant ce stage. Je l'ai choisi pour sa simplicité à implémenter des boutons poussoirs, des potentiomètres et à faire la communication série.

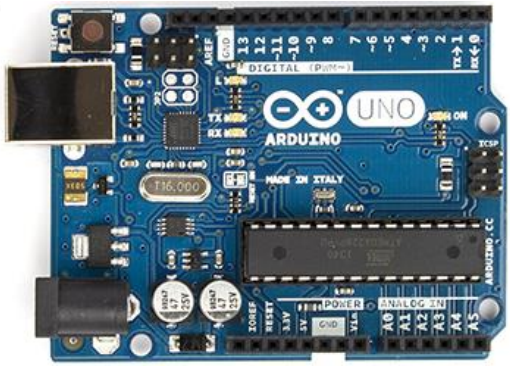


figure 40. Arduino UNO

Le contrôleur du robot *Adept* contient trois ports *RS-232* qui peuvent être configurés avec le programme *AdeptACE*. *Arduino* contient un port de communication USB pour une communication série TTL. Pour lier l'*Arduino* au contrôleur du robot j'ai besoin d'un composant (MAX232) qui transforme la communication TTL en RS-232.

Les joysticks commerciaux qui ont des liaisons USB sont compliqués à lier à l'*Arduino.* Pour simplifier l'implémentation du joystick j'en ai choisi un qui a trois potentiomètres. Le joystick que j'ai acheté est lié directement à l'*Arduino* vers les entrées analogiques.

figure 41. Joystick choisi

Dans le cahier des charges, on a prévu des options comme :

- Les mouvements du robot exclusifs selon chaque axe
- Le mouvement de rotation de l'antenne et translation du bras du robot
- Le contrôle par l'utilisateur de la vitesse maximale

J'ai ajouté une option qui permet à l'utilisateur de choisir le repère à utiliser : repère utilisateur ou repère WORLD. J'ai implémenté cette option ainsi que toutes les options prévues sur le cahier des charges.

Pour cela, j'ai acheté des boutons poussoirs équipé avec deux LEDs (un vert et un rouge), et un potentiomètre linéaire. Ce dernier est utilisé pour contrôler la vitesse maximale et le bouton poussoir est utilisé pour changer l'état de la LED qui indique l'option choisie.

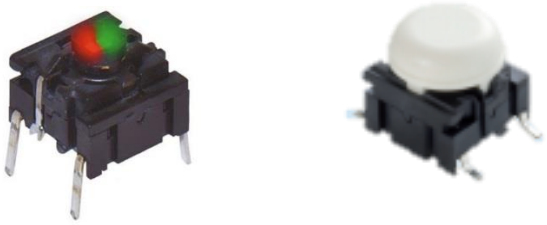


figure 42. Boutons poussoir équipé avec deux LEDs

Les LEDs sont connectées aux sorties numériques d'Arduino, les boutons poussoirs aux entrées numériques et le potentiomètre à une entrée analogique.

J'ai utilisé six boutons poussoir, un potentiomètre linéaire et un joystick « trois axes ». Chaque bouton correspond à un choix d'utilisation bien précis :

- Le démarrage du système joystick (ON/OFF)
- Le repère à utiliser
- Le mode : translation ou rotation
- Le mouvement en X
- Le mouvement en Y
- Le mouvement en Z

6.3. Partie software

L'objectif du joystick dans le banc de test *HANDY TRASMITTER* est d'ajouter un nouveau mode pour commander les mouvements du robot. Cet outil implémenté n'a pas besoin d'être lié au programme *LabVIEW*. J'ai conçu tout le software du joystick sur l'Arduino et sur le contrôleur du robot. Pour cela, j'ai établi un protocole de communication.

J'ai choisi de traiter les informations concernant la vitesse sur l'*Arduino* pour que la trame, envoyée par port série, soit plus compacte. Les options d'exclusivité du mouvement sur chaque axe ainsi que la configuration de la vitesse maximale sont traitées sur l'*Arduino*. Le contrôleur du robot gère les options de translation/rotation et l'option de changement du repère.

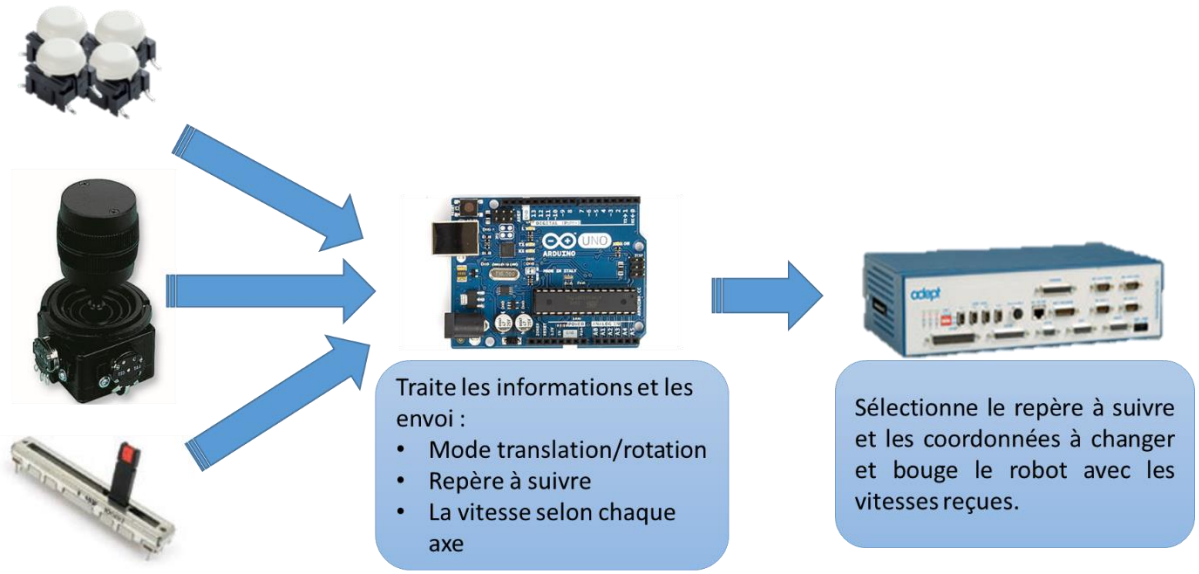
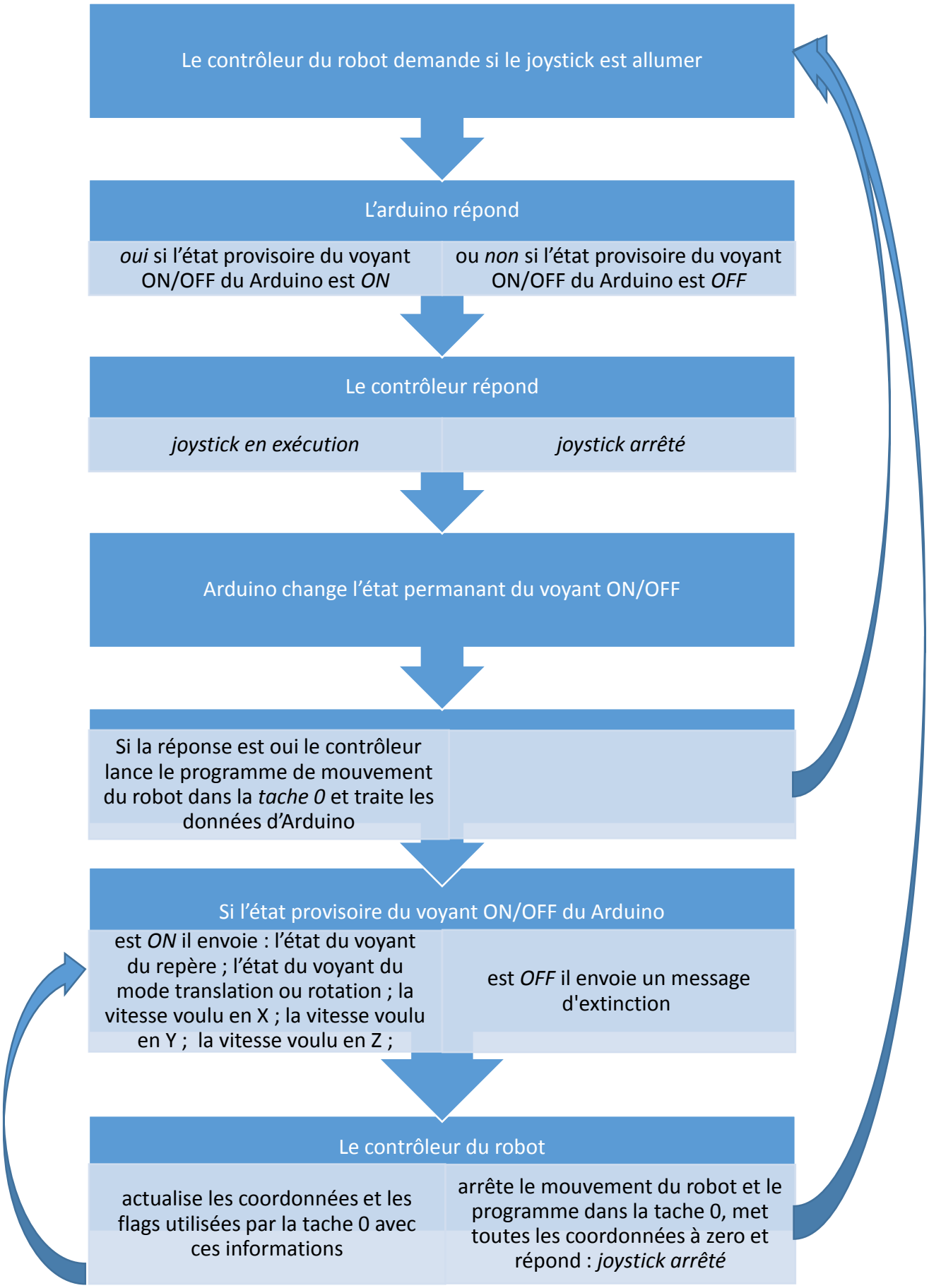


figure 43. Flux d'information du soft du joystick

Le contrôleur du robot établit la communication série dans la *tache 6*, au même temps qu'il établit la communication Ethernet dans la *tache 3*. Tous les mouvements du robot doivent être faits dans la *tache 0*. Ainsi, le robot ne peut pas avoir deux commandes de mouvements simultanément. L'utilisateur doit avoir l'option de désactiver le mouvement par rapport au joystick, pour cela j'ai ajouté le bouton *ON/OFF*.

Lorsque l'utilisateur connecte le joystick au contrôleur du robot et les allume, le système suit l'organigramme suivant :


Le contrôleur du robot demande si le joystick est allumer
L'arduino répond
oui si l'état provisoire du voyant ON/OFF du Arduino est ON
ou non si l'état provisoire du voyant ON/OFF du Arduino est OFF
Le contrôleur répond
joystick en exécution
joystick arrêté
Arduino change l'état permanant du voyant ON/OFF
Si la réponse est oui le contrôleur lance le programme de mouvement du robot dans la tache 0 et traite les données d'Arduino
Si l'état provisoire du voyant ON/OFF du Arduino
est ON il envoie : l'état du voyant du repère ; l'état du voyant du mode translation ou rotation ; la vitesse voulu en X ; la vitesse voulu en Y ; la vitesse voulu en Z ;
est OFF il envoie un message d'extinction
Le contrôleur du robot
actualise les coordonnées et les flags utilisées par la tache 0 avec ces informations
arrête le mouvement du robot et le programme dans la tache 0, met toutes les coordonnées à zero et répond : joystick arrêté

J’ai programmé une fonction qui permet à l’utilisateur de changer l’état provisoire du voyant *ON/OFF* en appuyant sur le bouton poussoir *ON/OFF*. L’état permanant du voyant ON/OFF correspond à la dernière réponse du contrôleur du robot. Quand ce dernier est différent de l’état provisoire, je fais clignoter la LED.

Le programme lancé dans la *tache 0* récupère la position actuelle du robot par rapport au repère *WORLD* ou au *repère utilisateur*, en dépendant du flag *repère*. Il ajoute aux coordonnées *x*, *y*, et *z* les valeurs *joys.x*, *joys.y*, *joys.z* respectivement si le flag *mode* est vrai, dans le cas contraire, ces valeurs sont ajoutées au *lacet, basculement* et *roulis* respectivement. Puis, le programme demande au robot de chercher cette nouvelle position. Cela se fait en boucle jusqu’à l’arrêt du joystick.

6.4.Conception du boitier

J’ai construit un boitier pour placer tous les boutons poussoirs et potentiomètres nécessaires pour l’implémentation du joystick. Ce boitier est petit, robuste et performant. Les utilisateurs droitiers et gauchers peuvent l’utiliser sans être gênés par les commandes, les voyants sont lisibles et il faut presser le bouton poussoir pour l’activer.

figure 44. Boitier conçu pour le joystick

Dans ce boitier, on retrouve la carte que j’ai conçue pour placer et lier l’Arduino aux boutons poussoirs et aux potentiomètres, ainsi que tous les composants nécessaires pour son bon fonctionnement.

7. Tests dans la chambre des essais CEM

7.1. Auto calibration

J'ai fait des tests pour valider la fonction d'auto calibration dans la chambre des essais CEM :

- Boîtier Volvo parallèle au plan de masse
- Boîtier Volvo parallèle au plan de masse et avec un angle de rotation par rapport au repère du robot (WORLD)
- Boîtier Volvo incliné par rapport au plan de masse

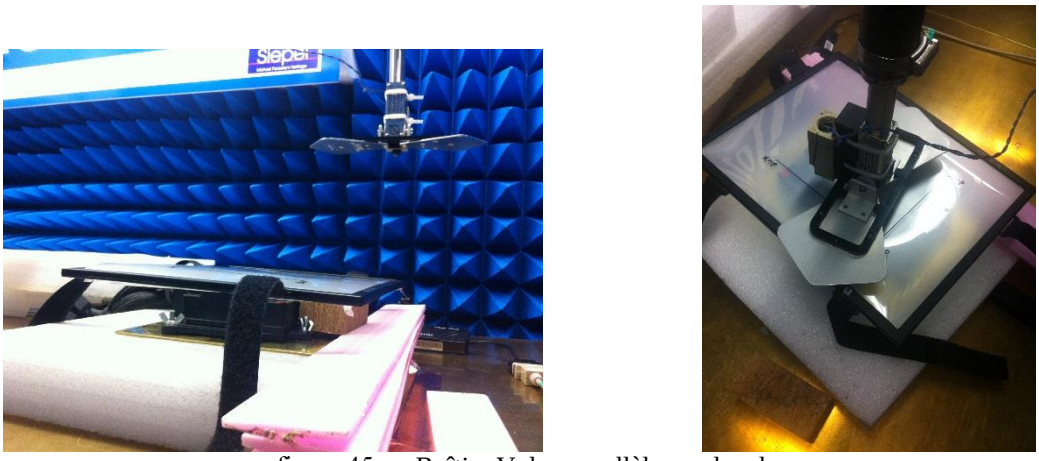


figure 45. Boîtier Volvo parallèle au plan de masse

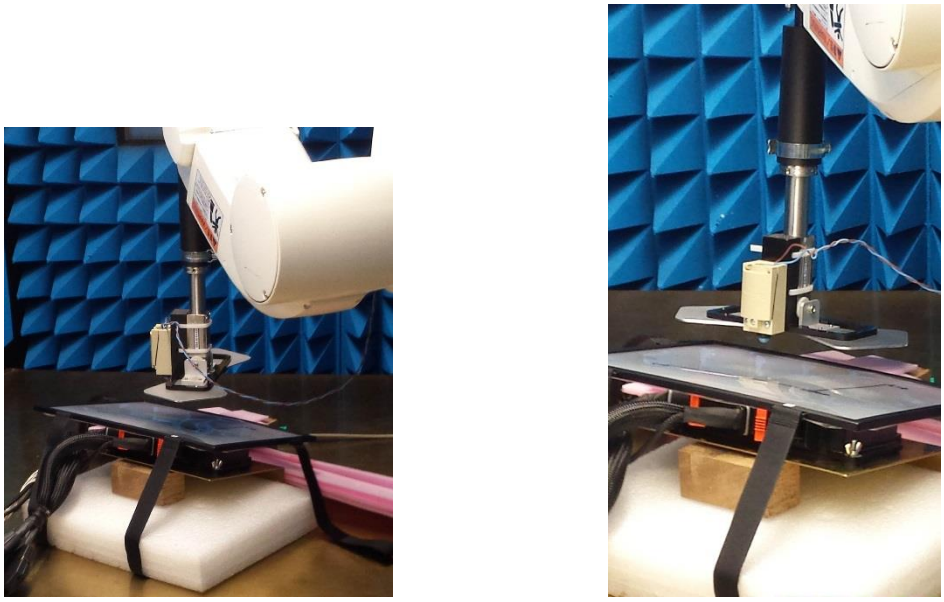


figure 46. Boîtier Volvo incliné par rapport au plan de masse

Pour tester si le repère a été bien créé, j'ai fait un fichier de positions correspondant aux autocollants rouges sur le contrôleur d'airbag. Avec le logiciel LabVIEW, j'ai téléchargé ce *fichier de positions* après chaque calibration automatique du robot.

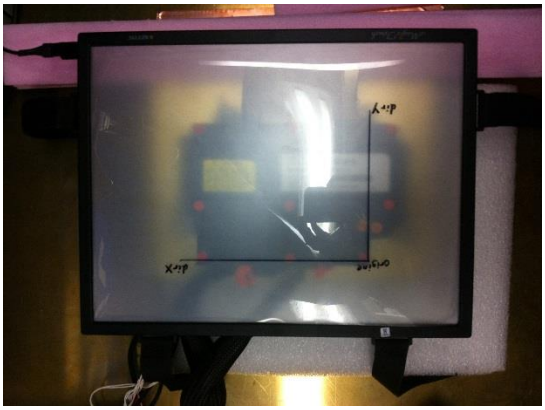


figure 47. Contrôleur Volvo avec la dalle et les autocollants

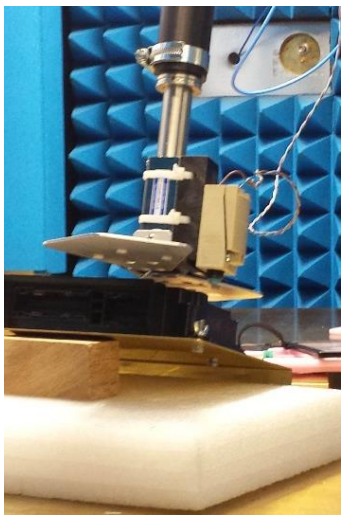


figure 48. Robot dans la première position du *fichier de position*

La position des points du *fichier de positions* est à une hauteur de deux millimètres près de la position voulue. Par conséquent, il faut prendre en compte l'épaisseur de la dalle. A l'œil nu, j'ai remarqué que l'antenne est bien positionnée par rapport aux autocollants, comme prévu dans le cahier de charges.

J'affiche les erreurs du robot à l'utilisateur pendant l'exécution du programme d'auto-calibration si :

- La télécommande est sélectionnée pour faire bouger le robot. Ainsi, elle empêche le mouvement automatique demandé par le programme LabVIEW.
- L'arrêt d'urgence est déclenché.
- Le robot n'est pas capable d'aller à une position demandée à cause de la limitation de ses articulations.
- La puissance du robot n'est pas mise.
- Le robot descend de dix centimètres et que le bouton poussoir n'est pas activé.

L'utilisateur peut annuler la création du repère à n'importe quel moment. Le programme affiche ce message :

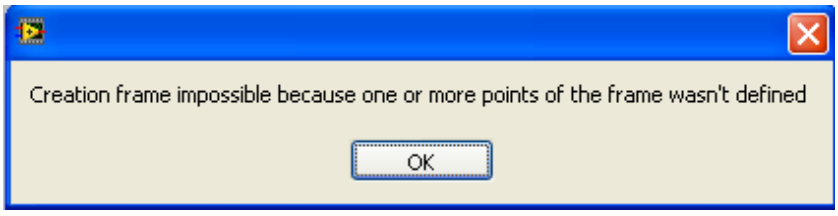


figure 49. Message d’erreur pour la création du repère

Le temps d’exécution de cette fonction est toujours entre quatre et dix minutes, ainsi l’utilisateur peut le faire tous les matins avant de commencer les essais CEM.

7.2.Joystick

Les tests de l’implémentation du joystick sont prévus pour la dernière semaine du stage. De ce fait, cette partie sera incluse dans la présentation finale et non dans le rapport.

8. Bilan personnel et technique

J'ai rempli tous les besoins du cahier des charges que j'ai fait au début du stage et j'ai pu rajouter certaines améliorations qui n'étaient pas prévues : la liaison série entre le joystick et le contrôleur du robot. Le joystick que j'ai développé est compatible avec l'autre robot du service. De plus, l'erreur de l'auto calibration du robot est négligeable pour faire le test d'émetteur portable.

Pour la suite du développement de cet outil voici mes suggestions :

- Ajouter une option qui récupère la position actuelle du robot pendant la création du *fichier de position* et pendant le test CEM.
- Etudier le logiciel du robot pour régler la position. En effet, la différence entre les valeurs de deux points n'est pas toujours égale au vrai déplacement du robot.
- Mettre en place les options de polarisation et de distance prévues dans l'onglet *Simulation,* du software LabVIEW.

Ce stage m'a donné une idée globale de ce qu'est un travail d'ingénieur. J'ai pu concevoir un système de software complexe qui utilise une interface homme machine et j'ai pu programmer un robot. J'ai eu l'occasion de faire un peu de hardware en développant une carte pour le joystick. Maintenant je me sens prête à franchir la prochaine étape dans ma vie professionnelle en tant qu'ingénieure en électronique. Grâce à ce stage, j'ai constaté que je suis capable de mettre en pratique toutes les études théoriques que j'ai acquises pendant mes cinq années d'ingénierie.

Pour mon stage, j'ai eu l'opportunité de travailler au sein d'une grande entreprise dans laquelle j'ai pu rencontrer des gens professionnels, compétents et sérieux qui m'ont accueilli chaleureusement. Ces personnes ont partagé avec moi leurs expériences professionnelles ce qui m'a rendu plus confiante pour le reste à venir.

Bibliographie

http://www.autoliv.com/AboutUs/Pages/CompanyHistory.aspx
Livret d'accueil
http://fr.wikipedia.org/wiki/Compatibilit%C3%A9_%C3%A9lectromagn%C3%A9tique
Utilisateur guides Adept
COFRAC
http://fr.wikipedia.org/wiki/Matrice_de_rotation
https://fr.wikipedia.org/wiki/Matrice_de_rotation
http://fr.rs-online.com/
http://fr.farnell.com/