École Polytechnique de l'Université de Tours
64, Avenue Jean Portalis
37200 TOURS, FRANCE
Tél. +33 (0)2 47 36 14 14
www.polytech.univ-tours.fr

Département Informatique
4e année
2011 - 2012

Projet d'Ingénierie du Logiciel

Rapport - Robot humanoïde NAO

Encadrant
Pierre GAUCHER
pierre.gaucher@univ-tours.fr

Université François-Rabelais, Tours

Étudiants
Joachim ALIBERT
joachim.alibert@etu.univ-tours.fr
Rémi CARUYER
remi.caruyer@etu.univ-tours.fr
Guillaume GEDEON
guillaume.gedeon@etu.univ-tours.fr
Chunhui LI
chunhui.li@etu.univ-tours.fr
Wei GONG
wei.gong@etu.univ-tours.fr
Benjamin PASQUET
benjamin.pasquet@etu.univ-tours.fr
Sébastien SCHAAL
sebastien.schaal@etu.univ-tours.fr
Cedric VERNOU
cedric.vernou@etu.univ-tours.fr

DI4 2011 - 2012

Version du 6 juin 2012

Table des matières

Table des figures

Introduction

Dans le cadre de notre formation au sein du Département Informatique de l'école Polytechnique de l'Université de Tours, un projet d'Ingénierie du Logiciel est effectué par groupe de huit afin de mettre en pratique nos connaissances acquises au cours de notre formation.

Nous tenons à remercier notre encadrant Monsieur Pierre GAUCHER pour ses conseils pertinents qui nous ont été d'un appui considérable pour accomplir notre projet.

Notre rapport est constitué de trois parties. La première est une présentation du projet et de ses objectifs. La deuxième partie présente le travail effectué. Enfin, la dernière partie détaille les outils et méthodes utilisés.

Première partie

Cahier des charges

1. Présentation de la société.

Aldebaran Robotics est une start-up dont le siège social est situé à Paris. La société est aujourd'hui considérée comme le leader mondial dans le domaine de la robotique humanoïde, notamment dans la sphère professionnelle.

En effet, son premier robot équiperait selon la société 480 universités ou écoles dans le cadre d'applications de recherches ou pédagogiques.

FIGURE 1.1 – Logo Aldebaran Robotics

La société a présenté le robot NAO pour la première fois au public fin 2006. Depuis, 6 prototypes ont été développés.

2. Présentation de NAO

NAO est un robot humanoïde contenant de multiples capteurs. Il est bien entendu programmable. Il mesure environ 58 cm et pèse moins de 5 kg.

NAO est équipée d'une centrale inertielle avec un accéléromètre 3 axes et 2 gyromètres , de 2 sonars utilisants des capteurs à ultrason (émetteurs et récepteurs), de 8 capteurs de pressions résistifs sous les pieds et de 2 bumpers.

FIGURE 2.1 – Robot NAO

Il dispose également d'un système multimédia évolué incluant quatre microphones (pour la reconnaissance vocale et la localisation de la source sonore), deux haut-parleurs (pour la synthèse vocale),et deux caméras HD (1280 x960) (pour la localisation ou la reconnaissance de visage ou d'objet). Ces 2 caméras positionnées verticalement, lui permettent de voir une personne de 1m80 de la tête au pied à 1,5 m.

Il a aussi des capteurs d'interactions tels que des zones tactiles sur le dessus de la tête et sur les mains, deux LED infrarouges ainsi que deux bumper sur l'avant des pieds.

On peut voir ci-dessous un schéma représentant les différents composants du robot.

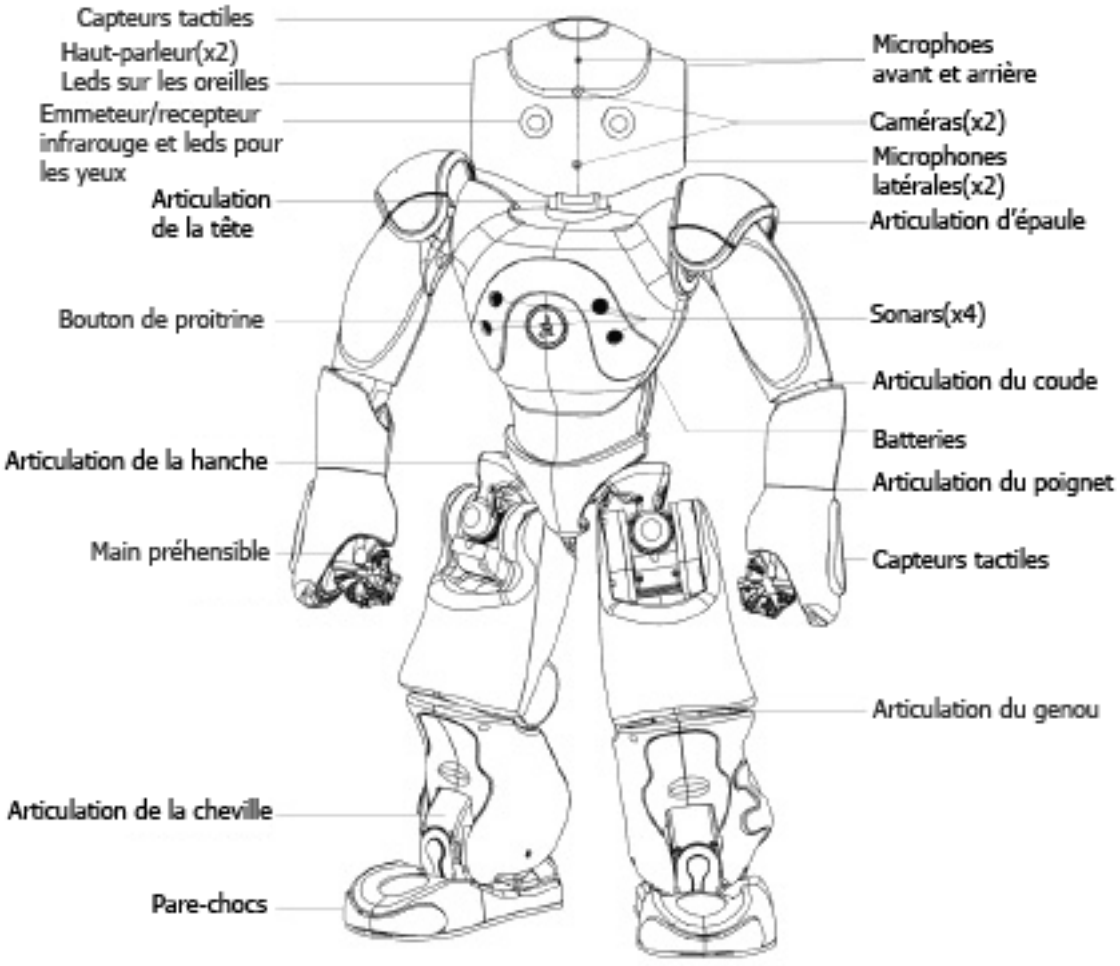

FIGURE 2.2 – NAO

3. Objectif du projet

Nos objectifs se décomposent en 4 parties principales.

3.1 S'approprier le robot NAO

Selon le schéma présenté dans la présentation de NAO, le robot est composé de multiples composants. Nous devons donc d'abord nous approprier et savoir utiliser tous ces composants. Ensuite, nous avons appris à utiliser le logiciel Choregraphe.

Pour apprendre à utiliser NAO, nous avons plusieurs choix qui s'offrent à nous :
- Choregraphe est un outil fourni par défaut avec NAO. C'est un logiciel assez simple à prendre en main qui propose une IHM conviviale et quelques blocs prédéfinis.
- Nous pouvons également coder directement en Python. Tous les blocs dans Choregraphe sont écrit dans ce langage.
- Nous pouvons aussi travailler en langage C++ ou JAVA.

La première partie du projet consiste donc à un auto-apprentissage et à une première manipulation de NAO et de tous ses composants.

3.2 Réalisation de tâches simples

Après avoir testé la plupart des composants, nous devons réaliser des tâches simples. Ces tâches doivent pouvoir nous aider à approfondir nos connaissances en programmation sur Choregraphe.

Ces dernières seront ensuite utiles pour réaliser la tâche principale.

Les tâches simples sont par exemple :

- Reconnaissance du visage et action suivante
- Interaction homme-machine (reconnaissance vocale et répondre)

3.3 Création d'un guide utilisateur

Un des buts de ce projet est de réaliser un Manuel Utilisateur.

Le Manuel Utilisateur contiendra les trois parties suivantes :
- Prise en main du robot et configuration de l'environnement de développement
- Manipulation du robot
- Précaution à prendre lors de l'utilisation du robot

C'est un manuel qui permet à un nouvel utilisateur de prendre en main rapidement Nao, mais aussi d'éviter les problèmes que nous avons rencontrés.

3.4 Réalisation d'une tâche spécifique

Enfin, le projet devra aboutir à la création d'une tâche qui sera à définir.

Le but est de montrer ce qu'il est possible de faire avec le robot grâce à l'utilisation du logiciel Choregraphe.

4. Contraintes

4.1 Contraintes de délais

- Début du projet : lundi 23 janvier 2012
- Fin du projet : lundi 4 juin 2012

Nous avons réalisé le diagramme de Gantt ci-dessous qui définit les différentes étapes du projet à réaliser.

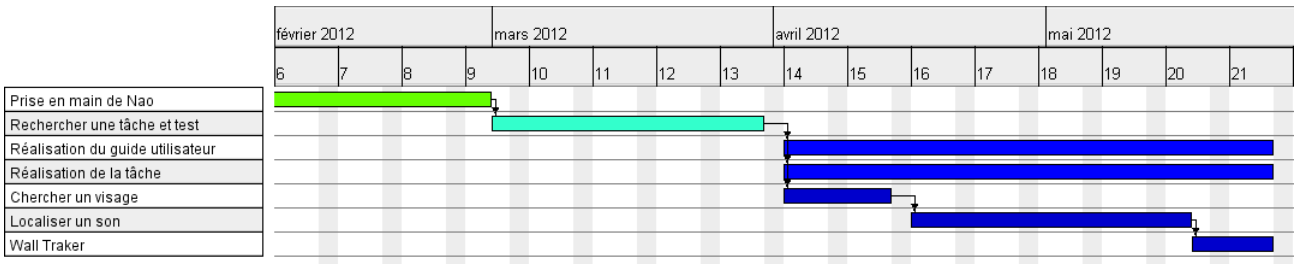

FIGURE 4.1 – Diagramme de Gantt

4.2 Contraintes techniques

Les contraintes techniques sont liées à la spécificité du projet. En effet, nous travaillons sur un robot totalement inconnu de tous les membres du groupe.

Il en est de même au niveau du logiciel de programmation du robot, Chorégraphe.

4.3 Contraintes de réalisation

Nous n'avons pas de contraintes de réalisations spécifiques. En effet, le choix du langage et des différents outils de programmation restent totalement libre.

Deuxième partie

Travail effectué

5. Apprentissage et premières manipulations

5.1 Réunion de prise en main

Lorsque le robot Nao a été mis a notre disposition, nous avons profité d'un créneau de 2heures pendant lesquelles Mickael Rousseau, ayant bénéficié d'une formation à l'utilisation du robot, nous a expliqué les bases de l'utilisation du robot et surtout ce qu'il ne fallait PAS faire avec, pour des raisons de sécurité.

Monsieur Rousseau nous a expliqué de quelle manière se connecter à NAO via ethernet, puis nous a fait une présentation du logiciel de programmation Chorégraphe et a terminé par quelques manipulations de base.

Le but de cette séance a été avant tout de nous sensibiliser sur les erreurs à ne pas faire pour préserver le robot.

Un exemple de ce que pouvait nous expliquer Monsieur Rousseau est :

- Lors de l'utilisation du robot via le câble Ethernet, il faut faire attention au mouvement de NAO, car en cas de chute en arrière le câble pourrait s'enfoncer dans « la tête » du robot.

5.2 Auto-formation

Suite à cela, nous sommes passé par une phase d'auto-apprentissage durant laquelle nous allions nous familiariser avec le logiciel fourni et les fonctionnalités du robot. Nous avons tout d'abord commencé à travailler avec une connexion filaire au robot, ce qui ne nous permettait de connecter qu'un seul ordinateur à la fois à NAO et donc un seul poste de travail pouvait fonctionner à la fois. Afin de palier a cette contrainte, nous avons utilisé une connexion via Wi-Fi permettant de nous connecter à plusieurs sur le robot.

Une fois que tout le monde a eu accès au robot et au logiciel Chorégraphe, nous avons commencé une série de tests afin de connaître et comprendre les différentes fonctions offertes par ce logiciel et comment s'en servir. Il nous a fallut étudier les limites et possibilités de NAO pour bien maîtriser ses mouvements. Cette phase d'auto-apprentissage nous a permis à la fois de savoir comment obtenir les résultats que l'on attendait pour le robot, mais aussi de connaître ce qui ne marche pas ou marche mal.

Cette phase nous a également aidé a rédiger le guide d'utilisation du robot car nous avons pu nous mettre à la place des personnes qui pourraient être amenées a lire ce guide : les nouveaux utilisateurs.

5.3 Rédaction du Manuel Utilisateur

Une importante partie du travail à effectuer lors de ce projet a été la réalisation du *Manuel Utilisateur*.

Pour réaliser ce dernier, nous avons utilisé différentes méthodes :

- Auto-apprentissage (cf. section Auto-formation).
- Lecture de la documentation officielle en anglais (http ://developer.aldebaran-robotics.com/doc/1-12/)
- Lecture de forums officiels ou non

Ces méthodes combinés ont donné un résultat positif puisque nous avons réussi à expliquer les différents points importants de manière simple et compréhensible par tous.

Ainsi, en lisant notre Manuel Utilisateur, chacun pourra apprendre et comprendre des choses simples comme *comment connecter Nao simplement à un ordinateur via le Wi-Fi*, ou plus complexes comme *comment créer des boîtes, les sauvegarder et les utiliser sous Choregraphe*.

6. Réalisation de la tâche

6.1 Réflexions préliminaires

Comme spécifié précédemment nous avions pour but de réaliser un guide utilisateur pour la prise en main du robot humanoïde NAO, et la réalisation d'une tache « complexe ». Cette tâche devant être définie collectivement et ayant pour but d'approfondir nos compétences acquises en terme de programmation de ce robot.

Dans un premier temps, suite à notre réunion de prise en main et de multiples lectures de documentation, sur internet notamment, nous avons cherché une tâche à réaliser.

Première orientation

Nous étions collégialement partie sur une tâche consistant à déposer NAO dans une zone délimitée, de lui montrer un objet, de « cacher » cet objet quelque part dans la zone, et nous aurions voulu que NAO retrouve l'objet et le ramène.

Nous nous sommes vite rendu compte que la réalisation de cette tâche serait très compliqué, voir irréalisable à cause des limites de nos connaissances, de la documentation du logiciel de programmation Choregraphe et des limites de ce logiciel.

Seconde orientation

Nous nous sommes donc restreint, et nous avons réorienté la tâche à réaliser par NAO. Il devra donc reconnaitre l'appel de son nom, chercher la personne qui l'a appelé et venir vers cette personne. Vous verrez dans la suite de notre rapport que même si cette tache semble « simple » elle nécessite déjà une bonne connaissance et maitrise de la programmation de NAO.

6.2 Explication détaillée de la tâche choisie

Avant tout il nous fallait déterminer des sous tâches composant notre tâche principale :

6.2.1 Détection et localisation de l'appel

Description

L'objectif ici est de déterminer la position d'un appel reconnu par NAO. Une personne utilise une formulation d'appel reconnu par NAO, le robot renvoie les coordonnées de l'appel relatives à la position de NAO.

Solution

Dans la librairie par défaut de Choregraphe, nous avons à notre disposition des box pouvant correspondre à la problématique posée :

– *Speech Reco.* : Elle prend en paramètre une liste de mots qui est traité grâce au *Process Word* qu'elle contient. Elle possède deux sorties *wordRecognized* (le mot reconnu) et *onNothing* (lorsque aucun mot n'est reconnu).

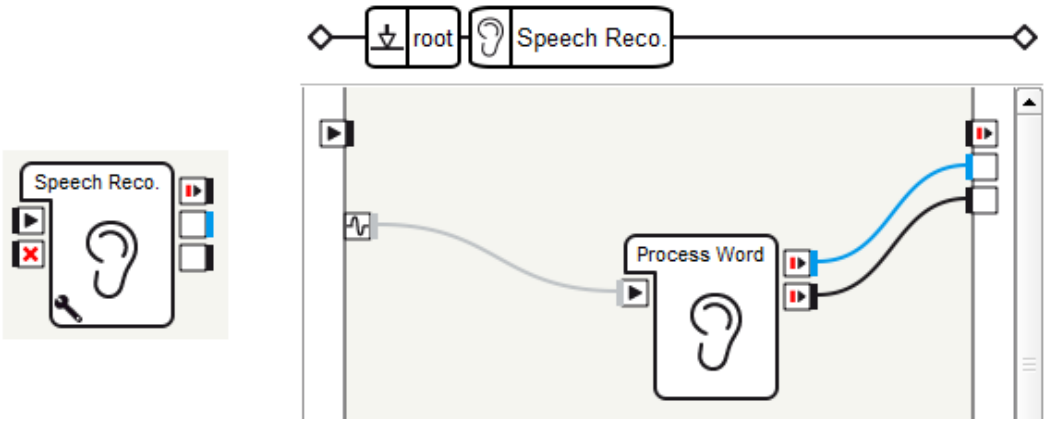

FIGURE 6.1 – Speech Reco.

– *Sound Loc.* : Elle contient *ProcessSoundLoc* qui renvoie deux tableaux *sourceLocation* (deux angles correspondant à la position de la source sonore relative à la tête de NAO) et *headPosition* (la position de la tête).

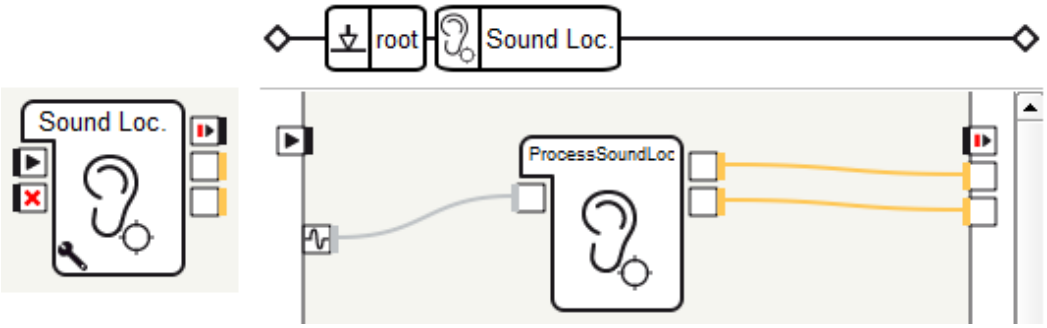

FIGURE 6.2 – Sound Loc.

Après une suite de test, nous nous sommes aperçu qu'il était extrêmement difficile de faire correspondre le mot reconnu par NAO avec le son qu'il localise. Nous avons donc rechercher à résoudre se problème en fusionnant les deux box citées plus haut. Pour cela nous avons créé une nouvelle box :

Reco and Locate

Pour réaliser cette fusion, nous sommes parti sur la base de la box *Speech Reco.*. Nous y avons intégré le *ProcessSoundLoc* au même niveau que le *Process Word*. *Reco and Locate* renvoie donc la valeur, uniquement, de l'angle horizontal correspondant à la position du son reconnu.

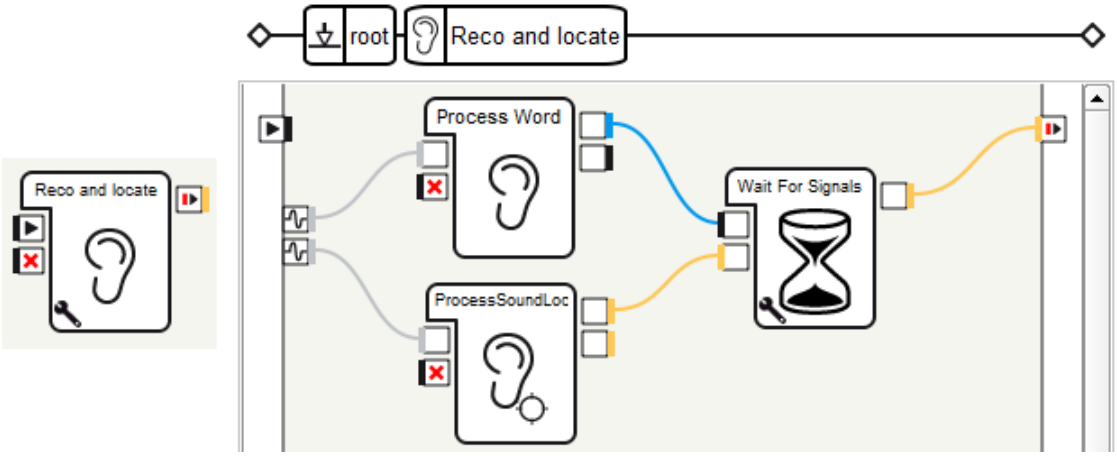

FIGURE 6.3 – Reco and Locate

Pour que le *ProcessSoundLoc* puisse fonctionner correctement, nous avons ajouté ses paramètres à la nouvelle box *Reco and Locate*. Voici la liste de mots reconnus par NAO : viens ici ; nao viens ici ; viens ici nao ; nao ici ; ici nao.

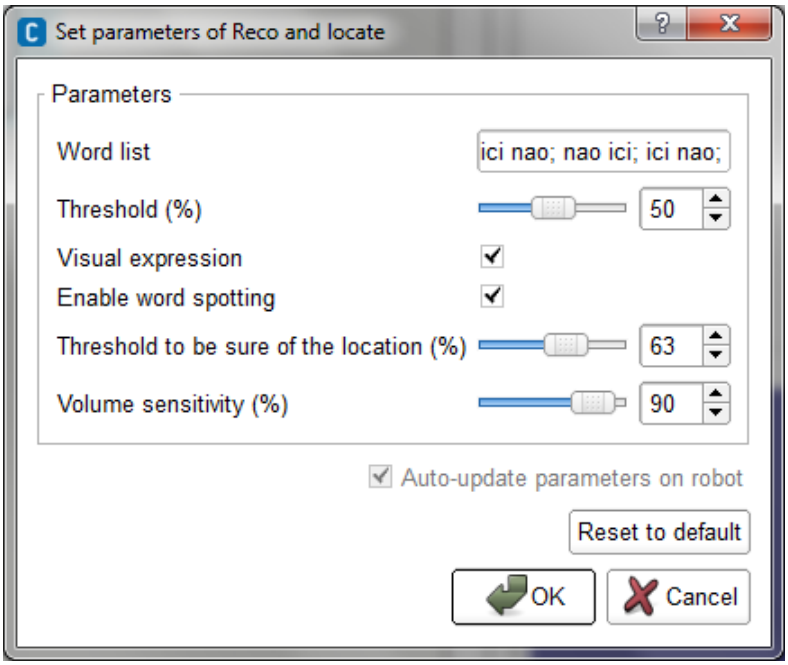

FIGURE 6.4 – Les paramètres de la box Reco and Locate

De plus, pour ne pas renvoyer le tableau n'importe quand, nous avons ajouté un *Wait*. Nous l'avons modifié pour qu'il prenne en paramètre deux signaux :

– Le signal envoyé par *Process Word* lorsque un mot est reconnu.

– Un tableau de deux Number contenant les deux angles correspondant à la position du son.

Nous avons dû modifier le script Python du *Wait* de base pour qu'il ne renvoie que le premier Number contenu dans le tableau d'entrée.

Wait For Signals

```
class MyClass(GeneratedClass):
    def __init__(self):
        GeneratedClass.__init__(self)

    def onLoad(self):
        self.ok = [False]*2

    def onUnload(self):
        #puts code for box cleanup here
        """

    def onStart(self, nInput):
        self.ok[nInput-1] = True
        bOutput = True
        for bOk in self.ok:
            bOutput = bOutput and bOk
        if( bOutput ):
            self.ok = [False]*2
            self.signalsReceived(self.getParameter("parameter"))

    def onInput_signal1(self):
        self.onStart(1)

    def onInput_signal2(self,p):
        self.setParameter("parameter",p[0])
        self.onStart(2)
```

FIGURE 6.5 – Wait de la box Reco and Locate

6.2.2 Recherche de la personne appelante

Description

Une fois la position de la personne appelante trouvée, nous voulons que NAO l'ait dans son champ de vision. Il doit donc pouvoir se tourner dans la direction de cible pour pouvoir établir un contact visuel.

Solution

Le robot entend et reconnait un appel. Il possède maintenant un angle correspondant à la position de la personne appelante, il doit donc l'utiliser et se tourner sur celui-ci. Ensuite, il lance une recherche du visage de la personne. Pour cela nous avons mis en place une box :

Turn and Search

Elle prend en paramètre d'entrée l'angle, puis se tourne dans cette direction grâce à la box *Turn Around*, que nous avons mis en place. Une fois que NAO se retrouve face à la personne, il recherche un visage grâce à *Face Spoting Slow*.

Turn and Search possède deux sorties :
- *onFaceSpoted* : Un visage est découvert
- *onNoFaceSpoted* : Aucun visage n'est trouvé

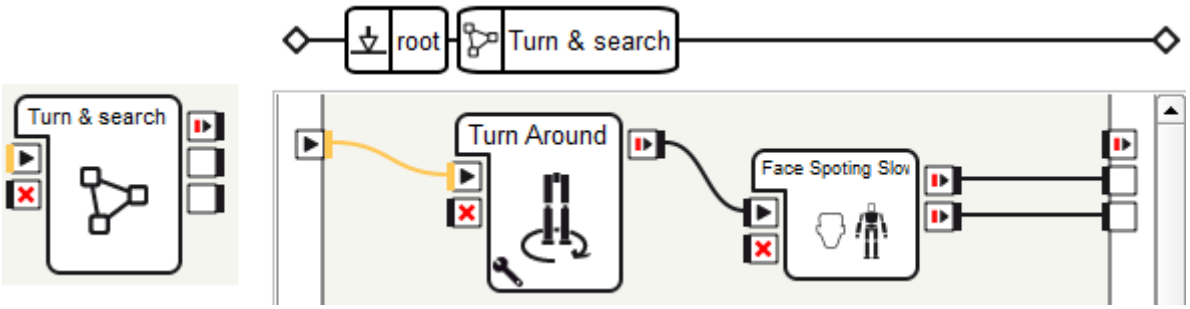

FIGURE 6.6 – La box Turn and Search

Turn Around

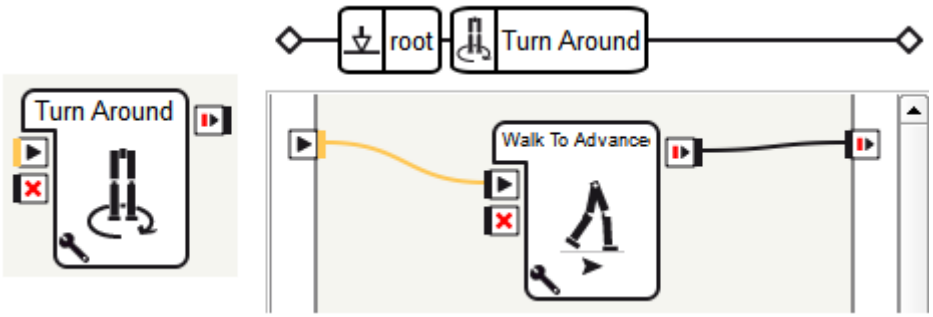

FIGURE 6.7 – La box Turn Around

Pour nous permettre de faire tourner NAO, nous utilisons la box *Walk To Advance* présente dans *Walk To*, qui permet de définir un déplacement du robot. Ici nous voulons uniquement faire tourner NAO. Nous utilisons l'héritage des paramètres des sous-box pour transférer l'angle de *Turn Around* à *Walk To Advance*. En entrée, elle récupère l'angle et le rentre comme valeur de son paramètre *Theta (rad)*, pour ce faire nous avons dû changer le script de *Turn Around*.

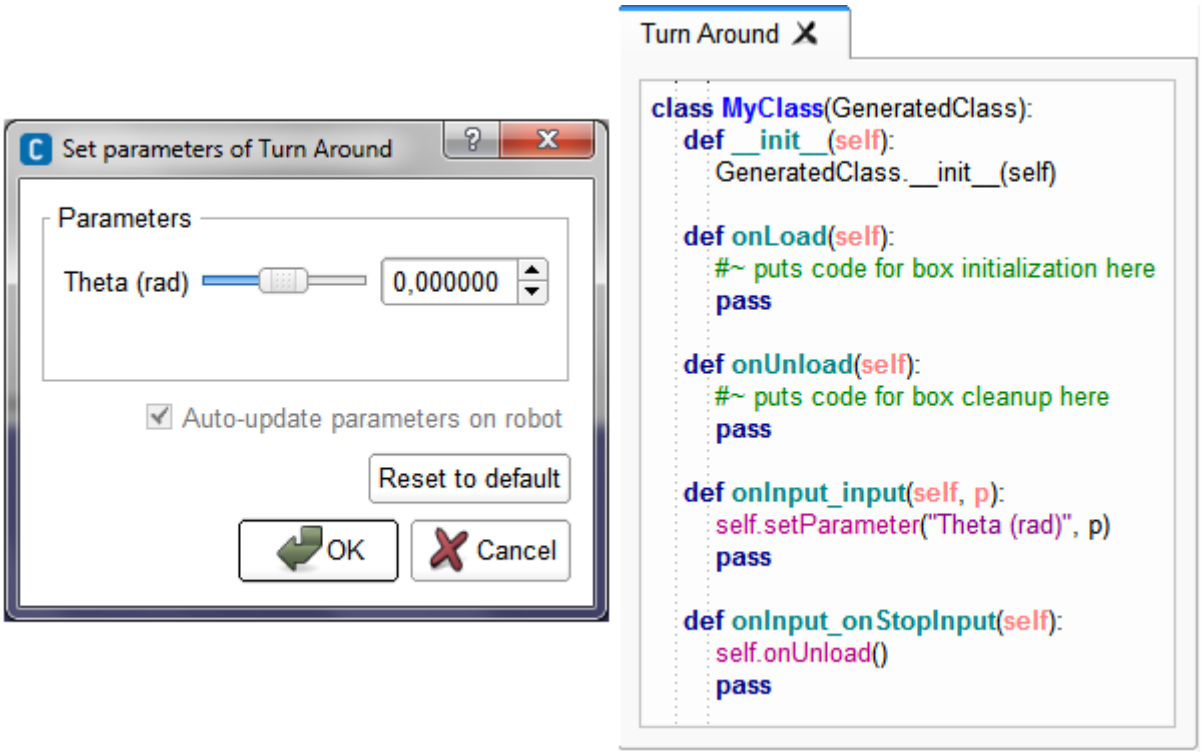

FIGURE 6.8 – Les paramètres et le script de la box Turn Around

Face Spoting Slow

Ici, nous avons pour objectif de trouver un visage dans le champs de vision de NAO. Pour cela, il existe une box (*Face Detection*) qui renvoie un signal quand un visage apparait dans le champs de vision. Ce signal contient le nombre de visage présent. Un autre signal est envoyé lorsqu'il n'y a plus de visage présent dans le champs de vision.

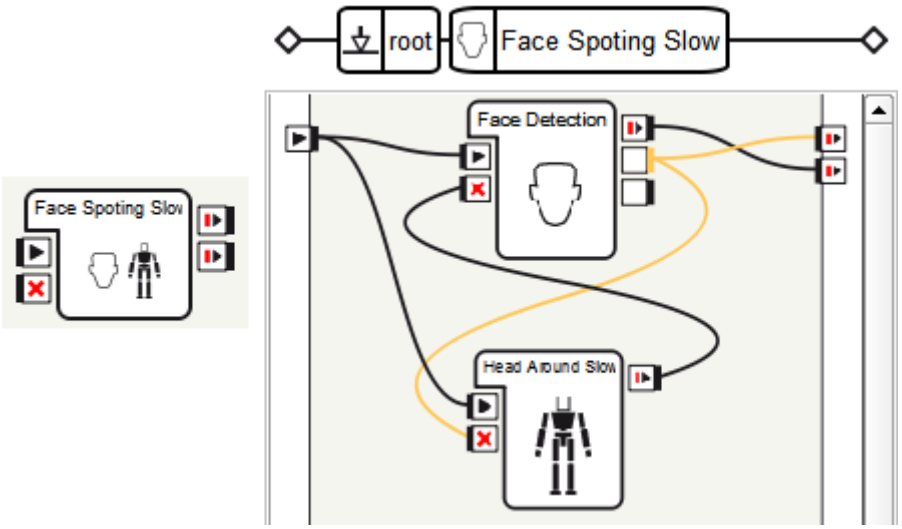

FIGURE 6.9 – La box Face Spoting Slow

Pour maximiser les chances de perception, et palier à des erreurs lors de l'acquisition de l'angle, nous avons mis en place un mouvement fluide et lent, permettant à NAO de faire tourner sa tête tout autour de lui à la recherche d'un visage. Ce mouvement est exécuté dans la box *Head Aroud Slow*.

Au signal d'entrée de *Face Spoting Slow* on lance le mouvement et la reconnaissance de visage. Le mouvement s'effectue et lorsque NAO détecte un visage, on stoppe le mouvement et on renvoie le signal de sortie correspondant. Dans le cas où le mouvement termine sans qu'un visage ne soit détecté, on arrête la détection et on signale que l'on a pas trouvé de visage.

6.2.3 Déplacement vers la personne

Description

Nous avons pour objectif de nous déplacer vers la personne appelante.

Solution

Il existe une box qui correspond totalement au problème posé. *Walk Tracker* utilise une cible comme objectif lors de la marche. Cette cible peut être un visage, c'est donc celle-ci que nous allons utiliser.

FIGURE 6.10 – La box Walk Tracker

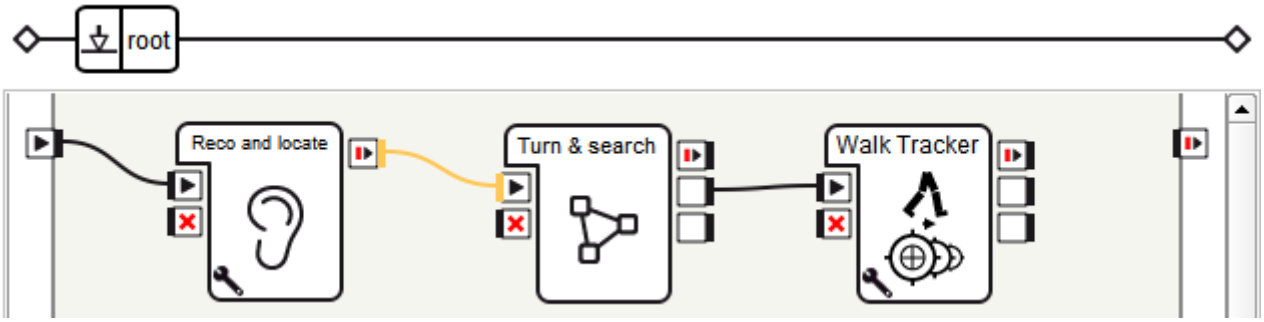

FIGURE 6.11 – Enchaînement des différentes tâches

Troisième partie

Outils et méthodes utilisés

7. Méthode Scrum

7.1 Explications de la méthode Scrum

Scrum est une méthode agile dédiée à la gestion de projets. Son objectif est d'améliorer la productivité des équipes auparavant ralenties par des méthodologies plus lourdes.

7.1.1 La méthode en quatre points

Cette méthode de travail se suit en respectant un certain nombres de points. Voici les différents types de réunions que nous avons effectué pour respecter la méthode Scrum :

- Les réunions quotidiennes : chaque jour, toute l'équipe se réunit, généralement debout, pendant 15 minutes environ pour répondre aux 3 questions suivantes : qu'ai-je fait hier ?, Que vais-je faire aujourd'hui ? Y a-t-il un obstacle gênant aujourd'hui ?
- Les réunions de planifications : toute l'équipe se réunit pour décider des fonctionnalités qui vont composer le sprint suivant et mettre à jour la liste générale.
- Les réunions de revue de travail : lors de cette réunion, chacun présente ce qu'il a fait pendant la durée du sprint. Une démonstration des nouvelles fonctionnalités ou de présentation de l'architecture est organisée. Il s'agit d'une réunion informelle de 2 heures environ à laquelle participe toute l'équipe.
- Les réunions de rétrospectives : à chaque fin de sprint, l'équipe fait le point sur ce qui à bien fonctionné et sur ce qui a moins bien fonctionné. Lors de cette réunion d'une durée de 15 à 30 minutes où chacun est invité et parle en son nom, un vote de confiance est organisé pour décider des améliorations à apporter.

Ci-dessous, on peut voir un schéma expliquant assez bien le principe de la méthode.

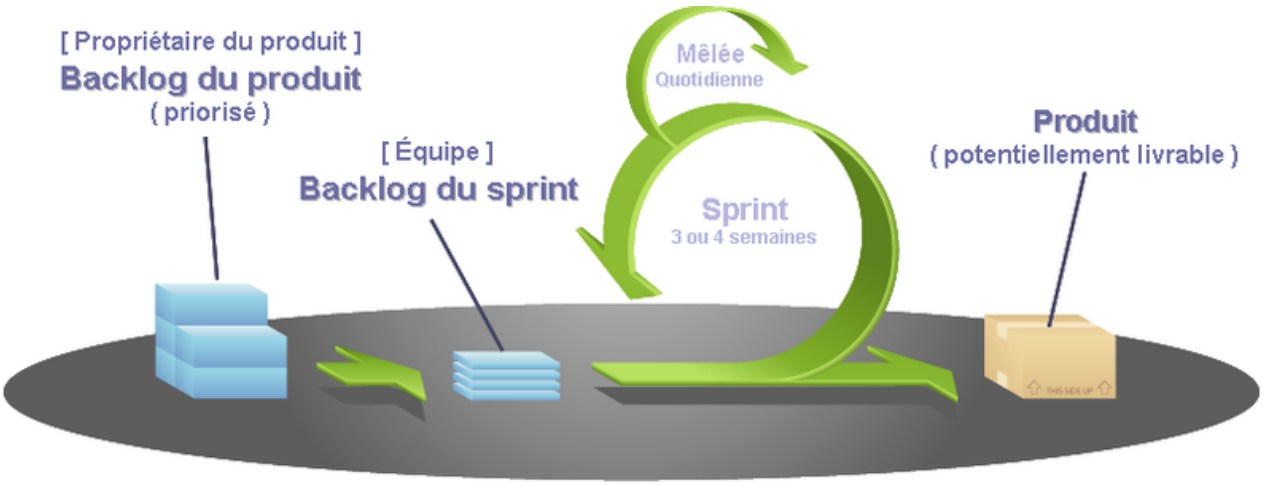

FIGURE 7.1 – Méthode Scrum

7.1.2 Les apports de cette méthode

Nous avons découvert cette méthode lors de ce projet et nous avons donc découvert parallèlement ses nombreuses qualités. Parmi celles-ci :

- Transparence sur l'avancement (résultats rapides)
- Adaptabilité : le client peut modifier son besoin
- Pas d'individualisme
- Recherche de la simplicité
- Pilotage au quotidien

L'apport de Scrum dans ce type d'équipe est donc une structuration relativement légère qui permet un recadrage en douceur.

7.2 Outils nécessaires

Nous avons décidé d'utiliser l'outil de gestion de projet en ligne earliz.com.

Ce site nous permet de définir les différents « Product backlog » avec ses tâches associées. Ces différentes tâches sont attribuées à un ou plusieurs membres de l'équipe et sont visibles dans le tableau scrum.

On peut voir sur le screen ci-dessous ce que cela donne.

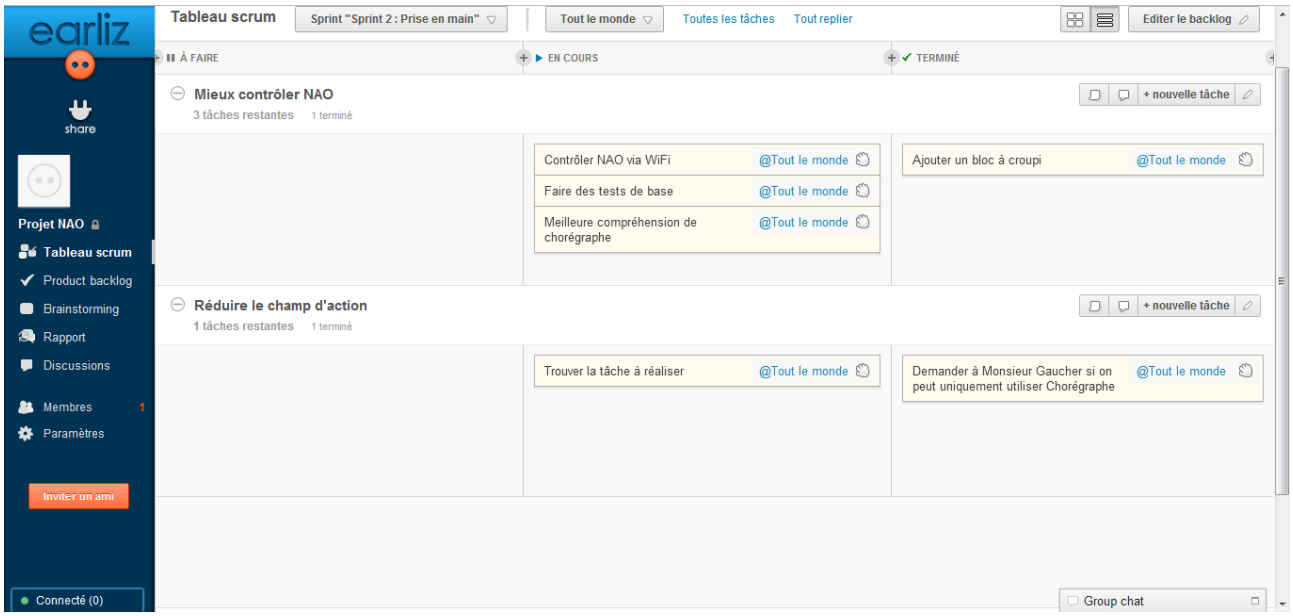

FIGURE 7.2 – Outil Earliz

8. Mise en commun du travail

8.1 Logiciel de gestion de versions

Nous utilisons un système de gestion de versions pour travailler plus efficacement durant ce projet. Ceci ne nous était pas imposé, mais il nous a semblé judicieux d'en utiliser un, pour gagner en temps et en efficacité.

Cela nous permet durant tout le projet de :
- Pouvoir revenir en arrière
- Voir qui a modifié un fichier en dernier

FIGURE 8.1 – Logo Subversion

Nous avons choisi Subversion (SVN). Ce choix s'explique par plusieurs critères :
- Documentations très riches, forums actifs
- Système de gestion de versions
- Interfaces graphiques sous Windows : intégré à l'explorateur via le plugin TortoiseSVN

Et nous avons également eu quelques heures d'initiation à SVN en troisième année.

Nous avons également décidé d'héberger notre espace de travail sur le site Assembla. Cela nous a permis de pouvoir transférer et récupérer notre travail de partout (Université, domicile...)

9. Logiciels utilisés

9.1 Choregraphe

9.1.1 Présentation de Choregraphe

Entièrement conçu et développé par Aldebaran Robotics, Chorégraphe est le logiciel de programmation qui permet aux utilisateurs de NAO de créer et d'éditer des mouvements et comportements interactifs.

Choregraphe dispose d'une interface graphique intuitive, sa bibliothèque de comportements standard et ses fonctions de programmation avancée.

Un des avantages de Chorégraphe est qu'il dispose d'exemples préconçus, ce qui aide énormément à la réalisation de nouveaux mouvements.

Chorégraphe accepte les langages Urbi et Python et peut donc directement appeler des modules C++ développés séparément.

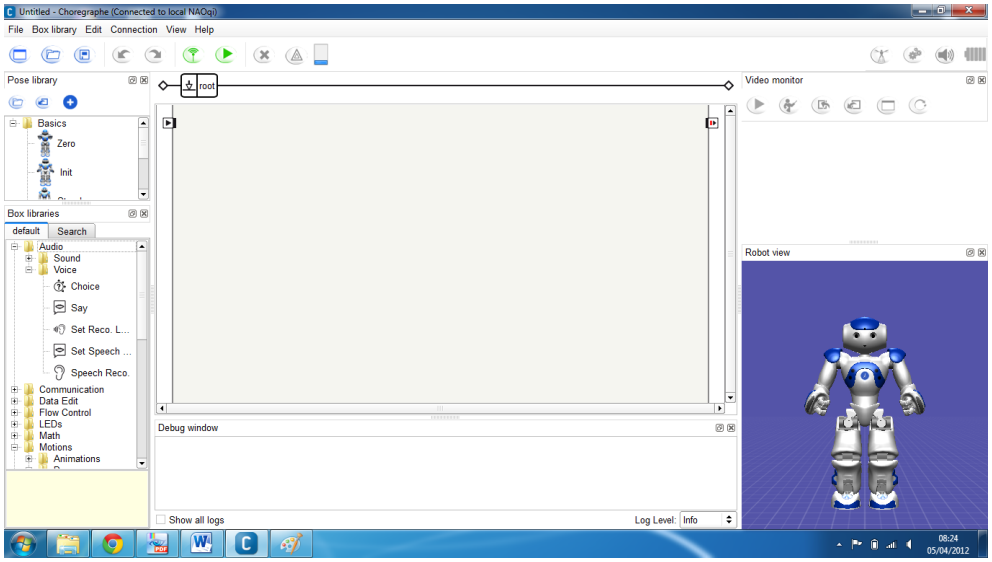

FIGURE 9.1 – Choregraphe

9.1.2 Utilisation de Chorégraphe

Durant notre projet, nous nous sommes concentré sur l'utilisation du logiciel Chorégraphe.

Le manuel utilisateur présente donc l'utilisation de NAO via ce logiciel.

Nous avons en très grande majorité travaillé avec l'interface graphique de Chorégraphe, mais il a parfois été nécessaire de toucher directement au code. Cela se faisait donc en Python. (Tous les blocs de Chorégraphe sont écrits en Python)

10. Problèmes rencontrés

Lors de notre étude sur Nao, nous avons été confrontés à plusieurs difficultés. Ces difficultés sont parfois liées à une mauvaise compréhension, d'autres fois lié à des problèmes matériels ou logiciels.

10.1 Connexion multiple à Nao

Il arrive que lorsque l'on se connecte à plusieurs à Nao, par l'intermédiaire de Choregraphe, des difficultés surviennent. Dans ce cas la communication est altérée, certains utilisateurs ne pourront pas obtenir l'image de la caméra etc...

Un autre problème peut survenir lors de la compilation. En effet, si deux personnes compilent en même temps, ou encore quand un utilisateur lance un programme et un autre essaie de compiler, alors un redémarrage de Nao est nécessaire. Cela est compréhensible ; c'est comme si deux compilateurs écrivaient dans un même fichier depuis des sources différentes.

10.2 Le Wi-Fi

Un routeur Wi-Fi (non fourni avec le robot) a été prêté par l'établissement. Nous l'avons utilisé pour connecter Nao à nos machines. Lorsque Nao était connecté à ce routeur, le robot avait des réactions non prévisibles. A plusieurs reprises, Nao indiquait par ses fonctions vocales qu'une erreur inquiétante s'était produite.

Une fois Nao déconnecté du routeur et redémarré, son fonctionnement redevient stable. Par la suite, nous avons réessayé d'utiliser le routeur et nous avons pu constater les mêmes problèmes.

Ce problème a été résolu, ou plutôt contourné en utilisant un point d'accès Wi-Fi créé à partir d'un smartphone Android. Étrangement, avec cette méthode, nous n'avions aucun problème de connexion.

10.3 La reconnaissance vocale

Une autre difficulté à laquelle nous avons été confronté est la reconnaissance vocale de Nao. Cette dernière comprend deux fonctionnalités : "la reconnaissance de mot" et "la localisation de la provenance du son".

La reconnaissance de mot et la localisation de la provenance du son, pour fonctionner correctement, nécessitent le calme dans les alentours. Lors de l'exécution d'un programme utilisant la reconnaissance vocale, si d'autres bruits sont produit dans la pièce, cela fausse l'analyse de Nao.

Parfois, Nao localise mal la provenance du son. En effet, nous avons beaucoup de difficulté à synchroniser la localisation avec la reconnaissance du mot. Nao nous renvoie donc la localisation de l'avant dernier son au lieu du dernier.

De même si Nao effectue une action qui provoque du bruit (marcher, s'asseoir, parler...), il entend son propre bruit et cela peut fausser la reconnaissance vocale.

Pour palier à cela, il est possible de régler la sensibilité de reconnaissance de Nao. Lors de la détection d'un son, une valeur de fiabilité est attribué pour chaque mot. Si cette valeur est supérieure à la sensibilité, alors le mot est reconnu.

Le problème est qu'il est très difficile de trouver la bonne valeur de sensibilité.

Conclusion

Ce projet nous a été bénéfique sur différents points, puisqu'il nous a permis de travailler en groupe de huit personnes et que nous avons pu découvrir un système nouveau : le robot NAO.

L'étude de ce robot nous a permis de nous familiariser avec certains principes de la robotique et d'y trouver une utilisation concrète. Nous avons été confronté a une technologie que nous ne connaissions pas et que nous avons du apprivoiser par l'essai et la logique.

Ce projet nous a permis de mettre à profit nos connaissances a la fois en informatique car c'est un aspect obligé lors de la programmation de NAO, et en robotique pour laquelle la logique de programmation diffère de ce que nous pouvons rencontrer lors de notre formation.

Malgré quelques contraintes (techniques et de réalisation), ce projet s'est avéré très intéressant et enrichissant. Le travail en groupe nous a permis de nous exercer sur le travail en équipe, ce qui est très important dans notre future vie professionnelle.

Rapport - Robot humanoïde NAO

Département Informatique
4e année
2011 - 2012

Projet d'Ingénierie du Logiciel

Résumé : Ce rapport couvre le déroulement du projet d'ingénierie logicielle à propos du robot NAO, robot mis à dispositions des écoles et entreprises par Aldebaran robotics, la société qui le fabrique. Le but de ce projet est dans un premier temps d'apprendre à utiliser le robot NAO. Une fois maîtrisé, ces connaissances devront être partagées par la rédaction d'un guide d'utilisation du robot, destiné à toute personne souhaitant se servir de NAO pour la première fois. Afin d'illustrer les possibilités du robot et concrétiser les connaissances acquises lors du projet, il est demandé de choisir une tache à faire réaliser à NAO et de programmer cette tâche. Les étapes de cette réalisation sont décrites dans ce rapport.

Mots clefs : NAO, Guide d'utilisation, Aldebaran, Projet Ingénierie Logicielle

Abstract: This report covers the progress of the software engineering project about the NAO robot, robot supplied for schools and businesses by Aldebaran Robotics, the company that manufactures it. The aim of this project is initially learning to use the NAO robot. Once mastered, this knowledge should be shared by writing a user guide, for anyone wishing to use NAO for the first time. To illustrate the possibilities of the robot and implement the knowledge gained in the project, we must choose a task to make NAO accomplish and code this task. The steps of this making are described in this report.

Keywords: NAO, User guide, Aldebaran, Software Engineering Project

Encadrant
Pierre GAUCHER
pierre.gaucher@univ-tours.fr

Université François-Rabelais, Tours

Étudiants
Joachim ALIBERT
joachim.alibert@etu.univ-tours.fr
Rémi CARUYER
remi.caruyer@etu.univ-tours.fr
Guillaume GEDEON
guillaume.gedeon@etu.univ-tours.fr
Chunhui LI
chunhui.li@etu.univ-tours.fr
Wei GONG
wei.gong@etu.univ-tours.fr
Benjamin PASQUET
benjamin.pasquet@etu.univ-tours.fr
Sébastien SCHAAL
sebastien.schaal@etu.univ-tours.fr
Cedric VERNOU
cedric.vernou@etu.univ-tours.fr

DI4 2011 - 2012