MEHARZI DERRADJI SERIE AT2

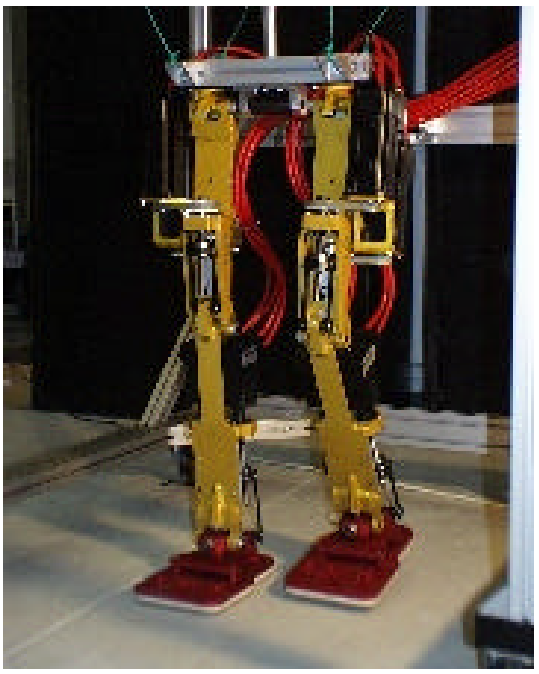
RAPPORT DE STAGE

# CAPTEURS DE PROXIMITE D'UN ROBOT BIPEDE

INRIA – IUT DE GRENOBLE

# Sommaire.

# 1 PRESENTATION DU STAGE



## 1.1 But du stage

L'enseignement apporté à l'I.U.T., qui a la particularité de reposer sur un suivi très proche des professeurs auprès les étudiants impose qu'une période de totale dépendance de l'élève existe. C'est pourquoi il est nécessaire que l'étudiant soit en complète autonomie et confrontée à une application pratique sur une période de plusieurs semaines :

- Il faut qu'il s'habitue à être dans un endroit tout à fait différent de l'environnement Universitaire, et s'intègre aussi dans une entreprise ;

- Il faut qu'il utilise les connaissances théoriques et pratiques acquises durant l'année universitaire à l'IUT dans un cadre professionnel de façon autonome ;

- L'encadrement du stagiaire doit aussi permettre d'améliorer ses connaissances en laissant libre court à sa liberté de créativité ;

- Enfin, l'entreprise où a lieu le stage peut offrir, quelques fois, un emploi pour les étudiants ne désirant pas poursuivre leurs études après le DUT ;

Toutes ces raisons font de ce stage, l'outil indispensable à l'étudiant de l'IUT, pour compléter sa formation.

## 1.2 Présentation de l'entreprise : l'INRIA.

**C**réé en 1967 à Rocquencourt près de Paris, l'INRIA (Institut national de recherche En informatique et en automatique) est un établissement public à caractère scientifique et technologique (EPST) placé sous la double tutelle du ministère de l'éducation nationale, de la recherche et de la technologie et du ministère de l'économie, des finances et de l'industrie.

Les principales missions de l'INRIA sont :

- entreprendre des recherches fondamentales et appliquées ;
- réaliser des systèmes expérimentaux ;
- organiser des échanges scientifiques internationaux ;
- assurer le transfert et la diffusion des connaissances et du savoir-faire ;
- contribuer à la valorisation des résultats de la recherche ;
- contribuer, notamment par la formation, à des programmes de coopération pour le développement ;
- effectuer des expertises scientifiques ;
- contribuer à la normalisation.

Avec le contrat d'objectifs signé avec l'état en janvier 1995, l'INRIA a confirmé son engagement à mettre l'excellence scientifique de ses chercheurs au service de son environnement national et international afin d'identifier les problèmes posés, de concevoir avec ses partenaires de meilleures solutions et de lancer rapidement ces dernières sur le marché.
Ainsi, l'INRIA est fermement engagé dans le transfert de technologie, soit par les partenariats noués avec des entreprises industrielles, soit par l'intermédiaire de ses sociétés de technologie.

Depuis 1967, l'INRIA a développé son implantation sur le territoire national. Il est désormais présent dans cinq régions :

- en Ile-de-France, une unité de recherche à Rocquencourt ainsi que le siège de l'institut également localisé à Rocquencourt.
- en Bretagne, une unité de recherche à Rennes (créée en 1980).
- en Provence-Alpes-Côte d'Azur, une unité de recherche à Sophia Antipoli (créée en 1982).
- en Lorraine, une unité de recherche à Nancy (créée en 1984).
- en Rhône-Alpes, une unité de recherche près de Grenoble (créée en 1992).

## ***1967-1972 : les premiers pas de l'INRIA***

A l'initiative de la Direction générale de la recherche scientifique et technique (DGRST), un groupe de dix personnalités indépendantes, choisies pour leur compétence scientifique ou économique, se réunissait, à la fin de l'année 1964 et courant 1965, sous la présidence du professeur Lelong pour conduire une réflexion sur l'importance des nouvelles techniques du traitement de l'information.

Le mot informatique n'avait pas encore cours ; il devait être proposé à quelque temps de là, mais le rapport que le groupe adressait en février 1966 au Comité consultatif de la recherche scientifique et technique (CCRST) mettait en lumière tous les concepts de base de cette nouvelle science et attirait l'attention sur l'influence décisive qu'elle allait jouer dans tous les domaines d'activités humaines.

Présentées par le CCRST, les propositions contenues dans le rapport des dix experts étaient adoptées dans leur principe, par un comité interministériel sur la recherche scientifique, présidé par G. Pompidou, alors Premier ministre, en 1966. Un peu plus tard, les instances politiques intéressées mettaient en œuvre l'ensemble des mesures connues sous le terme de « Plan Calcul ».

L'Institut de recherche d'informatique et d'automatique, IRIA, créé par le décret 67-722 du 25 août 1967, constituait l'un des organes principaux d'exécution de ce Plan Calcul. L'étendue de ses missions faisait participer l'INRIA à tous les aspects de l'opération. L'institut jouait, en effet, le rôle majeur, aussi bien pour la recherche que pour la formation des hommes et pour la diffusion de la connaissance scientifique et technique.

La période 1967-1972 a constitué une première phase dans la vie de l'INRIA, celle de la création ex-nihilo, l'époque des pionniers, sous la direction du professeur Michel Laudet, avec l'appui du professeur André Lichnerowicz, président du Conseil scientifique. Au cours de cette période, le domaine de Voluceau, laissé vacant par l'OTAN après avoir servi de cantonnement pour le personnel militaire américain, a été transformé en un instrument de recherche et de formation.

## ***1972-1980 : de l'IRIA à l'INRIA***

Le 25 février 1972, un conseil interministériel, sur proposition du Comité consultatif de la recherche scientifique et technique (CCRST), confirmait l'INRIA dans ses vocations et arrêtait une série de décisions propres à en amplifier les actions, notamment dans le domaine de la synthèse et orientation de la recherche française en informatique et dans le domaine de l'assistance technique donnée à la pénétration de l'informatique dans toutes les activités d'intérêt national :

- Regroupement de la recherche au sein d'un « Laboratoire de recherche d'informatique et d'automatique », le Laboria ;
- Développement des interventions ayant la nature d'une assistance technique à des applications exemplaires de l'informatique dans les différents secteurs d'activités d'intérêt national ;
- Rattachement direct à l'institut, qui en devient ordonnateur, des crédits d'irrigation de la recherche.

Les statuts de l'INRIA ont été partiellement modifiés suivant ces dispositions par le décret 73-338 du 13 mars 1973. La réforme de structure s'appuyait sur une pièce maîtresse, la création d'un Comité consultatif de la recherche en informatique, le CCRI, chargé d'élaborer les propositions d'une politique nationale de la recherche dans les domaines de l'informatique et de l'automatique. Le décret portant création du CCRI est le 73-130 du 12 février 1973.

Le 8 juin 1972, Michel Laudet, premier directeur de l'institut, passait ses pouvoirs à son successeur André Danzin.

A l'issue du Conseil d'administration du 28 juin 1972 :

- Michel Monpetit est appelé à la responsabilité de directeur adjoint de l'institut ;
- Le professeur Jacques-Louis Lions est nommé directeur du Laboria.

Parmi les missions auxiliaires de l'INRIA, intimement liées aux missions principales, on trouvait l'animation et la conduite de projets pilotes. Le premier étant la réalisation d'un réseau permettant l'interconnexion de plusieurs grands centres de calcul (Projet Cyclades).

Dans son action, l'INRIA se souciait non seulement d'accroître le potentiel en chercheurs, mais aussi d'orienter l'implantation des équipes, selon les préoccupations de la délégation à l'aménagement du territoire (DATAR). Il mettait ainsi à la disposition d'équipes de province, des postes budgétaires de chercheurs, notamment à Rennes. Ces équipes portaient le nom d'équipes associées de l'INRIA.

En 1975, à partir des équipes de Rennes, était créé l'Irisa, laboratoire de recherche associé à l'université de Rennes 1 et au CNRS.

Fin 1979, l'institut se recentre sur sa mission de recherche et de transfert et devient par décret du 27 décembre 1979, l'INRIA (Institut national de recherche en informatique et en automatique), établissement public à caractère administratif sous la tutelle du ministre de l'industrie.

Jacques-Louis Lions en devient le président directeur général.

Parallèlement est créée, par décret du 27 septembre 1979, l'Agence pour le développement de l'informatique qui est chargée, pour ce qui concerne les applications de l'informatique, de mener des expérimentations, des actions de sensibilisation, de formation et de prospection de nouveaux secteurs d'activité concernés. Elle est également chargée d'une mission d'animation et de développement de la recherche publique et privée sur les applications de l'informatique, sur les techniques susceptibles de favoriser leur développement. Cette agence sera dissoute en 1986.

***Après 1980 : l'essor de l'INRIA.***

Depuis les années 80, l'INRIA n'a cessé de se développer :

En 1980, l'unité de recherche de Rennes voit le jour en tant qu'une des composantes de l'Irisa.
En 1982, est créée l'unité de recherche de Sophia Antipolis.
En 1983, Alain Bensoussan devient président directeur général de l'INRIA.
En 1984, est créée l'unité de recherche de Lorraine, conjointement avec l'université de Nancy et le Centre de recherche en informatique de Nancy (Crin) du CNRS.
En 1984, Simulog voit le jour. Il s'agit de la première filiale industrielle de l'INRIA, dans le domaine de l'ingénierie assistée par ordinateur. C'est la première d'une série qui conduira à la création de 37 autres sociétés de technologie issues de l'INRIA.
En 1985, l'INRIA devient établissement public à caractère scientifique et technologique (EPST) placé sous la double tutelle du ministère chargé de la recherche et du ministère chargé de l'industrie.
Le décret 85-831 du 2 août 1985, portant sur l'organisation et le fonctionnement de l'institut, rappelle ses principales missions :

- Entreprendre des recherches fondamentales et appliquées ;
- Réaliser des systèmes expérimentaux ;
- Organiser des échanges scientifiques internationaux ;
- Assurer le transfert et la diffusion des connaissances et du savoir-faire ;
- Contribuer à la valorisation des résultats des recherches ;
- Contribuer, notamment par la formation, à des programmes de coopération pour le développement ;
- Effectuer des expertises scientifiques ;
- Contribuer à la normalisation.

En 1987, l'INRIA célèbre son 20e anniversaire : Alain Bensoussan déclare que la plus grande spécificité de l'institut consiste à obtenir que les résultats de la recherche soient transférés aussi vite que possible dans le secteur économique.
En 1987, la deuxième filiale de l'INRIA, Ilog, chargée de l'industrialisation de produits INRIA en intelligence logicielle, voit le jour.
En 1989, l'INRIA est un des fondateurs du consortium européen Ercim qui regroupe actuellement quatorze organismes de recherche.
En 1990, la troisième filiale industrielle de l'INRIA, O2 Technology, est créée dans le domaine des systèmes de gestion de bases de données orientées objet.
En 1992, l'unité de recherche de Rhône-Alpes voit le jour.
En mars 1994, l'INRIA adopte un plan stratégique visant à définir les principaux axes de la politique générale menée par l'institut.
Le 31 janvier 1995, l'INRIA est le premier organisme de recherche à signer avec le ministère de l'enseignement supérieur et de la recherche un contrat d'objectifs. Ce contrat associe également le ministère de l'industrie, des postes et des télécommunications, autre tutelle de l'institut.
En 1995, l'INRIA est choisi comme pilote européen du consortium W3C, créé en 1994, aux côtés du MIT pour les Etats-Unis.
En 1996, Bernard Larrouturou devient président directeur général de l'INRIA. En décembre, il fait adopter par le Conseil d'administration les principales orientations de la politique de l'institut pour les prochaines années.
En décembre 1997, un essai prospectif rédigé par Bernard Larrouturou en juillet 1997 et intitulé « INRIA 2007, l'INRIA dans dix ans » est diffusé aux partenaires de l'institut.

## 1.3 Les moyens robotiques de l'INRIA Rhône-Alpes.

Le rôle des moyens robotiques est la mise en œuvre et la maintenance de plate-forme robotiques pour réaliser les expérimentations des projets robotiques et vision.
Il a pour ambition de fonctionner à l'instar d'un service informatique et travailler en étroite collaboration pour les projets robotique et vision (BIP, MOVI, SHARP) de l'INRIA Rhone-Alpes.

Les missions qui lui sont attribuées sont de trois types :

Activité de service :

- maintenance des systèmes robotiques ;
- installation et maintenance de logiciels spécialisés ;
- interface entre les utilisateurs et le service informatique ;
- assistance aux utilisateurs .

Activité de développement :

- mise en place d'expérimentations ;
- développement de logiciels dédiés à la robotique.

Activité de recherche :

- conception de systèmes robotiques ;
- confrontation théorie et expérimentation .

**Le but du Service Robotique est de fédérer l'effort expérimental en favorisant :**

- les expérimentations inter-projets ;
- la mise en commun des moyens expérimentaux ;
- les outils réutilisables (environnement de développement, machine de vision...).

## 1.4 Le sujet du projet : L'étude des capteurs d'un robot autonome.

### Projet Bip

Ce projet, créé au 1er janvier 1994, a pour objectif la conception de robots marcheurs de type bipède, et plus spécifiquement la mise au point de leur commande. L'intérêt de tels robots réside dans leur capacité naturelle à évoluer dans les environnements de notre vie quotidienne, essentiellement conçus pour la bipédie. Ainsi, la classe d'application visée en priorité est la robotique de service, même si des retombées dans d'autres secteurs, comme la biomécanique, sont espérées.

Axes de recherche :

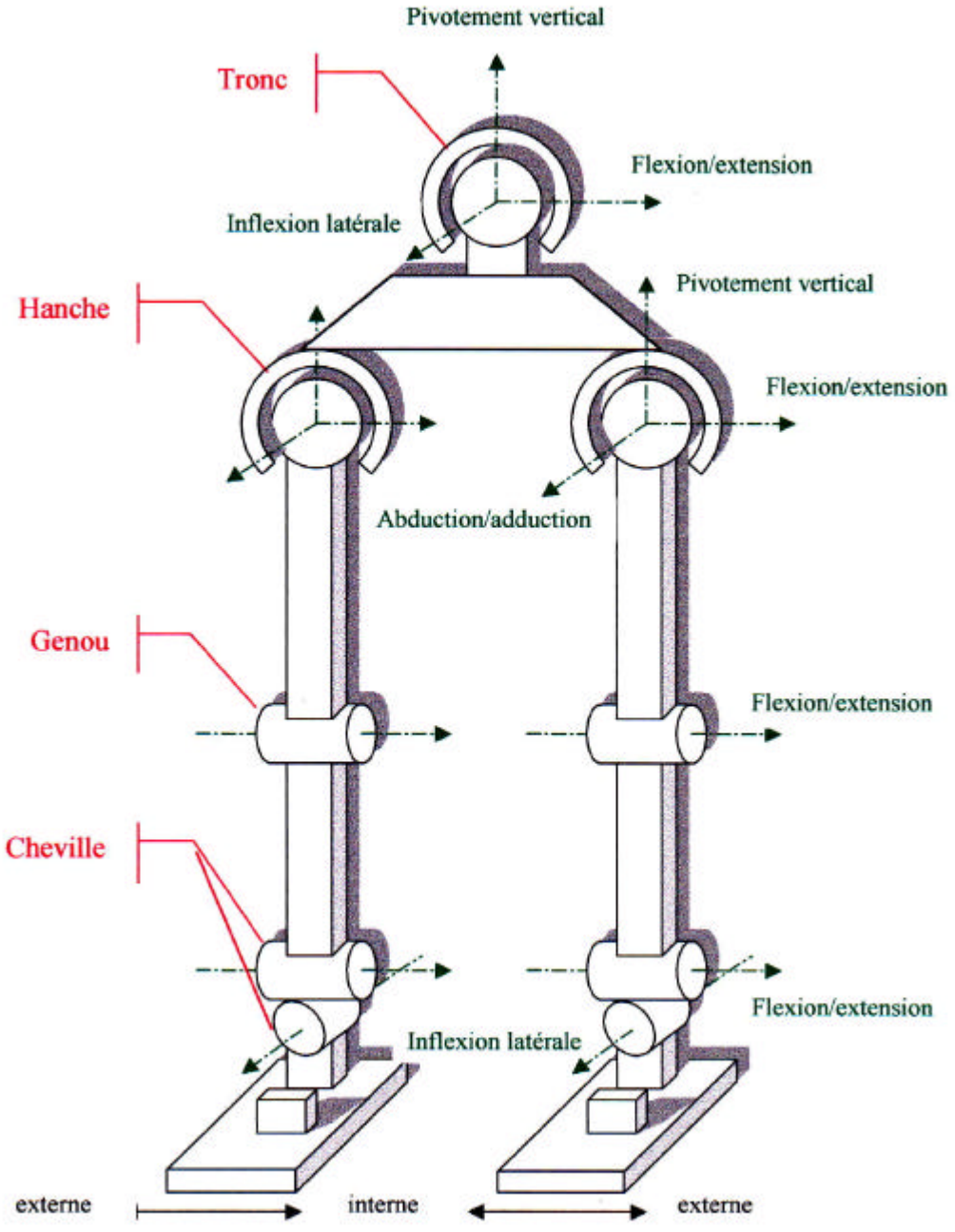
- Modélisation de la marche humaine dans diverses configurations : mesures de paramètres sur des ensembles de sujets, recherche d'invariants posturaux ou de mouvement, détermination des critères d'optimisation sous-jacents ;
- Etude de schémas de commande basés sur la stabilisation active de régimes passifs quasi-périodiques en conjonction avec des techniques de tâches redondantes et de commande référencée capteurs (force, proximité ou vision) ;
- Mise au point d'outils de conception/programmation/vérification pour l'ensemble du contrôle/commande à partir d'une approche synchrone ;
- Conception de sous-ensembles mécaniques particuliers ;
- Animation et synthèse de la marche.

Relations internationales et industrielles :

- Collaboration sous forme de projets ou contrats communs avec des biomécaniciens (UFR Staps de Grenoble, Groupe d'Analyse du Mouvement de Dijon), des automaticiens (Laboratoire dAutomatique de Grenoble) et des mécaniciens (Laboratoire de Mécanique des Solides de Poitiers) ;
- Contrats en cours dans le cadre du projet Via ;
- (GRD-PRC « Communication Homme – Machine ») et du programme Praxitèle ;
- Contrats en préparation : Salomon et Région ;
- Participation aux comités de programme Icar, Icra, Iser, Syroco ;
- Activités d'éditeur associé pour la revue IEEE Control Systems Technology.

**Un aperçu du robot Bipède :**

REPRESENTATION DES MOUVEMENTS ET ARTICULATIONS




Les axes représentés correspondent aux axes de rotation des mouvements indiqués

**La position du stage vis à vis du projet :**

L'implantation des capteurs sur le robot nécessite une parfaite connaissance des caractéristiques de ceux-ci. C'est pourquoi il est nécessaire de réaliser toutes les mesures possibles et ceci dans des conditions proches de l'utilisation sur le robot.
Cette caractérisation va permettre d'aider au placement et au choix des capteurs. Ces capteurs devront aider à conférer une certaine autonomie au robot.
Etant donné l'importance des capteurs dans le robot bipède, il va falloir faire l'étude de ces capteurs en plusieurs parties, ceci pour améliorer la performance de ces capteurs.
Ces parties sont:

- Exécuter quelques essais préliminaires pour déterminer les fonctionnalités des capteurs, leurs plage de détection, leur précision, …
- Ensuite, il faudra déterminer s'il existe des problèmes avec les capteurs.
- Une fois ces problèmes détectés – s'ils existent - il faut trouver un moyen soit logiciel soit matériel pour extraire les données utiles.
- Enfin procéder à des tests en se plaçant dans le cas le plus proche de la réalité du robot.

# 2 ETUDE THEORIQUE DES CAPTEURS

## 2.1 L'importance des capteurs dans un système tel que le robot bipède.

**L'intérêt des capteurs dans un système automatisé :**

Dans un système automatisé tel que le robot bipède, il est nécessaire de connaître en temps réel toutes les caractéristiques liées à la fois a la nature de l'environnement (état des surfaces, inclinaison du sol, présence ou non d'obstacles, …) et à celle du robot (coordonnées, position de tous les segments, inclinaison par rapport au sol, accélération, …).
En effet pour contrôler dans l'espace tous les segments du robot il faut avoir une information en temps réel de tous les événements externes et propres au robot.
On comprend vite l'utilité de ces capteurs. Ces capteurs vont être capable de fournir dans des temps assez courts des informations très importantes sur « l'état des lieux », et l'état du robot.

Ces capteurs placés sur le robot seront de deux natures différentes. Comme nous l'avons expliqué, soit ils informent le robot sur la nature de l'environnement, alors ce seront des capteurs ***extéroceptifs***, soit ils informeront le robot sur son état interne, alors ce seront des capteurs ***proprioceptifs***.

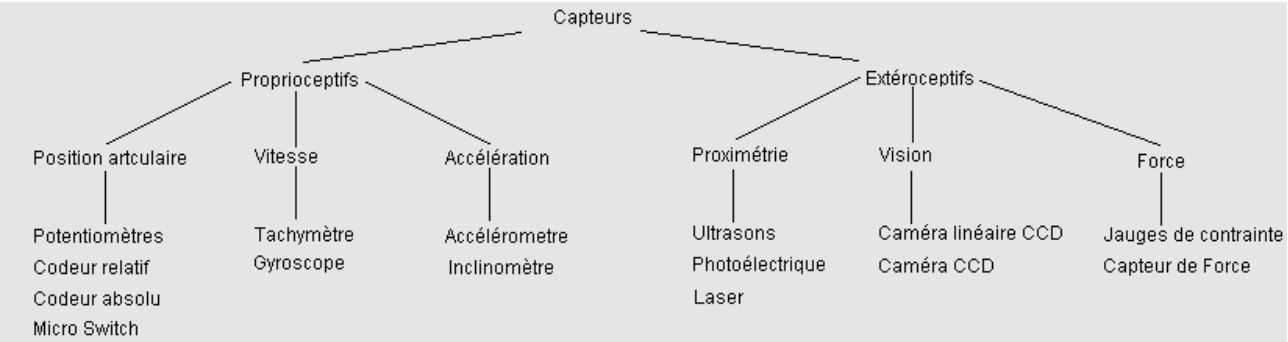
Il est facile de trouver des capteurs qui fournissent les informations requises, mais encore faut il faire le bon choix, savoir si l'information apportée est nécessaire, et placer judicieusement ces capteurs sur le robot.

En effet ces capteurs ne doivent pas gêner les mouvements du robot. On leur préférera donc un emplacement sur les articulations, là où aucun segment ne peut rentrer en collision avec les capteurs. De plus, il ne doit pas y avoir de conflit entre plusieurs capteurs. C'est à dire que la mesure d'une variable ne doit pas influer sur la mesure d'une autre variable. Ceci sera un problème lorsque plusieurs capteurs se trouveront sur le même segment.

D'autres problèmes se posent aussi avec les capteurs. En effet pour pouvoir traiter l'information apportée par le capteur, il faut connaître les relations entre la valeur donnée (tension ou courant), et la grandeur physique à mesurer. C'est pour cela qu'il est nécessaire de réaliser différents étalonnages et essais afin d'optimiser les acquisitions. Ces essais se diront sur table, à l'aide d'appareils de mesure, sans aucun outil informatique.

## 2.2 Présentation des types de capteurs existants.

L'arborescence suivante montre d'une manière générale , les capteurs existants ainsi que leur classifications :




Le tableau suivant indique le constructeur et la référence de tous les capteurs étudiés :

| **Type** | **Constructeur** | **Référence** |
|---|---|---|
| Micro – interrupteur | RadioSpare | 227 BTI |
| Potentiomètres | Megatron | Ma-850 |
| Ultrasons | Sensorex | UM 30-3000 A-HP |
| Photoélectrique miniature | Matsushita | UZ-B1601 |
| Photoélectrique moyenne portée | Matsushita | AKM17165 |
| Inclinomètre | Sensorex | 42745 |
| Gyroscope | Murata | ENV-05 X |

### 2.2.1 Les capteurs extéroceptifs :

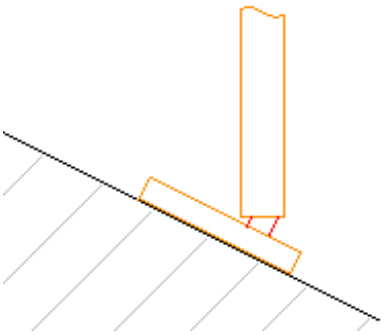
Ces capteurs fournissant des informations relatives à l'environnement, il faut connaître exactement leur caractéristiques pour pouvoir interpréter les informations reçues et les décoder.
De plus il faut savoir quand l'information reçue est valable ou pas(plage d'utilisation) .
Il est tout à fait possible de s'apercevoir lorsqu'une information est fausse, quand le système automatisé est muni de plusieurs types de capteurs qui mesurent une même grandeur physique.
Dans le cas du robot nous verrons que certaines informations sensorielles sont redondantes mais délivrées par plusieurs capteurs.

Il existe plusieurs types de capteurs extéroceptifs utilisés par le robot :

- Des capteurs donnant l'inclinaison par rapport à la verticale (***inclinomètre***) ;
- Des capteurs informant sur la nature du sol et la présence ou non d'obstacles (***capteur photoélectriques, à ultrasons***) ;
- D'autres qui indiquent la vitesse de rotation (***gyroscopes***).

a L'inclinomètre.

Ce capteur, informe l'opérateur sur l'angle qui existe entre la verticale du robot et l'axe de la force de gravité terrestre. Il sera placé sur le pied du robot et sera réservé à la sécurité et à l'étalonnage des capteurs absolus de position (cf. §2.2.2.a). En effet, ce capteur n'ayant qu'une très faible bande passante – de l'ordre de la dizaine de hertz - la commande du robot ne pourra pas se faire à l'aide de celui-ci.
Lorsque le sol n'est pas tout à fait perpendiculaire à la verticale terrestre, le robot peut rapidement se trouver en déséquilibre. Quand le robot a un pied posé à terre, il faut absolument redresser le robot pour qu'il retrouve son équilibre. C'est donc dans ce cas que l'inclinomètre se révèle essentiel. Grâce au croquis suivant, nous pouvons voir dans quel cas, ce capteur est vraiment utile :



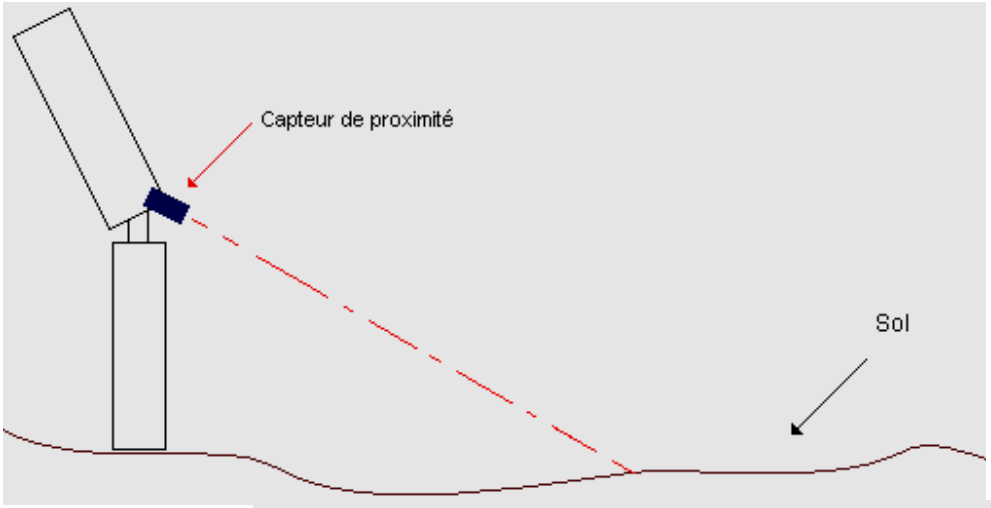
Sur un sol incliné comme celui – ci, le robot est constamment en mouvement pour ne pas perdre l'équilibre.
La photo suivante permet de se faire une idée plus précise de l'endroit où sera placé le capteur.

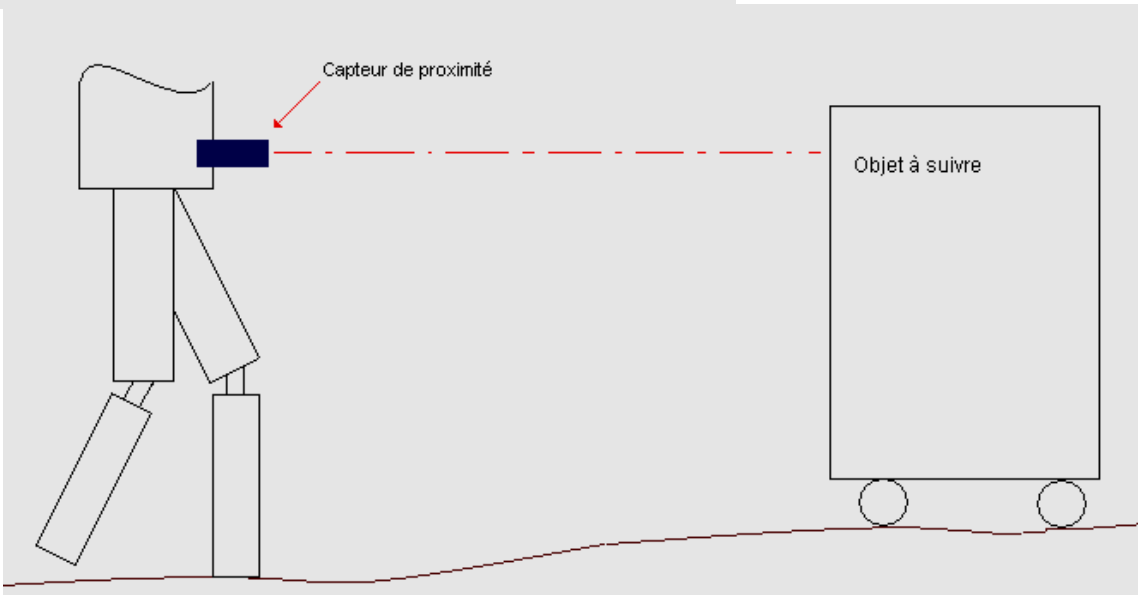
## b Les capteurs de proximité

Ces capteurs seront les outils qui serviront à déterminer les éventuels obstacles que rencontreraient le robot.
Il est prévu de placer ces capteur sur deux endroits différents :

- si l'on veut faire du suivi d'objet (*tracking)*, il faudra placer ces capteurs au niveau du pubis.
- par contre si l'on veut reconstruire l'environnement du robot, il faudra placer ces capteurs au niveau du genou.

Les croquis suivants montrent les deux types de montage.



### c Le gyroscope.

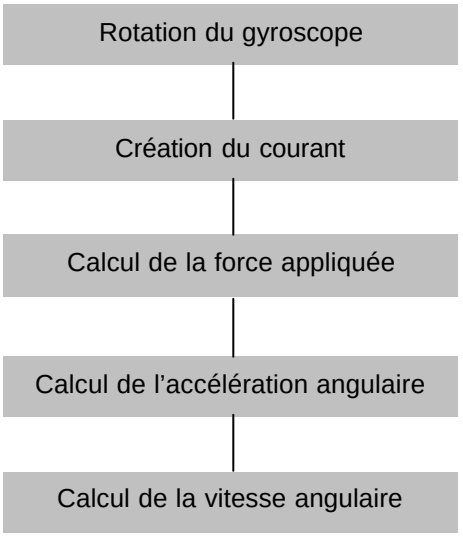
Ce capteur, qui fait partie de la famille des capteurs proprioceptifs est un capteur qui permet de déterminer la vitesse angulaire d'un objet.
Il fonctionne sur la base de l'effet piézo-électrique. En quelques mots, on peut dire que cet effet produit des charges électriques lorsqu'il est soumis à une force ($F = M \gamma$) à partir d'un choc d'une masse. Dans le cas du gyroscope, on fournit un courant qui permet d'annuler les charges crées par la contrainte. C'est la valeur du courant fournit qui va déterminer la force qui a été appliquée. Connaissant la valeur de la masse, on peut définir l'accélération qui en résulte par le calcul suivant :

$$\gamma = F / M$$

Où :

- F est la force appliquée sur la masse. Elle s'exprime en Newtons.
- M est la masse. Elle est exprimée en grammes.
- $\gamma$ représente l'accélération du mouvement du gyroscope. Cette accélération est donnée en mètre/secondes $^2$.

Une fois l'accélération calculée. Il est facile de retrouver la vitesse en faisant une simple dérivée.
On peut représenter le fonctionnement du gyroscope par le synoptique suivant :



### 2.2.2 Les capteurs proprioceptifs :

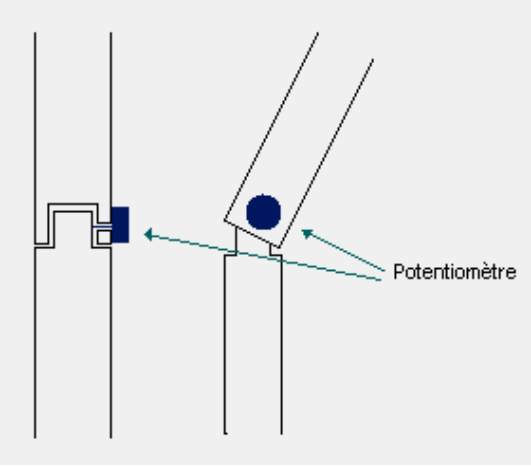
Ces capteurs fournissent une information interne au robot. Ils sont souvent placés sur les articulations pour mesurer les variations angulaires.

Comme pour les capteurs extéroceptifs, il existe plusieurs types :

- Certains fournissent une information relative à la position entre chaque segment (potentiomètres rotatifs mono-tour) ;
- D'autres signalent les position extrêmes des articulations, on les appelle butées fr fin de course.

#### a Les potentiomètres rotatifs mono – tour :

La valeur de ces potentiomètres est de **5 KOhms à +/- 10 %.** Ces potentiomètres d'une bonne résolution vont pouvoir donner l'angle formé entre deux axes du robot. Ce sont des potentiomètres spéciaux car dédiés à cet usage.
Ces potentiomètres vont fournir dés le démarrage la position angulaire de l'articulation. on dit qu'il s'agit de capteur de position absolue par opposition aux codeurs optiques (installés sur toutes les articulations de notre robot) qui fournissent une position relative mais avec une très grande précision : **$10^{-5}$ degré**.




Exemple :
Potentiomètres sur genou

Des potentiomètres sont aussi installés sur les chevilles pour mesurer la position absolue du pied par rapport à l'axe du « tibia ». On peut voir leur montage sur la photo suivante :

Ce type de montage va nous permettre de déterminer le profil du sol rencontré par le robot.

Dans tous les cas, on peut dire que ces capteurs sont complémentaires aux mécanismes moteurs + résolveurs. En effet l'ensemble moteur + résolveur informe l'opérateur sur la position relative des axes, ce sont ces variations relatives qui vont être utilisés pour le contrôle et l'asservissement des mouvements du robot.

b Les micro interrupteurs.

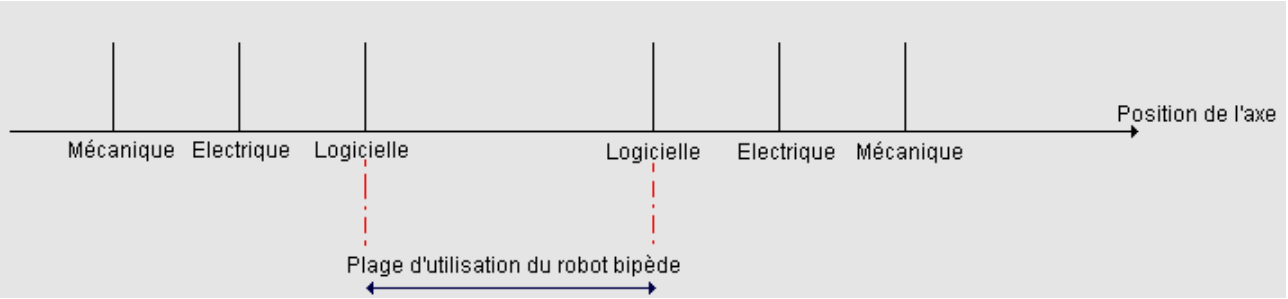
Ils sont placés sur chaque articulation afin de détecter les positions articulaires minimales et maximales avant les butées mécaniques.

Tous ces capteurs sont câblés en série sur chaque jambe du robot.

Le robot bipède dispose de trois sécurités au niveau articulaire :

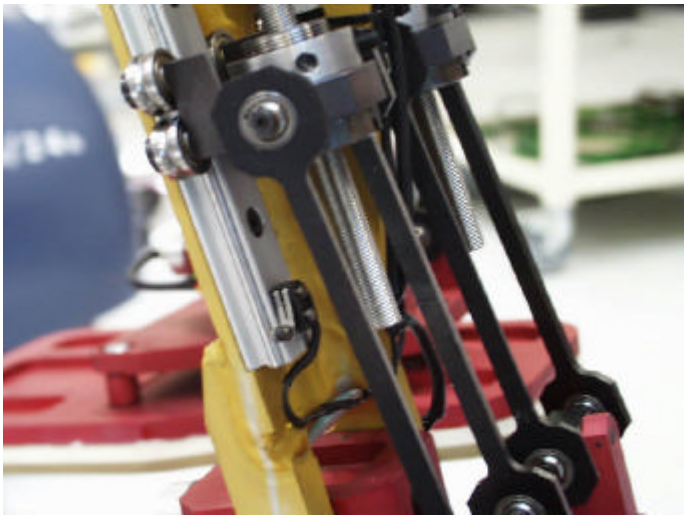
- Butée logicielle ;
- Butée électrique ;
- Butée mécanique ;

Le croquis suivant montre la plage d'utilisation normale du robot bipède.




Lorsque l'on utilise le robot dans sa plage normale, aucune sécurité n'intervient. Par contre dés que l'angle formé par deux segments articulés devient soit trop grand, soit trop petit, l'opérateur est averti par le logiciel d'un état d'urgence. Si l'opérateur n'en tient pas compte et que le robot continue toujours son mouvement, le système doit s'arrêter. En effet, les micro interrupteurs doivent déclencher un arrêt du robot via le système de sécurité « hard ».
Les articulations ne devraient jamais atteindre leur limite mécanique..

On peut voir sur la photo suivante, un exemple de montage de ces capteurs :

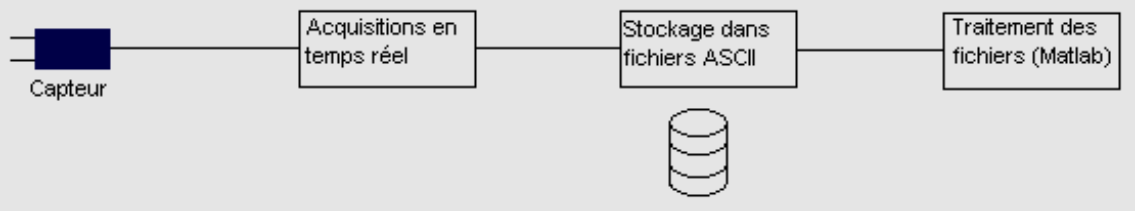


# 3 ETUDE PRATIQUE DES CAPTEURS

Cette étude portera sur une partie des capteurs qui seront intégrés sur le robot. Sont concernés, les capteurs de proximité (ultrasons et photoélectriques), les capteurs de position articulaire (potentiomètres), et l'inclinomètre.
cette étude sera basée en trois parties :

- Une première approche des capteurs ;
- Une phase d'acquisition dans des conditions proches de celles du robot ;
- L'étude de ces acquisitions à l'aide d'outils mathématiques ;

Le déroulement des deux dernières opérations se fera comme suit :



## 3.1 Première approche des capteurs de proximité utilisés dans le projet.

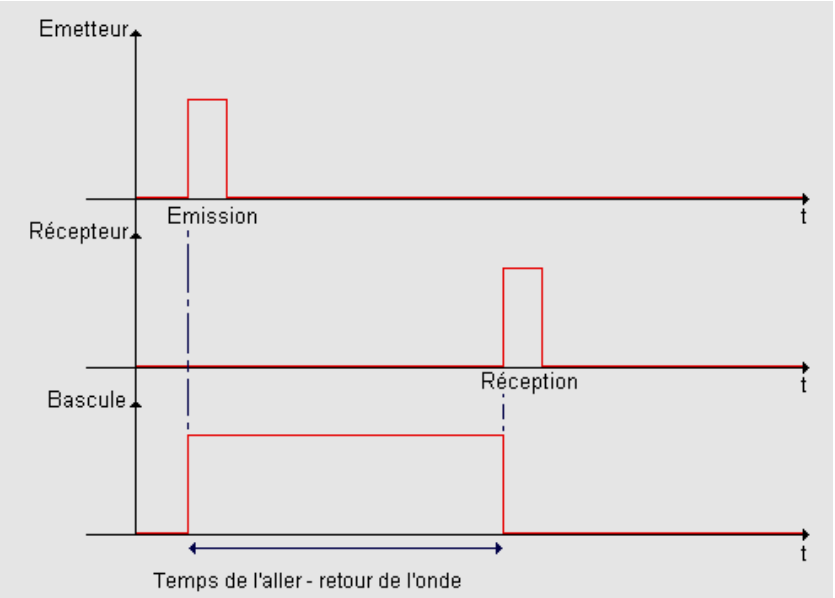
### 3.1.1 Le capteur à ultrasons.

(réf. UM 30-3000 A-HP  Constructeur : Sensorex) :

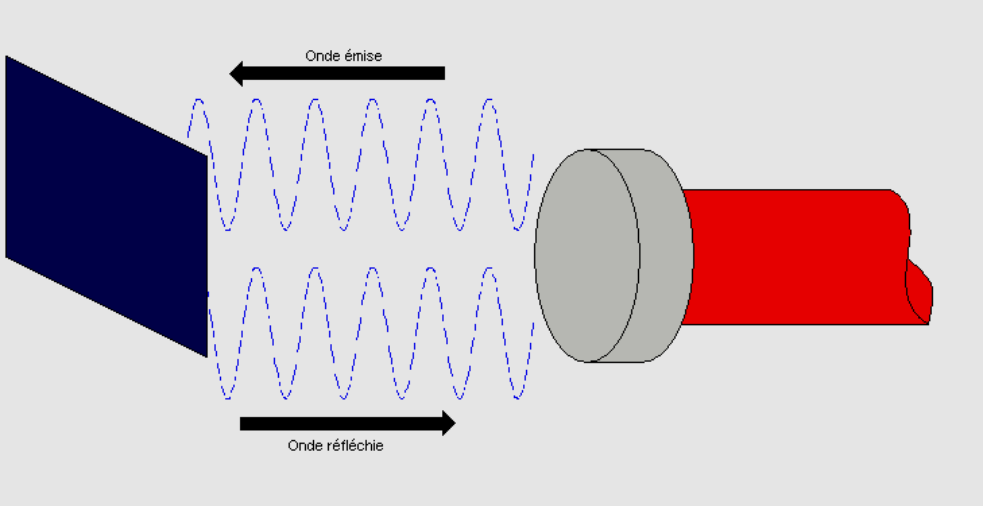
Ce capteur intègre un micro-contrôleur qui calcule grâce au temps de vol de l'onde émise, la distance qui le sépare de la cible. On peut représenter le fonctionnement du capteur par le synoptique suivant :


On comprend bien que durant l'émission du train d'ondes, le capteur ne puisse recevoir une autre onde. C'est cela qui va empêcher le capteur de ne pas mesurer les courtes distances.
Cette zone morte correspond au temps d'inhibition, et se retrouve aisément avec le calcul suivant :
L = (V x T) / 2
Où L représente la distance correspondant à la zone morte. Elle s'exprime en mètres ;
V, est la vitesse du son en mètres/secondes ;
T, est la durée d'inhibition en secondes. Elle est liée à la durée d'émission de l'onde.

Grâce aux chronogrammes suivants, il devient plus facile de comprendre la méthode employée pour calculer la distance qui sépare le capteur de la cible :




On peut voir avec le croquis ci-dessous la forme des signaux envoyés et des signaux réfléchis. On admettra que ces signaux sont des sinusoïdes.

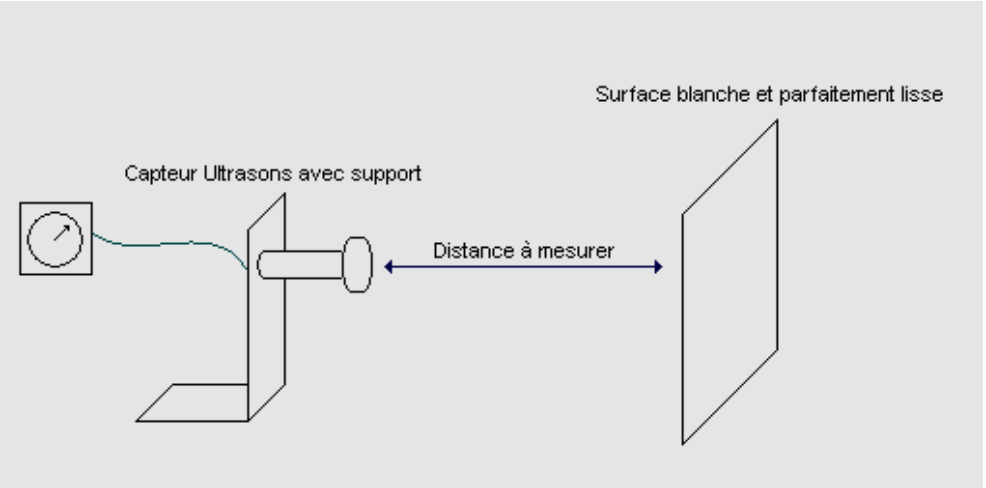



Il a été effectué plusieurs types de mesures afin de déterminer les différentes possibilités de ce capteur. J'ai pu déterminer :

- la portée maximum de ce capteur ;
- le portée minimum ;
- la relation liant la distance mesurée en fonction de la tension fournie par le capteur ;
- l'angle de réflexion du capteur ;
- le cône de mesure du capteur ;
- et les changements apportés par le changement de surface de réflexion.

a La plage de mesure du capteur.

Cette plage de mesure a été relevée avec une surface non rugueuse (métal ,bois, papier). On devrait donc obtenir la plus grande plage mesurable.
Avec la croquis ci – dessous, on peut apercevoir la réalisation pratique de la mesure.




Les mesures effectuées indiquent une plage de mesure allant de **30 centimètres à 4 mètres 50**. Le constructeur conseille d'effectuer des mesures entre **35 centimètres et 3 mètres**.
On dispose donc d'une plage suffisante pour notre application.

b La fonction de transfert du capteur.

Ayant déterminé la plage de mesure maximum, on va se fixer une plage de mesure fournissant une information correcte, c'est à dire entre **40 centimètres et 4 mètres**.
En premier lieu, avant de procéder à des mesures, on effectue un réglage grâce aux deux boutons poussoirs situés sur le capteur. Ces deux boutons permettent d'exécuter deux types de réglage, un qui permet de choisir la plage de mesure et un qui permet de choisir soit une mesure croissante (la tension augmente en fonction de la distance ) soit une mesure décroissante (la tension décroît en fonction de la distance.
Une fois ces réglages effectués, on peut commencer les mesures. Nous avons d'abord mesuré la tension du capteur pour les distances extrêmes. Ceci va nous donner la fonction de transfert du capteur.
On a relevé pour une distance de 40 centimètres, une tension de **300 mV** et pour une distance de 4 mètres, une tension de **9.3 volts.**
On a donc la relation suivante où U est la tension en volts aux bornes du capteur et L, la distance mesurée en mètres :

$$\boxed{U = 2.5 \times L - 0.7}$$

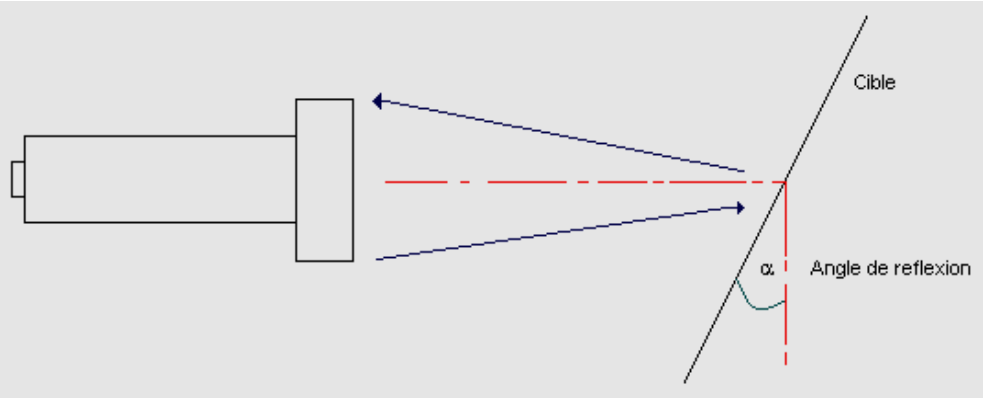
En plus deux cette relation, il a été possible de mesurer les erreurs de non linéarité du capteur grâce aux différentes mesures à intervalles réguliers. Ces mesures ont donné les résultats suivants :

| Distance mesurée (m) | Tension aux bornes du capteur (V) | Tension théorique (V) | Erreur relative (%) |
|---|---|---|---|
| 0.5 | 0.56 | 0.55 | 1.8 |
| 1 | 1.81 | 1.8 | 0.555 |
| 1.5 | 3.06 | 3.05 | 0.328 |
| 2 | 4.299 | 4.3 | 0.023 |
| 2.5 | 5.54 | 5.55 | 0.18 |
| 3 | 6.79 | 6.8 | 0.147 |
| 3.5 | 8.04 | 8.05 | 0.124 |

On s’aperçoit que l’erreur maximum obtenue est de 1.8 % et ceci lorsque l’on effectue une petite mesure. Par contre, cette erreur devient quasi nulle lorsque l’on effectue de grandes mesures.

c L'angle de réflexion.

Cette manipulation va nous permettre de déterminer les conditions de mesure. Cet angle de réflexion varie en fonction de la distance de mesure et de la nature de la cible c'est pour cela que les tests ont été effectué avec toujours la même surface, c'est à dire une cible blanche et parfaitement lisse.
Avec le croquis suivant, on peut voir comment s'est effectuée la mesure :



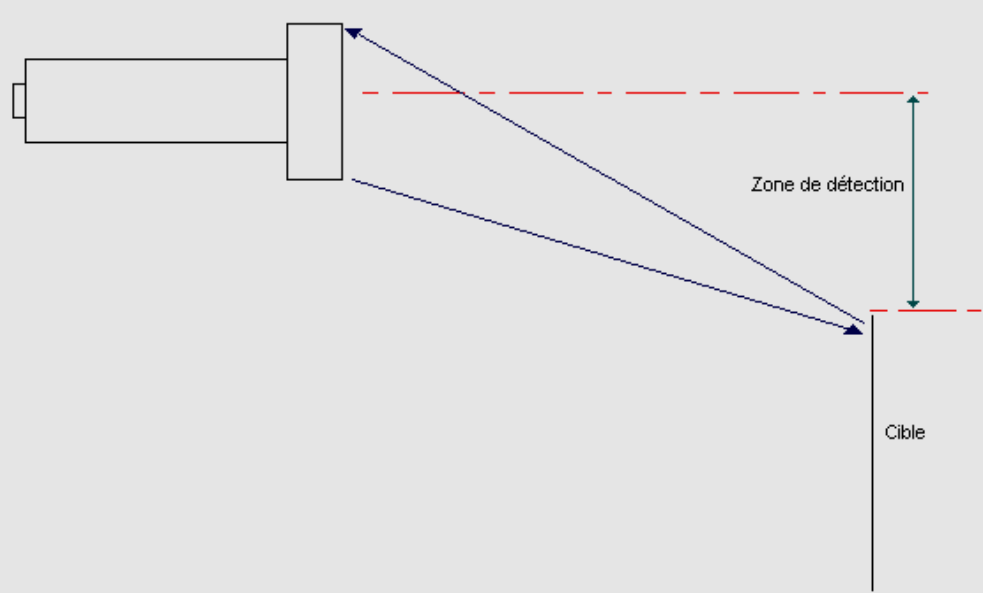

Dans le tableau suivant, on peut voir la relation existant entre la distance mesurée et l'angle de réflexion :

| Distance mesurée (m) | Angle de réflexion (degrés) |
|---|---|
| 0.5 | 72 |
| 1 | 28 |
| 1.5 | 24.8 |
| 2 | 17.1 |
| 2.5 | 12 |
| 3 | 12 |
| 3.5 | 8.8 |
| 4 | 8.8 |

On s'aperçoit que l'angle de réflexion décroît en fonction de la distance mesurée. Il faudra en tenir compte dans les autres mesures. Cette caractéristique du capteur peut être à la fois très gênante pour les mesures et un avantage.
En effet si le robot n'est pas parfaitement perpendiculaire aux obstacles, il pourra quand même les détecter. Par contre il existe un problème, si la cible n'est pas en face du robot et qu'un autre obstacle est entre le capteur et la cible, l'information sera faussée. Nous étudierons ce problème plus en détail dans une autre partie.

d La zone de détection du capteur.

Cette zone de détection s'exprime en mètres et indique la distance entre l'axe du capteur et la cible. Pour être plus précis, le croquis suivant donne une idée de cette grandeur :



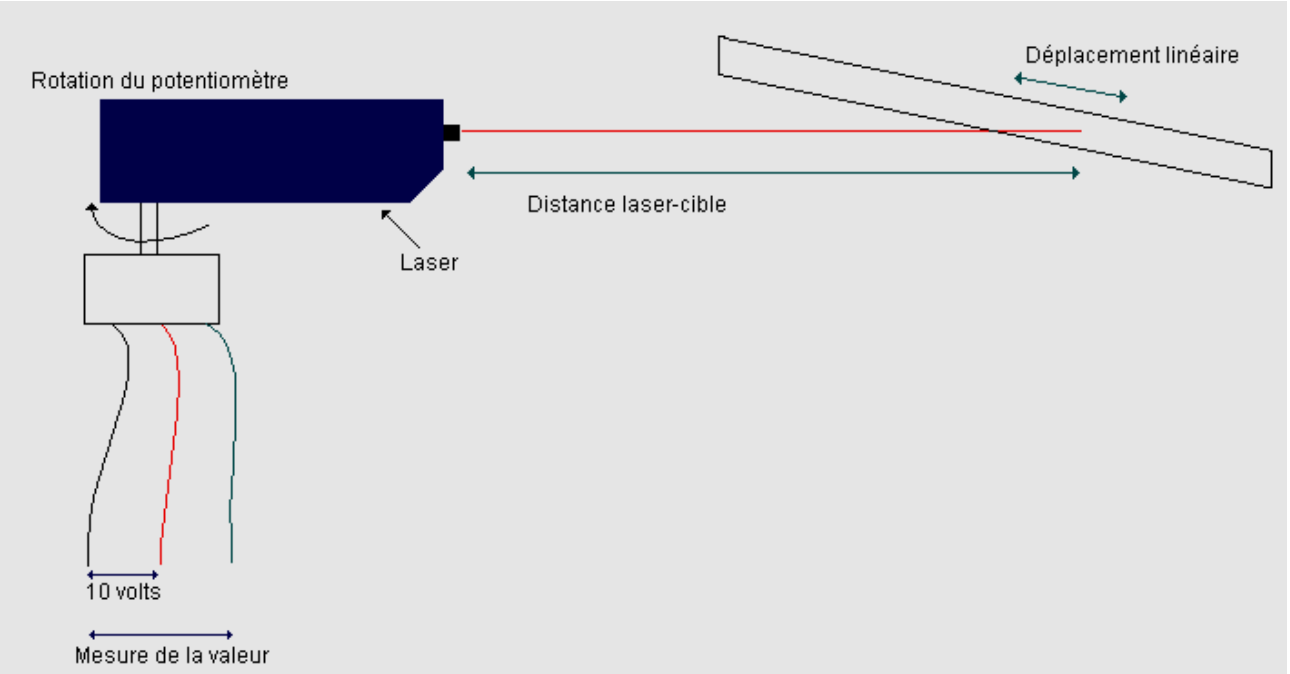

Cette zone évolue aussi en fonction de la distance mesurée. Avec le tableau suivant, on pourra mieux situer les performances du capteur :

| Distance mesurée (m) | Zone de détection (cm) |
|---|---|
| 0.4 | 18 |
| 0.5 | 21 |
| 1 | 29 |
| 1.5 | 37 |
| 2 | 47 |
| 2.5 | 58 |
| 3 | 48 |
| 3.5 | 41 |
| 4 | 41 |

### 3.1.2 Le potentiomètre.

(réf. Ma – 850 Constructeur : Megatron) :

Les essais qui ont été menés avec ce type de capteur avaient pour but de déterminer leur fiabilité et leur précision. Pour effectuer ces tests, nous avons monté un pointeur laser sur le potentiomètre. Lorsque nous faisions tourner le potentiomètre, la trace du laser faisait une translation. Connaissant la distance qui séparait le laser de la cible, on pouvait déterminer l'angle de rotation qui avait été effectué.
Le schéma suivant montre le type de montage réalisé pour avoir une idée plus précise de l'expérience :



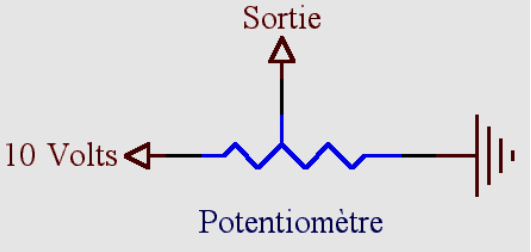

Pour trouver l'angle de rotation qui a été effectué, il suffit de mesurer le déplacement qu'a effectuée la trace du laser. Ensuite un simple calcul permet d'obtenir la valeur de cet angle :

$$a = \arctan\left(\frac{d}{D}\right)$$

$\alpha$ représente l'angle parcouru par le potentiomètre ;
d, le déplacement de la trace ;
D, la distance qui sépare l'axe du potentiomètre de la cible ;

Dans les essais que l'on a effectué, la distance potentiomètre – cible était de **5,73 mètres**.
Ainsi pour une variation de **½ degré**, le déplacement résultant est de **5 centimètres**.
Cette longueur de 5 centimètres était tout a fait mesurable, avec tout de même un précision de l'ordre du **½ centimètre** dû à la divergence de l'impact du laser. Il en résultait une précision, sur la mesure, de **0.05 degrés**.

Le potentiomètre, d'un point de vue électrique est branché comme le montre le croquis suivant :

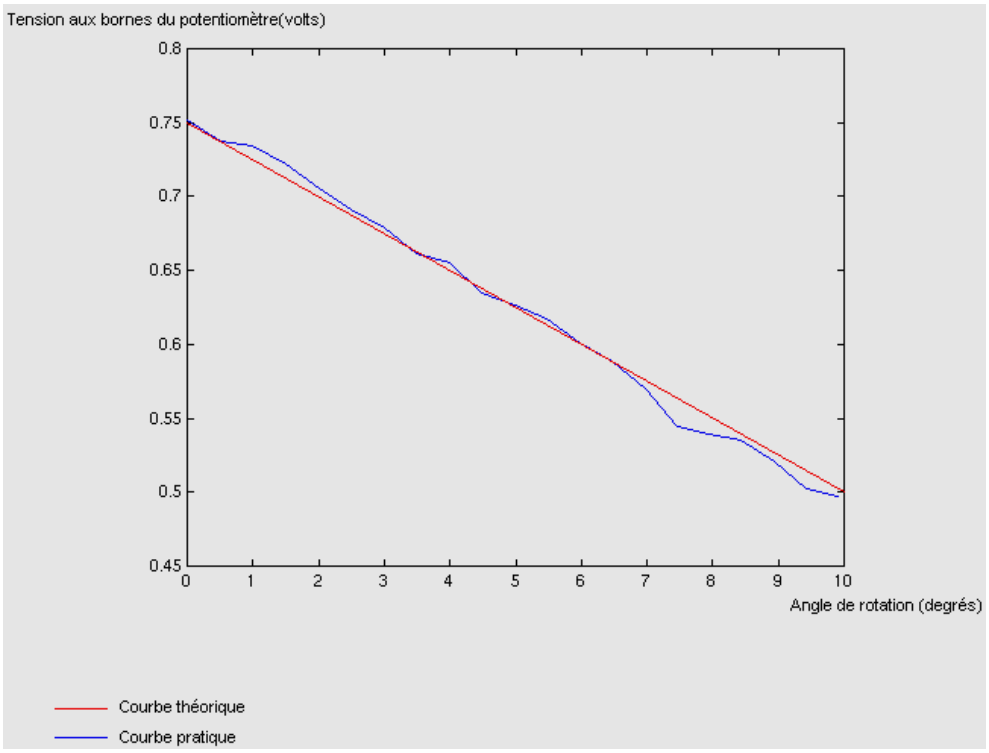



Il sera facile de faire un lien entre l'angle de rotation et la valeur de la tension aux bornes du potentiomètre. En théorie, cette relation est du type :

$$\alpha = k \times U$$

où $\alpha$ est l'angle de rotation exprimé en degrés ;
k, le facteur d'échelle en degrés/volts ;
U, la tension relevée aux bornes du potentiomètres donnée en volts ;

Grâce a cette réalisation, nous avons pu faire les relevés suivants :




Cette courbe présente un facteur d'échelle de **25 millivolts/degrés**. En théorie, celui – ci est donné par la relation suivante :

**k = U/360**

où U représente la tension d'alimentation du potentiomètre, exprimée en volts.

### 3.1.3 L'inclinomètre.

(réf. 42745 Constructeur : Sensorex) :

Cette campagne de mesures a le même rôle que celle du potentiomètre. Il faut vérifier, s'il est possible d'obtenir la résolution recquise par les futures applications.
Pour le robot, une précision du 1/10 degré est suffisante. Le constructeur indique une résolution de l'ordre du 1/100 de degré.

Les mesures se font par pas de 1/10 degré. Nous avons exécuté le même type de mesure que pour le potentiomètre. Grâce à la relation suivante, il a été possible de calculer l'angle d'inclinaison par rapport à un référentiel quelconque. Ce référentiel sera le premier point de l'acquisition.
Le croquis suivant montre la manipulation exécutée :


La variation angulaire se détermine par la relation suivante :

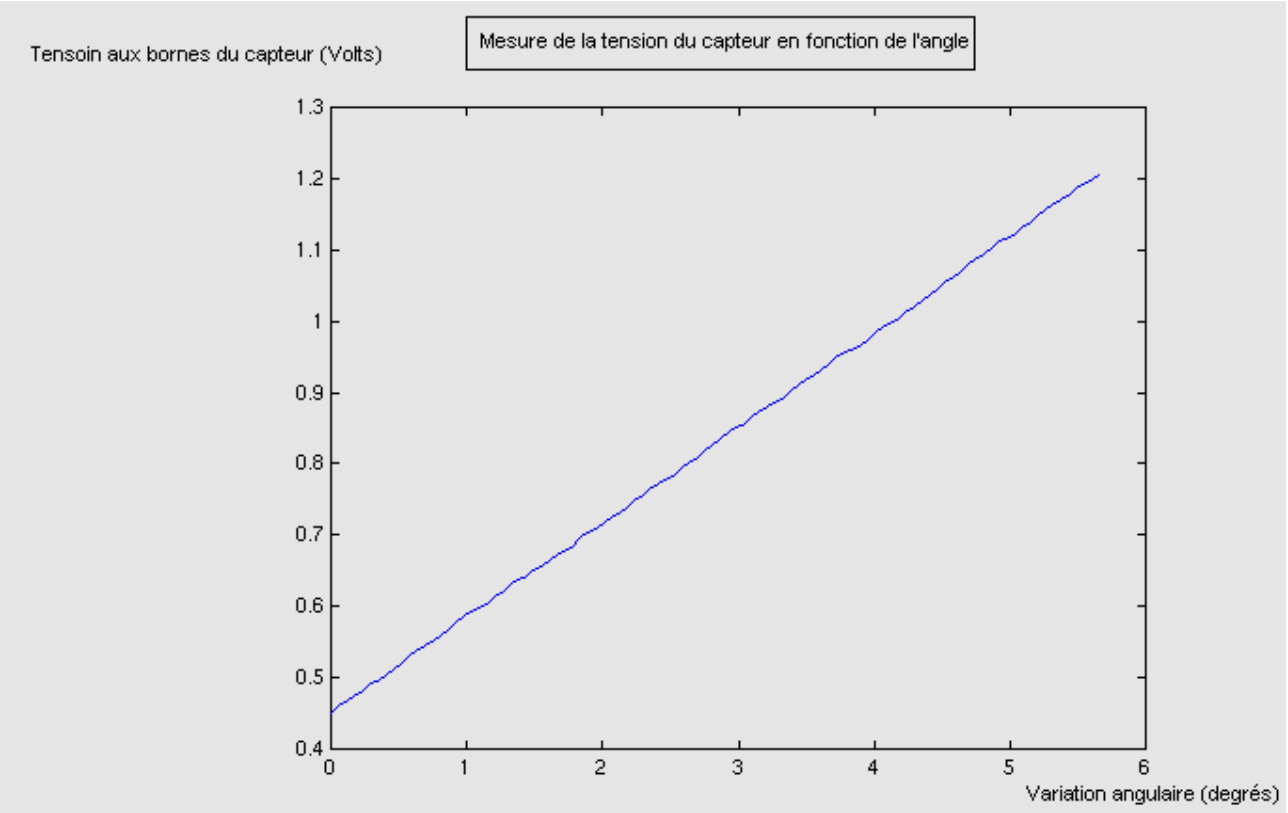
$$a = \arctan(\frac{d}{D})$$

$\alpha$ représente l'angle parcouru par le potentiomètre ;
d, le déplacement de la limite de l 'axe ;
D, la distance qui sépare l'inclinomètre de la limite de l'axe ;

Dans de nos essais, l'axe a une longueur de **1 mètre**. Nous déplaçons l'axe de **1 millimètres**. Cela faisait don une variation angulaire de **0.0573 degrés**. La précision voulue était atteinte.

En théorie, ce capteur délivre une tension de **130 millivolts** en sortie pour un variation de 1 degré. Quand l'axe du capteur est posé sur un sol plan et perpendiculaire a la verticale terrestre, la tension de sortie doit être nulle. Il faudra donc procéder à un essai pour déterminer si il y a un offset. Si cet offset existe, il faudra l'annuler.

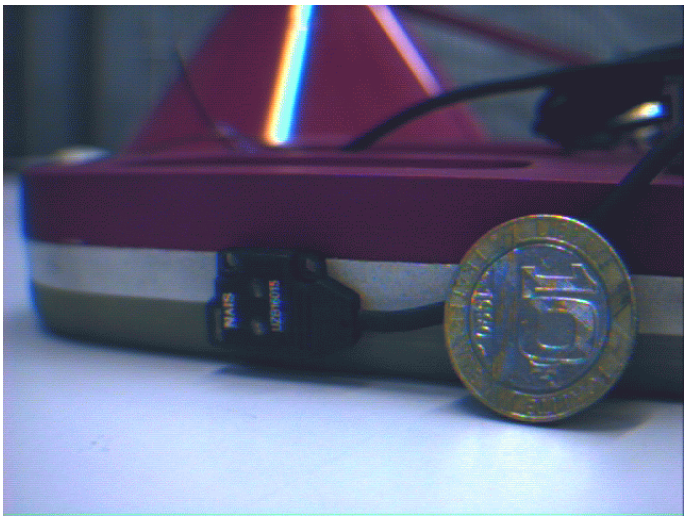
Cette manipulation nous a permis de faire les relevés suivants :




Les caractéristiques constructeurs donnent une variation de **130 millivolts par degré**. Les calculs issus de nos mesures fournissent un facteur d'échelle de **132 millivolts par degré**.

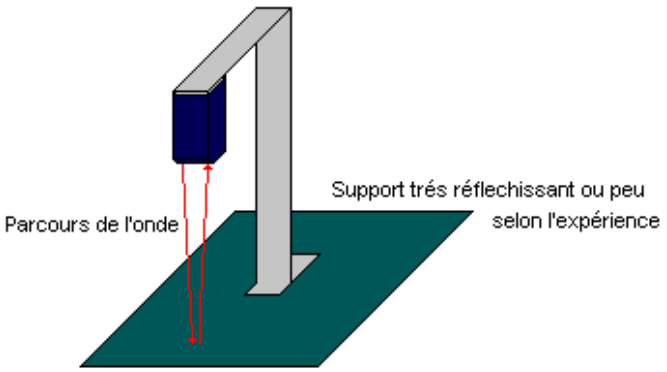
### 3.1.4 Les capteurs infra rouges.

#### a Photographie des capteurs.



#### b Les essais effectués avec les capteurs.

Grâce au croquis suivant, on peut voir la manipulation effectuée avec les deux capteurs :



c Le capteur courte distance.

(réf. UZ-B1601 Constructeur : Matsushita) :

Ce capteur qui a la particularité d'être très petit – comme le montre les photos précédentes - présente l'avantage de mesurer de très courtes distances. Il serait idéal pour détecter l'approche du pied sur le sol lors des mouvements du robot.
Ce capteur ayant une sortie tout – ou – rien, il est difficile de connaître la position exacte de la cible. Ce capteur n'est capable de détecter que des seuils, il faudra donc l'utiliser judicieusement sur le robot pour pouvoir profiter des avantages de ce capteur.
Les essais réalisés avec ce capteur ont donné deux résultats. En effet, on sait que l'information retournée par ce type de capteur dépend essentiellement de la distance capteur – cible et de la nature de la cible. Nous avons effectué deux essais, un avec une surface très réfléchissante ( une feuille d'aluminium) et un essai avec une surface assez absorbante (une feuille de mousse noire).
Les résultats suivants indiquent la distance à laquelle, la sortie du capteur commute.

| Type de surface | Distance de commutation |
|---|---|
| Très réfléchissante | 12 mm |
| Peu réfléchissante | 58 mm |

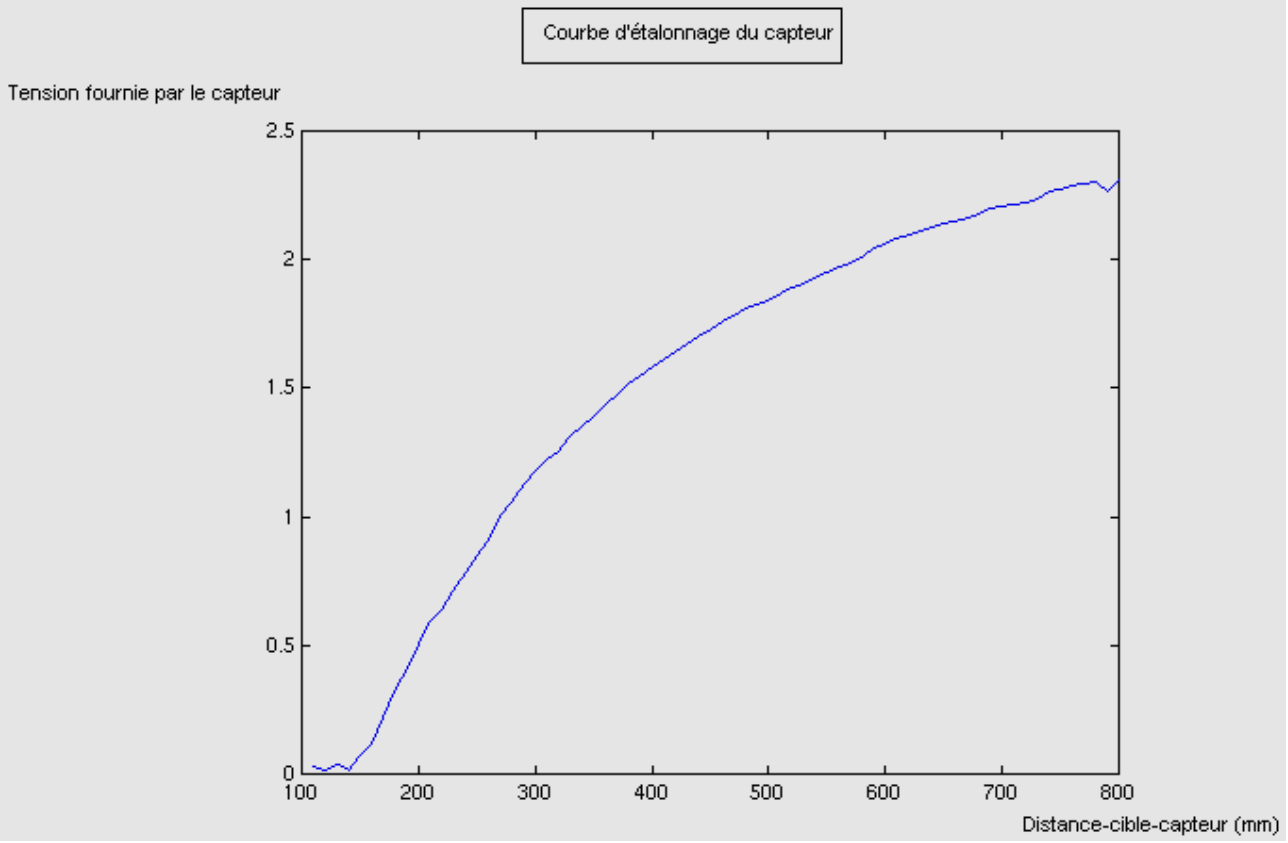
Ce capteur présente un inconvénient majeur. (Cf. § 3.2.2)

d Le capteur moyenne distance.

(réf. AKM17165 Constructeur : Matsushita) :

Ce capteur photoélectrique, qui a l'avantage de fournir une information analogique, est complémentaire au capteur à ultrasons. En effet, sa technologie est différente et les effets parasites ne sont pas le mêmes que ceux de l'ultrasons.
Les essais qui ont été réalisés avec ce capteur, ont permis de déterminer les caractéristiques nécessaires à son implantation sur le robot.
Nous avons fait quelques mesures qui serviront plus tard à retrouver la distance mesurée avec le capteur.
Etant donné que le capteur n'est pas linéaire, il n'existe pas de correspondance entre la distance mesurée et la tension relevée aux bornes du capteur. Il faudra donc retrouver la distance avec un tableau de valeurs. Bien sûr il faudra faire quelques interpolations, mais nous garderons toujours une précision supérieure à deux millimètres.

On peut voir, grâce au graphique suivant, qu'il est difficile de trouver une relation entre la distance et la tension.

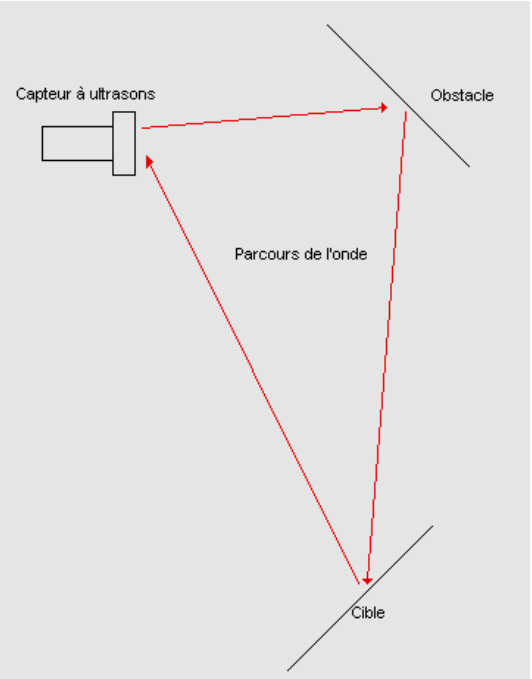



Cette courbe est utilisable sur un partie de la pleine échelle. En effet, pour une distance comprise entre **0 et 150 millimètres**, les résultats sont incohérents avec le reste de la courbe. On note ceci aussi, lorsque la distance à mesurer est supérieure à **775 millimètres**.
Bien sûr, on pourrait intuiter l'expression de cette courbe, mais le résultat ne serait pas assez convenable pour pouvoir exploiter les futures acquisitions.

## 3.2. Les inconvénients de certains capteurs et les moyens d'y remédier.

### 3.2.1 Le capteur à ultrasons.

Ce capteur présente quelques inconvénients. Le premier est un problème de détection. En effet, si l'axe du capteur n'est pas parfaitement perpendiculaire aux obstacles, il pourra quand même les détecter. Par contre il existe un problème, si la cible n'est pas en face du robot et qu'un autre obstacle est entre le capteur et la cible, l'information sera faussée. On pourra mieux comprendre ce problème avec le croquis ci – dessous.




De plus ce capteur présente un autre défaut. Lorsque la distance capteur – cible est comprise dans la zone morte (**entre 0 et 30 cms** ), le capteur fournit une information totalement fausse et pas du tout exploitable. Il faudra donc faire attention à ce problème et trouver une solution soit matérielle, soit logicielle pour avertir l'opérateur et bien sûr ne pas tenir compte de ces mesures.

On peut par exemple étudier le signal et faire un montage en conséquence pour détecter ce seuil. On sait que l'on atteint une distance inférieure à **30 centimètres** lorsque la tension aux bornes du capteur devient inférieure à **400 millivolts**.
Un comparateur à seuil conviendra pour ce type de détection.

On peut aussi réaliser un détecteur de seuil à l'aide d'un programme. Un indicateur (**flag**) avertirait l'opérateur si le seuil était atteint. Le réarmement s'effectuerait aussi d'une manière logicielle.

### 3.2.2 Les capteurs photoélectriques.

Le problème majeur rencontré avec les capteurs photoélectriques concerne uniquement le capteur courte distance. En effet comme je l'ai signalé, l'information portée par ce capteur dépend essentiellement de la nature du sol. Etant donné que le robot ne sera jamais dans le même endroit, en l'occurrence ne rencontrera jamais le même type de sol, on ne peut pas faire entièrement confiance à ce capteur.
L'utilisation de ce capteur pour la détection du sol n'est donc pas possible. On pourra par contre utiliser des micro interrupteur, placés sous la voûte plantaire. Ces capteurs apporteront une information plus sûre.

Quant au capteur moyenne distance, son seul inconvénient est sa faible portée. Mais il sera résolu en l'utilisant conjointement avec le capter à ultrasons.

## 3.3 Acquisitions des mesures sur rack VME.

### 3.3.1 Présentation des outils.

J'ai effectué les mesures avec les outils suivants :

- un processeur sur VME, cartes E/S ;
- logiciel d'acquisition.

#### a Le bus VME.

Le bus VME est un bus multiprocesseur supportant des cartes d'Entrées/Sorties. Sur ce bus sont connectées plusieurs cartes d'entre sortie, et une carte Motorola à base de processeur 68040 pilotée par le système temps réel VxWorks. Nous utiliserons les cartes d'Entrée analogique pour effectuer la mesure des variables capteurs.
Il existe 4 types de cartes qui sont connectées sur le bus :

- Module IP – Quadrature :

Ce module gère le décodage des signaux renvoyés par les résolveurs moteurs du bipède. Cette carte implante 4 décodeurs de 12 entrées.

- Module IP – Unidig I-O :

Ce module gère 24 entrées ou sorties tout – ou – rien.

- Module IP – 16 ADC :

Cette carte possède 16 convertisseurs Analogique – Numérique simple voie ou 8 en différentiel. Le fait d'utiliser la carte en différentiel permet de limiter le bruit sur les voies.

- Module IP – DAC SU :

A l'opposé du module IP – 16 ADC, cette carte contient 16 voies de conversion Numérique – Analogique qui fonctionnent jusqu'à 10 kHz.

Dans le cas de nos mesures, il sera nécessaire d'utiliser le module qui gère les acquisitions et les conversions. Il s'agit du module IP – 16 ADC. Il faudra donc configurer cette carte à l'aide de pilotes qui seront écrits avec le langage C.

b Le programme d'acquisition.

(Cf. Annexe 1)

Ce programme dispose d'un menu qui permet de sélectionner plusieurs types d'acquisitions dont les principales caractéristiques sont le nombre de voies à saisir et le type de fréquence (c'est à dire soit une fréquence élevée de 1000 Hertz, soit une fréquence plus faible comprise entre 0 et 60 Hertz).

Des acquisitions rapides étant nécessaires, il faut disposer d'un programme qui puisse exécuter des tâches à des fréquences élevées. Pour cela, au lieu de stocker le valeurs directement dans un fichier ASCII, il faudra d'abord stocker ces valeurs dans des tableaux.
Ces tableaux devront comporter deux dimensions, une qui indiquera l'instant de l'échantillon et une qui indiquera la valeur acquise.
L'intérêt avec ces tableaux est que le temps d'écriture est très faible (de l'ordre de la micro seconde ), on peut donc, en théorie travailler à des fréquences élevées. Ce sera une des particularités de ce programme.

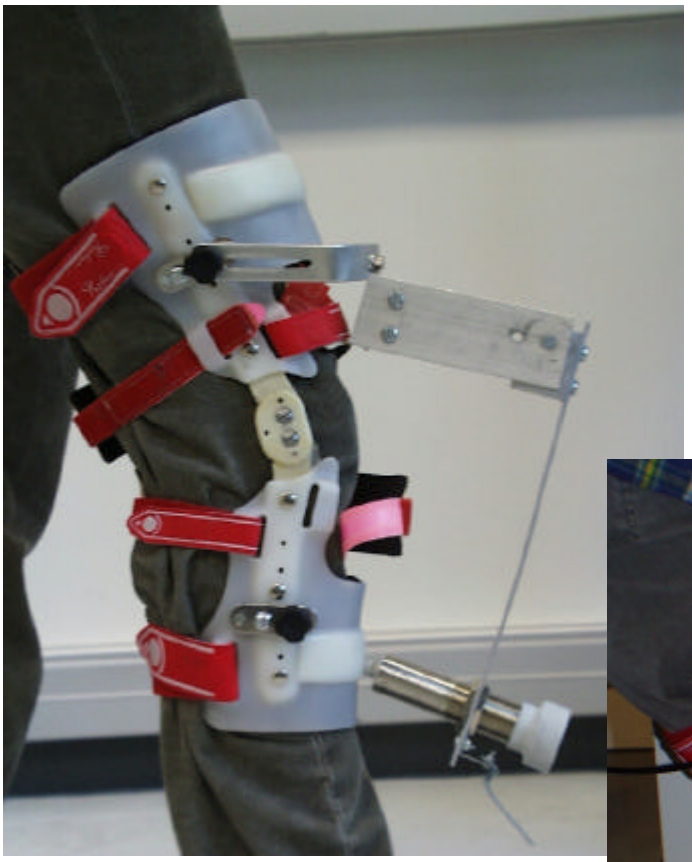
D'autre part le programme d'acquisition doit être capable d'ouvrir des fichiers textes et d'y écrire toutes les informations nécessaires. Ces fichiers textes seront structurés d'une manière bien définie. Ce fichier comportera une colonne qui indiquera l'instant de l'échantillon et puis autant de colonnes que de nombre de voies acquises.
Le fait de faire des acquisitions sur plusieurs voies en même temps se révèle très intéressant pour les capteurs de force qui sont étudiés par Damien Tourres (Cf. Bibliographie N°1).
En effet ces capteurs permettent de connaître avec exactitude les forces appliquées sur le pied. Ensuite, grâce à des calculs, il est possible de retrouver toutes les données relatives au torseur qui agir sur ce pied.

Ce type de mesure nous servira aussi lorsque l'on voudra faire des acquisitions synchronisées des différents capteurs de proximité et de l'inclinomètre. Ceci devrait nous aider, grâce aux transformées inverses, à retrouver le profil du sol rencontré par le robot bipède.

Pour avoir quelques compléments sur le programme utilisé pour les acquisitions, le fichier source se trouve en annexe. Il ne comporte pas toute la partie qui concerne l'initialisation et la configuration du module IP.

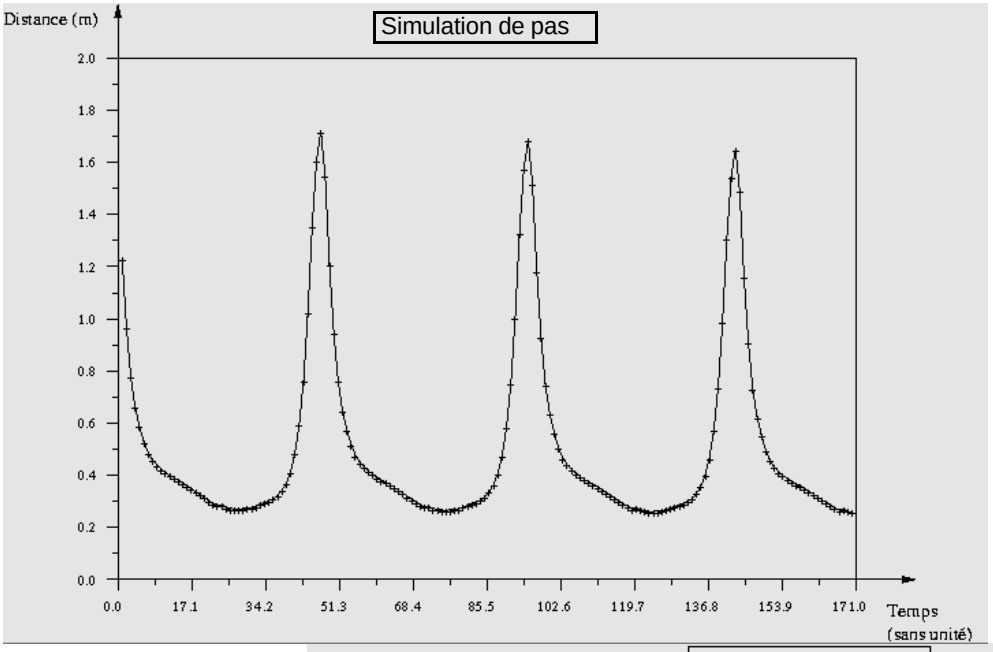
### 3.3.2 Les acquisitions.

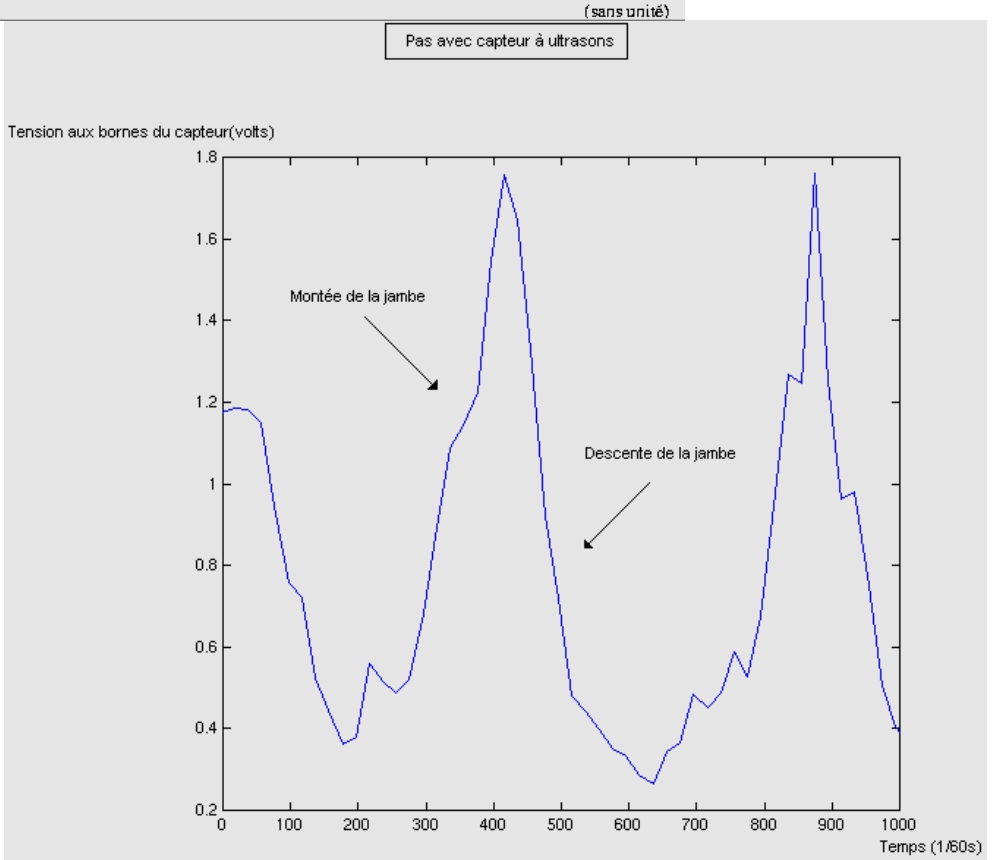
Les acquisitions qui vont être effectuées avec la carte ADC du bus VME ne concernent que le capteur à ultrasons, le capteur photoélectrique, et l'inclinomètre.
Les mesures qui vont être effectuées se situeront dans les meilleures conditions, celles qui se rapprochent le plus de celles du robot.
Les capteurs à ultrasons et photoélectrique seront placés sur la jambe pour obtenir des résultats qui pourraient servir à l'étude du robot bipède.
Ils formeront un angle de 20° avec la perpendiculaire du tibia et seront à 16 centimètres du sol.
Avec les photos ci – dessous, on peut apercevoir l'emplacement de ces deux capteurs.



a La simulation de pas avec le capteur à ultrasons.

Les simulations se feront pour voir comment le robot peut réagir aux différentes contraintes. Les résultats obtenus pourront être confrontés avec ceux obtenus par Laure France (Cf. bibliographie N° 2), étudiante en thèse. Les simulations que Laure France a menées ont été réalisés à l'aide d'outils informatiques. Ces simulations ont été menées en essayant de se placer dans les mêmes conditions que les nôtres . C'est à dire, angle entre le capteur et la jambe, distance du capteur au sol, et portée du capteur.
La courbes suivantes montrent les simulations faites par Laure France et notre équipe :



Sachant que l'on peut décomposer le pas humain en trois phases, on peut retrouver ces étapes :

- La première phase commence lorsque le pied est posé à terre. A ce moment, le pied se trouve au plus prés du sol et le « tibia » est perpendiculaire au plan formé par le sol (en considérant que le plan du sol est perpendiculaire à la verticale terrestre). Donc par rapport à la seconde courbe, on se trouve à l'instant numéro 200.

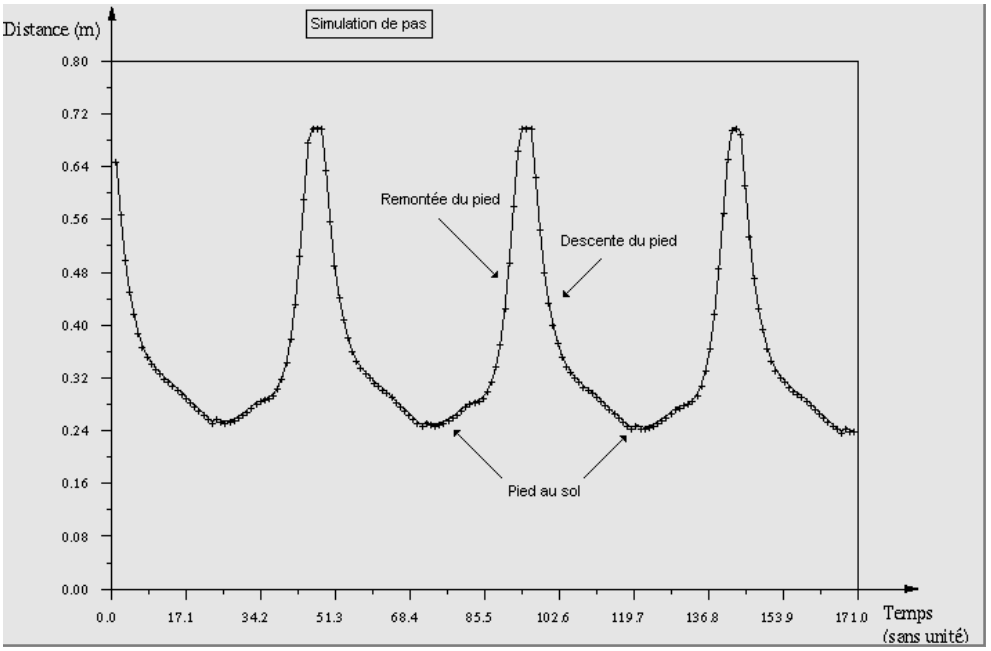
- Ensuite, lorsque le pied remonte, le tibia fait un arc de cercle et la distance doit théoriquement progresser linéairement. C'est bien le cas pour la simulation de Laure France. Lors de l'essai pratique, la remontée du tibia est en pratiquement linéaire. On note une zone, à l'instant 350, une non linéarité. En effet la marche humaine ne correspond pas tout à fait avec celle simulée par Laure France.

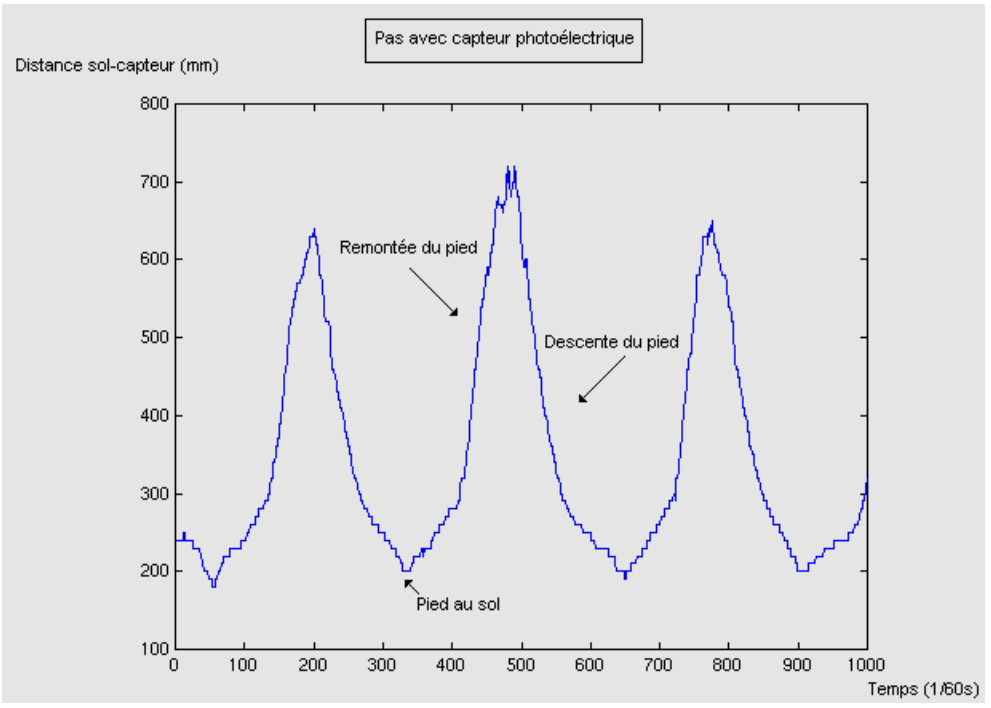
- Enfin, il existe une dernière phase où le pied redescend. Durant cette étape, on peut faire les mêmes remarques que précédemment. En plus de ça, lorsque le pied approche du sol, on s'aperçoit qu'il ne descend plus linéairement. Ceci est tout à fait normal, car l'homme freine naturellement sa course pour pouvoir relancer son pied, ceci pour exécuter un nouveau pas.

Les tests menés avec ce capteur, nous ont aidés à rechercher les problèmes que l'on pouvait rencontrer avec ce type de capteur. En effet les capteur à ultrasons, bien qu'offrant une plus grande plage de mesure que les autres capteurs de proximité utilisés, présentent un inconvénient majeur. De temps en temps l'information reçue est totalement aberrante. L'onde émise par ce type de capteur peut réfléchir sur plusieurs surfaces successives sans que l'on soit averti. On comprend vite la nécessité d'intégrer au robot un autre type de capteur de proximité.
Cependant cette expérimentation valide la simulation théorique de la marche (au niveau capteur), c'est un point très important, et l'un des objectifs de ce stage.

b La simulation de pas avec le capteur photoélectrique.

Cette simulation s'effectuera avec les mêmes caractéristiques que celles du capteur à ultrasons. C'est à dire que le capteur se situera à 16 centimètres du sol et formera un angle de 20° avec la perpendiculaire du tibia.
Les résultats obtenus sont les suivants :

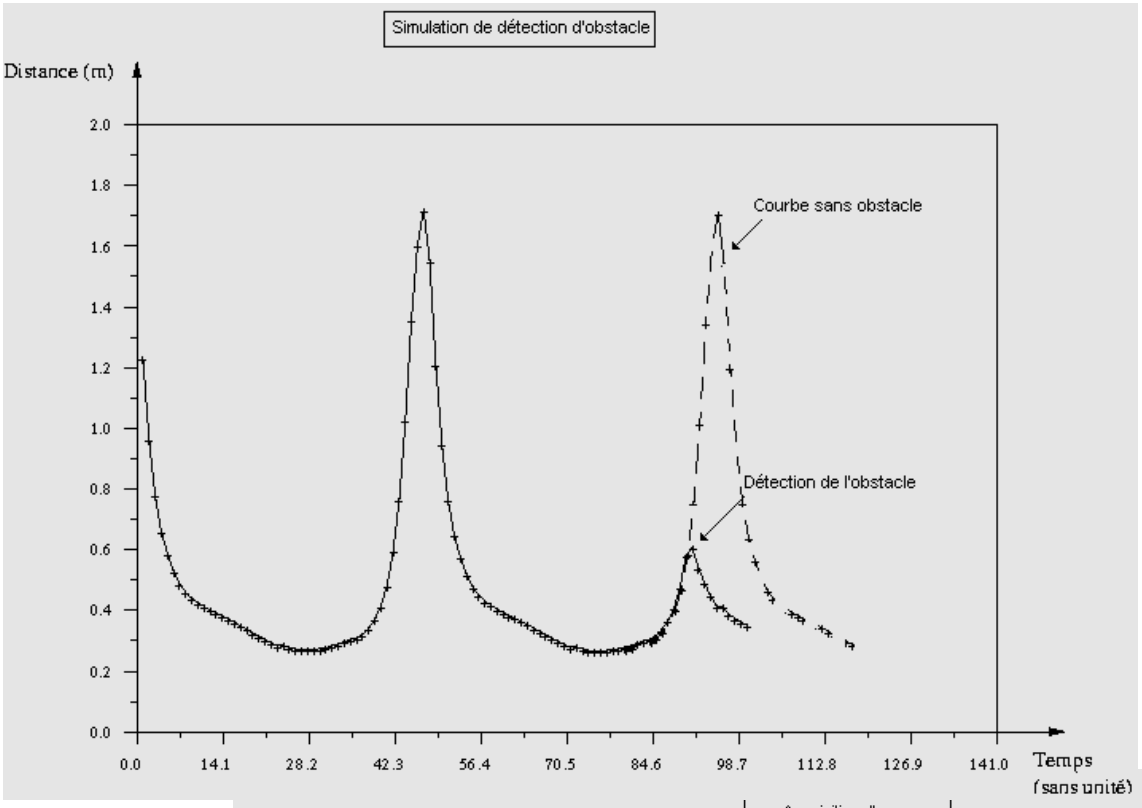


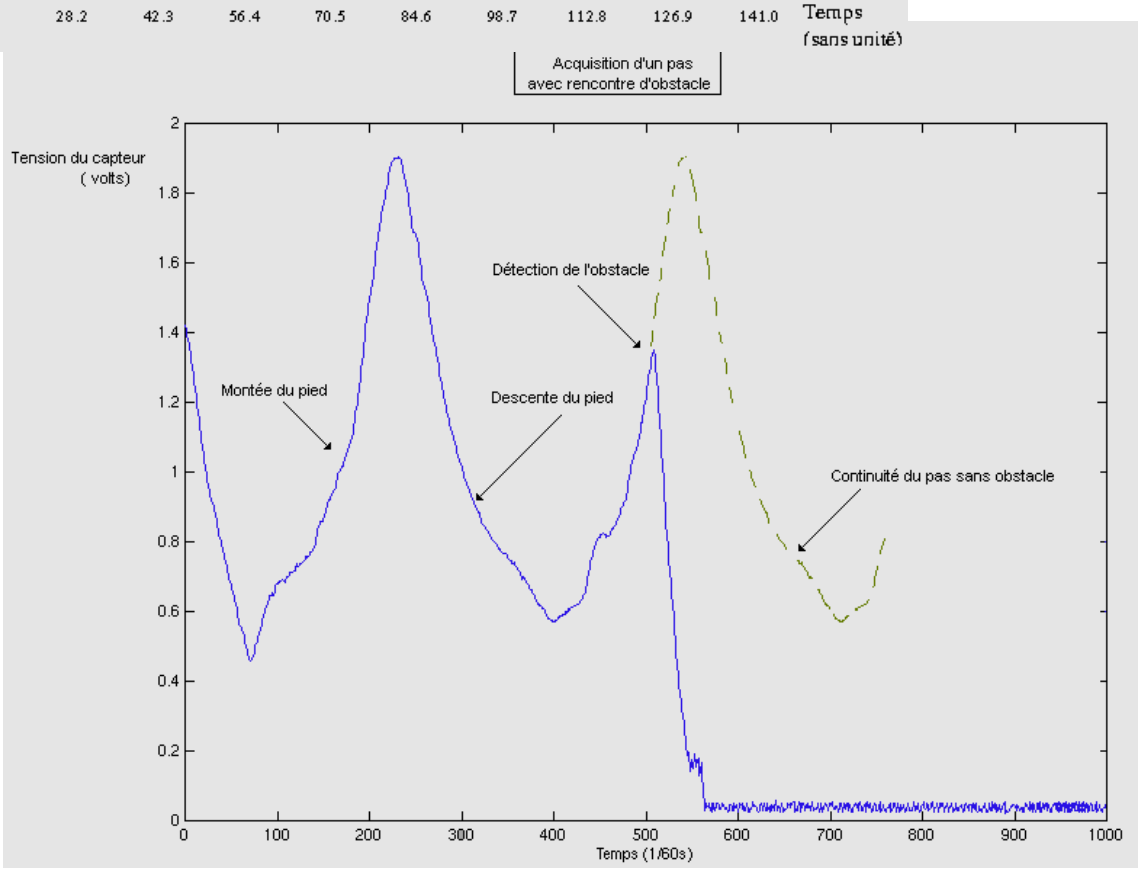




Il existe toujours les mêmes étapes en ce qui concerne le pas.
La seule différence et que les mesures sont plus fidèles aux capteurs.
C'est une des raisons qui autorise l'autorisation de ce capteur photoélectrique en complément de l'ultrasons.

c Détection d'obstacle avec le capteur photoélectrique.

Cette détection d'obstacle a pour but de comprendre comment réagissent les capteurs de proximité dans ces conditions. Ces essais et simulations ont été réalisées en entreposant une surface carrée sur la course soit de l'individu réel ou virtuel.
Nous arrivons Laure France et notre équipe, à des résultats assez convaincants. En effet simulation et expérimentation fournissent des courbes similaires.



## 3.4. Utilisation du logiciel Matlab pour l'étude plus approfondie des capteurs.

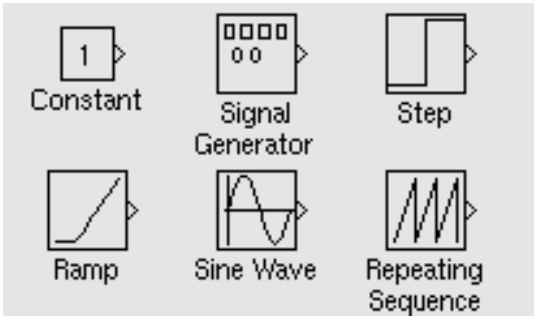
Cet outil mathématique dispose d'une interface graphique qui permet de modéliser et simuler les problèmes posés par les systèmes automatisés. Ce logiciel est partagé en deux parties :

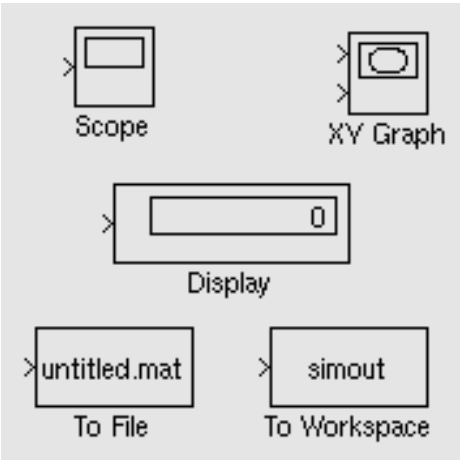
- une qui permet donc des calculs mathématiques ;
- une autre qui permet de faire des simulations ;

D'autres modules (acquisition temps réel, traitement du signal, …) peuvent compléter le logiciel.

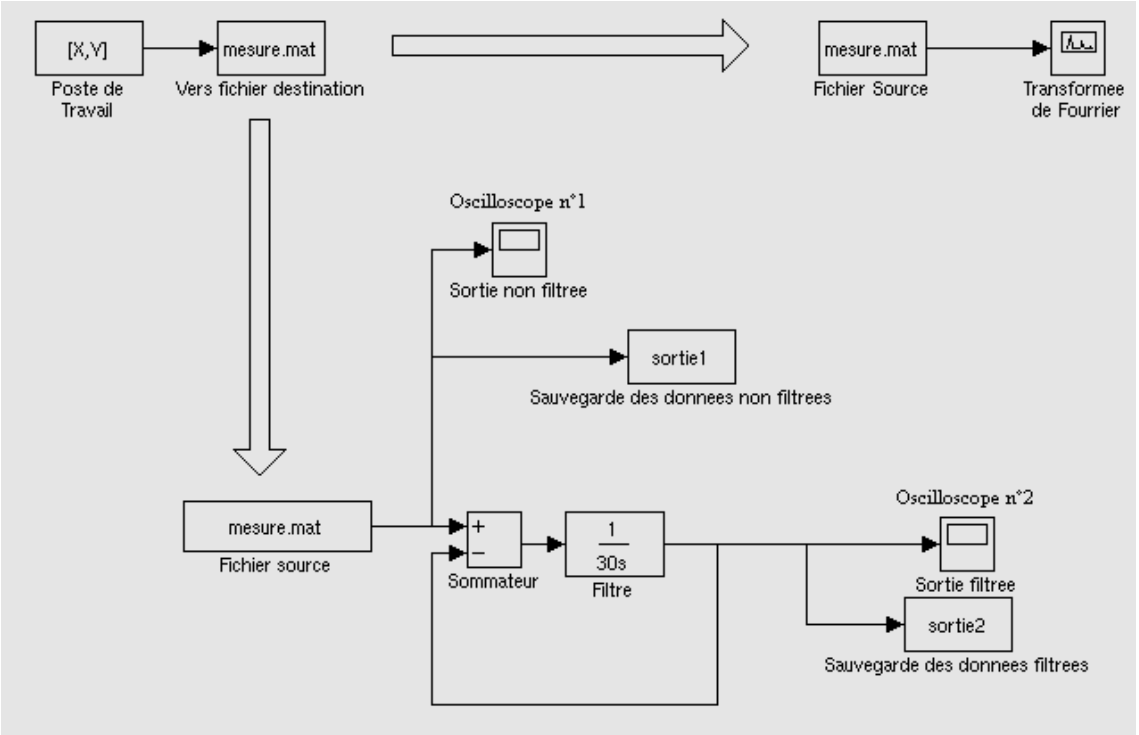
***Simulink :***

Cette partie du logiciel comme son nom l'indique permet d'exécuter des simulations grâce à des outils très développés tels des blocs fonctionnels du Enième degré, des dérivateurs, des intégrateurs, des blocs mémoires, des outils d'affichage, …
Ces outils se présentent sous forme d'icônes comme le montre la figure ci – dessous :



En combinant les différents icônes, on peut arriver à faire aussi bien des systèmes très compliquées mais aussi des systèmes très simples. Le croquis suivant montre un aperçu d'espace de travail de Matlab.




Bien entendu les résultats obtenus sont souvent différents de la pratique, c'est pour cela que l'on peut inclure des outils qui permettent d'inclure des non linéarités dans les systèmes. On dispose d'effet de saturation, d'offset, de zone morte, de mémorisation, de limitation de pente, …
Tous ces outils vont nous permettre de modéliser et simuler toutes sortes de systèmes physiques .

En rajoutant des commandes propres à Matlab, il est possible de créer des scripts qui permettront d'exécuter des tâches entièrement automatisées.
Ces tâches sont de plusieurs types :

- Affichage des courbes sur des graphiques ;
- Détermination de la forme mathématique des courbes obtenues ;
- Calcul des paramètres des courbes obtenues ;
- Calcul de moyennes, d'écarts-type, de tangentes, … ;
- Détermination de non linéarité, de saturation ;

# CONCLUSION

Faire ce stage dans un Institut de recherche comme l'INRIA, apporte beaucoup de connaissances tant au niveau théorique que pratique (utilisation d'outils spéciaux).
En effet, cela m'a apporté toutes les notions qui serviront quant à l'utilisation de tous les outils électroniques, informatiques, …

Le fait d'avoir passé les quelques semaines à l'INRIA Rhone-Alpes, où j'ai pu rentrer en contact avec les différentes personnes de l'Institut, m'a apporté une certaine confiance dans le travail qui m'était confié.
Je pense que les tâches demandées ont été pleinement réalisées. Avec la participation des ingénieurs des Moyens Robotiques.
Chacune des personnes qui forment cette équipe m'ont aidé lorsque des connaissances me manquaient :

- M. Pascal Di Giacomo et Gérard Baille, en ce qui concernent toute la partie réalisation des circuits électriques, étude des capteurs, ..
- M. Hervé Mathieu et Roger Pissard, m'ont fourni toutes les informations qu'il me manquait pour utiliser le matériel informatique et les logiciels.

Je tiens à remercier ces personnes pour le temps qu'ils m'ont consacré, et pour les connaissances que j'ai acquis à leur côté.

# BIBLIOGRAPHIE

Différents rapports de stage et de thèse :

[1] D. Tourres, « Les capteurs de force d'un robot bipède », rapport de stage IUT1 Grenoble, 1999 ;
[2] L. France, « Simulation d'un robot bipède dans un environnement », thèse de 3ème cycle en cours de rédaction, Grenoble 1999 ;
[3] S. More, « Schémas électriques d'un prototype de robot bipède », rapport de stage IUT 1 Grenoble, 1998 ;

Catalogues de capteurs et composants électroniques:

[4] Radiospare : Catalogue Radiospare 1999 ;
[5] Sensopart : Documentation technique Sensopart 1998 ;
[6] Sensorex : Documentation technique Sensorex 1996 ;
[7] Corelem : Documentation technique Corelem 1996 ;

Documentations des logiciels utilisés :

[8] Matlab : documentation en ligne sur système UNIX solaris 2.6 - 1996;
[9] Logiciels d'acquisitions, et de traitement d'images;
[10] UNIX ;

[11] G. Asch, « Les capteurs en Instrumentation Industrielle », Dunod - 1996.
[12] P. Di Giacomo, « Les capteurs en robotique »cours ITMI - 1985.

# ANNEXE 1

Sources du programme utilisé pour les acquisitions.

**Ce listing ne comporte que la partie chargée des acquisitions. La partie réservée à la configuration du module IP n'est pas présente**.

```
#include <stdio.h>
#include <math.h>
#include <taskLib.h>
#include <tickLib.h>
#include <semLib.h>
#include <sysLib.h>

/* structure de tableau pour
stocker les valeurs en temps reel pour plusieurs voies*/
typedef struct {
        double valeur [4];
        int temps;
} EFFORT_DATA_MULTI;

/* structure de tableau pour
stocker les valeurs en temps reel pour une seule voie*/
typedef struct {
        double valeur;
        int temps;
} EFFORT_DATA;


/* ************************************************************/
/* Define et Variables pour Acqui sur horloge                 */
/* ************************************************************/

#define HL_ACQUI  900       /* nb de ticks par seconde                */
#define MAX_ACQUI 9000      /* nb maxi d'acqui                        */

static int EndAcqui;        /* Flag pour l'arret de l'Acqui           */
static SEM_ID SemHl;        /* Semaphore pour synchroniser l'acqui    */
static int SampleAcqui;     /* Compteur du nbre d'echantillons        */

extern int InitAcqui();     /* Fct d'init de l'it                     */
extern int SynchroAcqui();  /* Fct d'it pour l'acqui                  */
extern int Acqui();         /* Fct d'acqui synchronise                */

/* tableau de stockage des donnees pour une seule voie*/
EFFORT_DATA effortData[MAX_ACQUI+1];

/* tableau de stockage des donnes pour plusieurs voies*/
EFFORT_DATA_MULTI effortDataMulti[MAX_ACQUI+1];

/* variable de choix multi-voies ou une seule voie*/
static int acquiMulti = 0;
```

```
void effortMenuPrint()
{
	/* Menu de selection */
	printf("\nAttention faire avant tout autre chose: i for init ! !\n");
	printf("i- Init ADC\n");
	printf("e- Exit menu\n");
	printf("a- Acqui d'une voie de l'ADC\n");
	printf("m- Multpile sequence d'acquisition des capteurs de force (valeur, temps)\n");
	printf("s- Sequence d'acquisition de couples pour une voie (valeur, temps)\n");
	printf("c- Sequence d'acquisition avec demande de Confirmation pour etalonnage
capteurs\n");
	printf("r- acquisition Rapide d'une ou plusieurs voies\n");
}
void effortMenu()
{
	char c;   /* variable de selection de fonction */
	int fin = 0;                                    /* flag de fin de programme      */
	while(fin == 0) {
		effortMenuPrint();                      /* affichage du menu             */
		c = getchar();                          /* saisie du choix               */
		if ((c != '\n') && (c != '\r')) {       /* si rien n'est rentré au clavier, on revient
	au menu */
		switch(c) {
		case 'e':
			fin = 1;                        /* mise à  du flag de fin de programme */
			break;

		case 'i':
		if (ipInitADC() != OK) {                /* fonction d'initialisation */
			fprintf(stderr, " Probleme pour initADC\n");
		} else {
			fprintf(stderr, " initADC ok\n");
		}
		break;

		case 'w' : {
		EFFORT_DATA *effortData;        /* pointeur du tableau EFFORT_DATA */
		int nb;                         /* Nbre d'echantillons           */
		int i;                          /* numero d'echantillon          */
		int DistMes;                    /* distance mesuree              */
		int DistDep;                    /* Distance de depart            */
		int DistFin;                    /* Distance d'arrivee            */
		int voie;                       /* Numero de la voie             */
		FILE *fp;                       /* Variable file pour stocker les donnees */
		printf("quelle voie?\n");
		scanf("%d", &voie);             /* saisie du numero de la voie   */
		printf("distance de depart ? ");
		scanf("%d",&DistDep);           /* saisie de la dist. de depart  */
		printf("distance de d'arrivee ? ");
		scanf("%d",&DistFin);           /* saisie de la distance d'arrivee */
		printf("combien d'acquisition?\n");
```

```
                        scanf("%d", &nb);                   /* saisie du nbre d'acquisitions   */
                        effortData = (EFFORT_DATA *)malloc(nb * sizeof(EFFORT_DATA));
                        if (effortData == NULL) {
                                fprintf(stderr, "Probleme d'allocation memoire\n");
                                break;
                                }

/* calcul de l'intervalle entre deux mesures  pour l'étalonnage du capteur infra – rouge */
                        DistMes = DistDep + (DistDep-DistFin)/nb;
                        for (i=0;i<nb;i++) {
/* decrementation de la distance mesurée */
                        DistMes = DistMes - (DistDep-DistFin)/nb;
/* saisie d'une variable quelconque pour confirmation de l'acquiisition */
                        c = getchar();
/* affectation a effortData.valeur de la valeur sur le CAN*/
                        effortData[i].valeur = ipReadAnalog(voie) ;
/* affectation a effortData.temps de la distance mesurée */
                        effortData[i].temps = DistMes ;
/* affichage des acquisitions a l'ecran */
                        printf("%.3f     %d     \n",(float)effortData[i].valeur,     effortData[i].temps);
                                }
/* création du fichier hop.data */
                        fp = fopen("/home/chimay/vxworks/duvel/hop.data", "w") ;
                        if (fp == NULL) {
                                fprintf(stderr, "Probleme d'ouverture du fichier <%s>\n",
                                "/home/chimay/vxworks/duvel/hop.data");
                                break;
                                }
                        for (i=0;i<nb;i++) {
/* écriture des données dans le fichier destination */
                        fprintf(fp, "%d %.3f\n",
                                effortData[i].temps,
                                (float)effortData[i].valeur);

                        }
/* fermeture du fichier ASCII */
fclose(fp);
printf("sequence acquise.\n");
} break;

case 'a': {
int voie;                       /* Numéro de la voie               */
printf("quelle voie?\n")
scanf("%d", &voie);             /* saisie du numéro de la voie     */
printf("channel %d : %f\n",
                        voie,
                        (float)(ipReadAnalog(voie)));   /* affichage du numero de la voie et de la
                valeur de l'axquisition */
} break;
```

```
case 'm': {
EFFORT_DATA_MULTI *effortDataMulti;   /* pointeur du tableau EFFORT_DATA  */
int j;                                /* numero de la voie                */
int nb;                               /* Nbre d'echantillons              */
int i;                                /* numero d'echantillon             */
int delta_t;                          /* Periode d'echantillonnage        */

/* creation des fichiers de stockage de donnees*/

FILE *fp1;                    /* Variable file pour stocker les donnees de la voie 4*/
FILE *fp2;                    /* Variable file pour stocker les donnees de la voie 5*/
FILE *fp3;                    /* Variable file pour stocker les donnees de la voie 6*/

printf("combien d'acquisition pour chaque voie ?\n");
scanf("%d", &nb);                   /* saisie du nombre d'acquisition */
printf("combien de temps (1/60sec) entre chaque acqui (0 aucun)?\n");
scanf("%d", &delta_t);              /* saisie de l'intervale de temps */
effortDataMulti = (EFFORT_DATA_MULTI *)malloc
(nb * sizeof(EFFORT_DATA_MULTI));
if (effortDataMulti == NULL) {
fprintf(stderr, "Probleme d'allocation memoire\n");
break;
}
tickSet(0);                   /* initialisation de l'horloge */
for (i=0;i<nb;i++) {
effortDataMulti[i].temps = tickGet();
printf("\n %d",effortDataMulti[i].temps);
for (j=0;j<=2;j++) {
/* on saisie les valeurs des voies 4 a 6 */
effortDataMulti[i].valeur[j] = ipReadAnalog(j+4);
if (delta_t != 0)
taskDelay(delta_t);
printf(" %.3f",  (2500/4.8)*(float)effortDataMulti[i].valeur[j]);
/* affichage des acquisitions a l'ecran sans l'offset des capteurs */
}
}
fp1 = fopen("/home/chimay/vxworks/duvel/force1.data", "w");
/* ouverture du fichier ASCII */
if (fp1 == NULL) {
fprintf(stderr, "Probleme d'ouverture du fichier <%s>\n",
                        "/home/chimay/vxworks/duvel/force1.data");
break;
}
for (i=0;i<nb;i++) {
fprintf(fp1, "%d  %.3f\n",
effortDataMulti[i].temps,
85.31+(2500/4.8)*(float)effortDataMulti[i].valeur[0]); /* calcul de la force
                                                          en Newtons avec prise
                                                          en compte de l'offset du capteur */
}
 fclose(fp1);
```

```
/* fermeture du fichier */
fp2 = fopen("/home/chimay/vxworks/duvel/force2.data", "w");
/* ouverture du fichier ASCII */
if (fp2 == NULL) {
fprintf(stderr, "Probl/eme d'ouverture du fichier <%s>\n",
                        "/home/chimay/vxworks/duvel/force2.data");
break;
}
for (i=0;i<nb;i++) {
fprintf(fp2, "%d  %.3f\n",effortDataMulti[i].temps,
                        162+(2500/4.8)*(float)effortDataMulti[i].valeur[1]);/* calcul de la force
                                                        en Newtons avec prise
                                                        en compte de l'offset du capteur */
}
fclose(fp2);
/* fermeture du fichier */

fp3 = fopen("/home/chimay/vxworks/duvel/force3.data", "w");
/* ouverture du fichier ASCII */
if (fp3 == NULL) {
fprintf(stderr, "Probleme d'ouverture du fichier <%s>\n",
              "/home/chimay/vxworks/duvel/force3.data");
break;
}
for (i=0;i<nb;i++) {
fprintf(fp3, "%d  %.3f\n",effortDataMulti[i].temps,
                        8.88+(2500/4.8)*(float)effortDataMulti[i].valeur[2]);/* calcul de la force
                                                        en Newtons avec prise
                                                        en compte de l'offset du capteur */
}
fclose(fp3);
/* fermeture du fichier */
printf("\n sequence acquise.");
} break;

case 's': {
EFFORT_DATA *effortData;      /* pointeur du tableau EFFORT_DATA      */
int nb;                       /* Nbre d'echantillons                  */
int i;                        /* numero d'echantillon                 */
int voie;                     /* Numero de la voie                    */
int delta_t;                  /* Periode d'echantillonnage            */
FILE *fp;
printf("quelle voie?\n");
scanf("%d", &voie);           /* saisie du numero de la voie          */
printf("combien d'acquisition?\n");
scanf("%d", &nb);             /* saisie du nombre d'echantillons      */
printf("combien de temps (1/60sec) entre chaque acqui (0 aucun)?\n");
scanf("%d", &delta_t);
effortData = (EFFORT_DATA *)malloc(nb * sizeof(EFFORT_DATA));
if (effortData == NULL) {
fprintf(stderr, "Probleme d'allocation memoire\n");
```

```
break;
}
tickSet(0);                          /* initialisation de l'horloge                    */
for (i=0;i<nb;i++) {
effortData[i].valeur = ipReadAnalog(voie);   /* la valeur effortData recupere la valeur
                                                                                du CAN */
effortData[i].temps = tickGet();         /* saisie du numero d'echantillon */
if (delta_t != 0)
taskDelay(delta_t);
printf("%.3f %d\n", (float)effortData[i].valeur, effortData[i].temps);                  /* affichage des
valeurs a l'ecran */
}

            fp = fopen("/home/chimay/vxworks/duvel/mesure.data", "w");
            if (fp == NULL) {
              fprintf(stderr, "Probleme d'ouverture du fichier <%s>\n",
                        "/home/chimay/vxworks/duvel/mesure.data");
              break;
            }
            for (i=0;i<nb;i++) {
              fprintf(fp, "%d %.3f\n",
                        effortData[i].temps,(float)effortData[i].valeur);
            }
            fclose(fp);
            printf("sequence acquise.\n");
    } break;

    case 'r': {

            int NBacqui = 0;
            int i;                  /* numero d'echantillon                                  */
            FILE *fp1;              /* Variable file pour stocker les donnees de la voie 4   */
            FILE *fp2;              /* Variable file pour stocker les donnees de la voie 5   */
            FILE *fp3;              /* Variable file pour stocker les donnees de la voie 6   */

            /* choix entre acquisition sur une voie ou plusieurs */
            printf("\nsingle(0) ou multi(1) ?\n");
            scanf("%d", &acquiMulti);



            /* Init l'acquisition */
            if (InitAcqui(NBacqui)==ERROR){
              printf("Pb d'horloge !!!\n");
              break;
            }

            /* Lance l'acquisition */
            sysAuxClkEnable();

            /* Attend la fin de l'acqui */
```

```
while(EndAcqui==0){
  taskDelay(2);
}

/* Arrete l'horloge */
sysAuxClkDisable();
printf("End Acqui \n");

if (acquiMulti == 0) {
  fp1 = fopen("/home/chimay/vxworks/duvel/ForceRapide.data", "w");
  if (fp1 == NULL) {
    fprintf(stderr, "Probleme d'ouverture du fichier <%s>\n",
                    "/home/chimay/vxworks/duvel/ForceRapide.data");
    break;
  }

  /*printf("End Acqui 1\n");*/
  for (i=0;i<SampleAcqui;i++) {
    if (acquiMulti == 0) {
      fprintf(fp1, "%d %.3f\n",
                     effortData[i].temps,(float)effortData[i].valeur);
    }
  }
}



if (acquiMulti != 0) {
  printf("\nMULTI\n");

  fp1 = fopen("/home/chimay/vxworks/duvel/ForceRapide1.data", "w");
  /* ouverture des fichiers textes */
  if (fp1 == NULL) {
    fprintf(stderr, "Probleme d'ouverture du fichier <%s>\n",
                    "/home/chimay/vxworks/duvel/ForceRapide1.data");
    break;
  }
  for (i=0;i<SampleAcqui;i++) {
    fprintf(fp1, "%d  %.3f\n",
                    i,
                    20+(2500/4.8)*(float)effortDataMulti[i].valeur[0]);
  }
  fclose(fp1);

  fp2 = fopen("/home/chimay/vxworks/duvel/ForceRapide2.data", "w");
  if (fp2 == NULL) {
    fprintf(stderr, "Probleme d'ouverture du fichier <%s>\n",
                    "/home/chimay/vxworks/duvel/ForceRapide2.data");
    break;
  }
  /*printf("End Acqui 1\n");*/
```

```
                for (i=0;i<SampleAcqui;i++) {
                  fprintf(fp2, "%d %.3f\n",
                                i,
                                92+(2500/4.8)*(float)effortDataMulti[i].valeur[1]);

                }
                fclose(fp2);
                fp3 = fopen("/home/chimay/vxworks/duvel/ForceRapide3.data", "w");
                if (fp3 == NULL) {
                  fprintf(stderr, "Probleme d'ouverture du fichier <%s>\n",
                                "/home/chimay/vxworks/duvel/ForceRapide3.data");
                  break;
                }
                /*printf("End Acqui 1\n");*/
                for (i=0;i<SampleAcqui;i++) {
                  fprintf(fp3, "%d %.3f\n",
                                i,
                                11+(2500/4.8)*(float)effortDataMulti[i].valeur[2]);

                }
                fclose(fp3);
              }
              printf("End Acqui 2\n");
              printf("sequence acquise.\n");
      } break;
      }
    }
  }
  fprintf(stdout, " fin de effortMenu.\n");

  }



/* Fct d'init de l'it          */
int InitAcqui()
{
  int status=OK;
  /* Connecte l'Horloge a` la fonction */
  sysAuxClkRateSet(HL_ACQUI);
  sysAuxClkConnect(SynchroAcqui,0);
  SemHl=semBCreate(SEM_Q_PRIORITY,SEM_EMPTY);
  EndAcqui = 0;
  SampleAcqui = 0;
  if (taskSpawn("tAcqui",5,VX_FP_TASK|VX_STDIO,
                          10000, (FUNCPTR) Acqui,
                          0,0,0,0,0,0,0,0,0,0)==ERROR)
    status = ERROR;
  else {
    taskDelay(10);
    printf("Acquisition ");
```

```
    status = OK;
  }

  return (status);
}

/* Fct d'it pour l'acqui         */
int SynchroAcqui()
{
  semGive(SemHl);
  if (SampleAcqui<MAX_ACQUI)
    SampleAcqui++;
  else
    EndAcqui=1;
  return OK;
};

/* Fct d'acqui synchronise       */
int Acqui()
{

  while(EndAcqui==0){
    int NumEchantillon = 1;
    semTake(SemHl,WAIT_FOREVER);
    if ((SampleAcqui%HL_ACQUI)==0)
      printf("%d ",SampleAcqui);
    if (acquiMulti == 0) {
      effortData[SampleAcqui].valeur = ipReadAnalog(4);
      effortData[SampleAcqui].temps = SampleAcqui;
    } else {
      /* les voies d'acquisitions sont les voies 4 a 6 */

      NumEchantillon++;
      effortDataMulti[SampleAcqui].valeur[0] = ipReadAnalog(4);
      effortDataMulti[SampleAcqui].valeur[1] = ipReadAnalog(5);
      effortDataMulti[SampleAcqui].valeur[2] = ipReadAnalog(6);
      effortDataMulti[SampleAcqui].temps = NumEchantillon ;

    }

  }
  printf("\n");
  return OK;
};
```

# ANNEXE 2

## Sources du programme utilisé pour le traitement des données.

```
%********************************************************************%
%                                                                    %
%  File           :  Open_file_XY.m                                  %
%  Specifications  :  1. Open files (ascii) for X and Y              %
%                     2. Plot the datas in the current window        %
%                                                                    %
%                     INRIA Rhone-Alpes                              %
%  Author          :  Espiau Francois-Xavier                         %
%  Date            :  July 1996                                      %
%                                                                    %
%********************************************************************%

k=1;

nomfich = '';
nomreper = '';

fprintf(1,'\n=======================================================\n');
fprintf(1,'If the two files have not the same lenght, we take the\n');
fprintf(1,'minus between them and we cut the largest one.\n\n');

if k==1,

  [nomfich,nomreper] = uigetfile('*','Open file for both X and Y',140,190);

  if nomfich~=0,

    eval(['load ',nomfich,' -ascii']);
    i=1;
    while nomfich(i)~='.', tmp(i)=nomfich(i);i=i+1;end;
    clear i;
    eval(['X=',tmp,'(:,1);']);
    eval(['Y=',tmp,'(:,2);']);
    clear tmp;

    fprintf(1,'The array X contains the first column of the file %s\n',nomfich);
    fprintf(1,'and the array Y the second one.\n');
    fprintf(1,'=======================================================\n\n');

    clear nomfich;
    clear nomreper;
  end;

elseif k==2,

  [nomfich,nomreper] = uigetfile('*','Open file for X',140,190);

  if nomfich~=0,

    eval(['cd ',nomreper]);
    fid = fopen(nomfich,'r');
```

```
    [X,count] = fscanf(fid,'%f',inf);
    fclose(fid);

    fprintf(1,'The array X contains the data loaded from the file %s\n',nomfich);

    clear nomfich;
    clear nomreper;
    clear count;
  end;

  [nomfich,nomreper] = uigetfile('*','Open file for Y',140,190);

  if nomfich~=0,

    eval(['cd ',nomreper]);
    fid = fopen(nomfich,'r');
    [Y,count] = fscanf(fid,'%f',inf);
    fclose(fid);

    fprintf(1,'The array Y contains the data loaded from the file %s\n',nomfich);
    fprintf(1,'=======================================================\n\n');

    clear nomfich;
    clear nomreper;
    clear count;
  end;
end;

%-------------------------------------
% X and Y must have the same lenght if
% we want to plot (X,Y)
% So, we use the minus between size(X)
% and size(Y). And we cut the biggest
% signal to the size we've found
%-------------------------------------
min_xy = min(size(X),size(Y));
for i=1:min_xy,
  X_tmp(i,1) = X(i);
  Y_tmp(i,1) = Y(i);
end;

clear X;
clear Y;
X = X_tmp;
Y = Y_tmp;
clear X_tmp;
clear Y_tmp;
clear k;

clf;
plot(X,Y);
```

# ANNEXE 3

Documentations des Capteurs .