République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Abderrahmane Mira Bejaïa

Faculté de Technologie

Département de Génie Electrique

Mémoire de fin d'études

En vue de l'obtention du diplôme de Master en Automatique

Thème

Étude et développement d'applications pour le robot humanoïde NAO

Préparé par :

M^r^. MOUSSAOUI Messaoud

M^r^. MOUMENE Amazigh-Youcef

Encadré et dirigé par :

M^r^. MENDIL Boubekeur

Devant le jury composé de :

M^me^. Gagaoua Meriem

M^r^. Hadji Slimane

Soutenu le : Mardi 25/06/2019

Année Universitaire : 2018/2019

Remerciements

Nous adressons nos plus sincères remerciements à notre encadreur Mr B. MENDIL pour sa disponibilité, ses orientations et ses remarques, ses conseils, encouragements et toute la confiance qu'il a mise en nous.

Nos profonds remerciements aux membres du jury pour l'honneur qu'ils nous ont fait en acceptant d'évaluer notre travail.

Nos remerciements vont également à l'ensemble de l'effectif du département de Génie électrique et à l'ensemble de nos enseignants qui ont contribués à notre apprentissage.

Nous tenons à exprimer toutes nos reconnaissances à tous ceux qui ont contribué de prêt ou de loin à la réalisation de ce modeste travail.

Nous remercions finalement celui qui a été très cool et très sympathique avec nous « NAO ».

Dédicaces

Nous dédions ce modeste travail :

À nos très chers parents pour leurs efforts et sacrifices afin que nous puissions aujourd'hui réaliser notre rêve en travaillant sur ce projet remarquable et instructif.

À nos frères et sœurs, et à l'ensemble des membres de nos familles qui nous ont encourager pour fournir plus d'efforts et le meilleur de nous-même.

À nos ami(e)s et collègues automaticiens pour avoir passé de très bons moments lors des périodes de travails et pendant le cursus universitaire et à ceux qui n'ont pas eu la chance de travailler sur ce thème.

Tables des matières

Liste des figures

Liste des tableaux

Liste des abréviations

ddl : degrés de libertés.

B.D. : Boston Dynamics.

CPU : Central Processing Unit (Microprocesseur).

IEEE 802.11 : Institute of **E**lectrical and **E**lectronics **E**ngineers.

$\boldsymbol{mW/sr}$ **:** Milliwatts par Stéradian.

FSR : Force Sensitive Resistors.

LFsr : Left Fsr, **RFsr** : Right Fsr, **FL** : Front Left, **FR** : Front Right, **RL** : Rear Left, **RR** : Rear Right

MRE : Magnetic Rotatry Encoders (Codeur rotatif).

SOC : System On a Chip (Système embarqué sur une seule puce).

Fps : Frame per second (Image par seconde).

Int : Integer (nombre entier).

PC : Personal Computer.

IP : Internet Protocol.

V5 : Version 5.

SDK : Software Development Kit (Kit de développement logiciel).

IDLE : Python's Integrated Development and Learning Environment.

tts : Text to speech (texte à lire).

Introduction générale

Introduction générale

De nos jours, on entend presque quotidiennement que des scientifiques développent de nouvelles technologies ou de nouvelles structures qui ne nous laissent pas indifférent. Ces nouvelles techniques nous influencent en changeant notre vision et notre comportement vis-à-vis de notre environnement et de notre civilisation. Une tendance fascinante est le développement de robots humanoïdes qui seront, un jour, capables d'imiter l'attitude conviviale de l'être-humain.

L'objectif de notre travail étant de réaliser une étude et de développer des applications sur le robot humanoïde NAO. La première étape était d'étudier ce robot en faisant, en premier lieu, connaissances de ses ressources matérielles et logicielles. La deuxième concerne la mise en marche, la configuration et la préparation du robot pour l'implémentation des applications (installation des logiciels nécessaires, configuration du wifi, ajout de la deuxième langue). La dernière étape était l'étude et l'implémentation des applications.

Pour mieux présenter notre travail, on a organisé le manuscrit en quatre chapitres. Dans le premier, on a présenté des généralités sur la robotique (historique, évolution, différents types de robots). Dans le second chapitre, on s'est focalisé sur la description matérielle et la mise en marche de NAO. Par la suite, on a expliqué dans le troisième chapitre les fonctionnalités du logiciel Choregraphe permettant de commander le robot. Dans le dernier chapitre, on a exposé les applications qu'on a pu réaliser avec le logiciel Choregraphe et le langage de programmation Python.

Chapitre I : Généralités sur la robotique et les humanoïdes

I.1. Introduction

La robotique est un domaine technique qui regroupe plusieurs disciplines permettant de concevoir des machines automatiques qui fonctionnent en leurs fournissant une énergie de manière continue ou depuis des batteries rechargeables.

Étymologiquement parlant, le mot robot vient de la langue tchécoslovaque (anciennement) « robota » qui signifie le travail dur.

D'après le dictionnaire anglophone « Oxford English Dictionary », le terme robot est utilisé pour la première fois par Asimov en 1940 dans l'une de ses œuvres **[Web 1].**

Selon ATILF, la définition du robot est la suivante : « Appareil effectuant, grâce à un système de commande automatique à base de micro-processeur, une tâche précise pour laquelle il a été conçu dans le domaine industriel, scientifique, militaire ou domestique » **[Web 2].**

On peut comprendre donc que le champ d'application de cette technologie est important de nos jours. Cela dépend du type de robot, car ce dernier peut être conçu sous forme d'une machine, humanoïde ou sous forme d'un bot informatique (système d'exploitation, virus...).

I.2. Historique

Partant de l'idée de créer des mécanismes qui exécutent des tâches que l'être humain ne peut pas réaliser, les scientifiques ont pu franchir une nouvelle frontière dans le développement technologique.

Les premières inventions en robotique qui ont impressionné le monde entier sont apparues au début du 20 ème siècle, tels que la création du chien électrique en 1915 par « John Hammond » et « Benjamin Miessner ».

Quarante ans plus tard, les inventeurs ont proposé de nouveaux concepts dont leur particularité était de reproduire certains réflexes d'animaux, qui sont, les tortues cybernétiques de « William Grey Walter » (1950), le renard électronique de « Albert Ducrocq » (1953) ou l'homéostat de « W. Ross Ashby » (1952).

La robotisation de l'industrie a commencé dans les années 1960 et elle est devenue rapidement indispensable dans ce secteur pour des raisons de compétitivité et de la production en masse.

À la fin du 20ème siècle, la robotique de transport de personnes et de marchandises portuaire font leur apparition dans certains coins du monde.

Les robots domestiques destinés au grand public, quant à eux, font leur apparition plus tard, au début du 21ème, on cite à titre d'exemple les aspirateurs et les tondeuses autonomes.

Toujours au début du 21ème mais au niveau militaire cette fois-ci, se développent les tourelles automatiques sur les navires de guerre et les drones **[1].**

« Hier fiction, maintenant réalité, le robot humanoïde, machine intelligente créée sur le modèle humain, concrétise des rêves ancestraux, avec pour ambition que la copie puisse égaler ou dépasser son modèle et créateur. » **[2].**

De nos jours, les humanoïdes sont en voie d'intégration dans la société civile, pour leurs permettre d'apprendre et d'acquérir de nouvelles capacités.

I.3. Types de robots

I.3.1. Robots industriels

Les robots industriels sont les premiers à avoir été produits en grand nombre afin d'optimiser la production industrielle. Ils se trouvent plus particulièrement sur les chaînes de montage et le plus souvent dans l'industrie automobile. Dans ce secteur, on peut trouver des robots de soudures, de nettoyage, d'emballage ou de surveillance.

Figure I.1. Soudure effectuée par un robot.

Figure I.2. Robot nettoyeur.

I.3.2. Robots médicaux

Un robot médical est avant tout un système de chirurgie intégré à un ordinateur (computer-integrated-surgery). En d'autres termes, le robot lui-même n'est qu'un élément d'un système plus vaste conçu pour aider le chirurgien à effectuer une intervention chirurgicale pouvant inclure la planification et l'enregistrement dans les plans préopératoires et à utiliser de façon combinée une assistance robotique et d'outils à commande manuelle pour la réalisation du plan ainsi que la vérification et le suivi postopératoire **[3].**

Il peut certes réaliser quelques gestes de façon automatique, mais il est le plus souvent soit télé-manipulé, via une interface appropriée (généralement un bras maître), soit manipulé par

une action directe du chirurgien sur l'instrument porté par le robot. Le choix du mode de commande des robots (automatique, télé-manipulé ou comanipulé) dépend du geste chirurgical à accomplir et, donc, de considérations d'ergonomie de l'interface homme-machine. Les robots télé-manipulés sont les plus nombreux, car c'est le mode de commande du robot *Da Vinci*, le plus répandu aujourd'hui **[Web 3].**

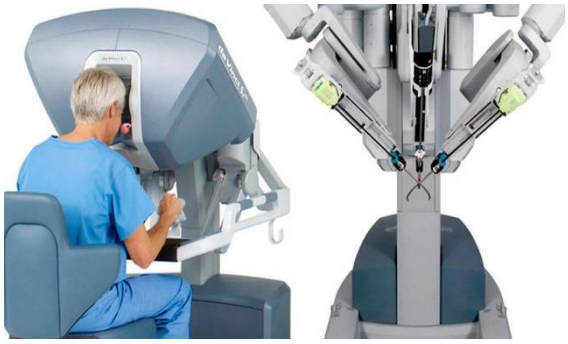


Figure I.3. Robot chirurgical Da Vinci.

I.3.3. Robots domestiques

Il s'agit généralement de robots destinés à usage particulier. On peut citer l'exemple d'une tendeuse à gazon automatique qui coupe le gazon de façon autonome sur le périmètre souhaité en évitant de se heurter aux obstacles.

Figure I.4. Tondeuse automatique.

Pour le divertissement et l'accompagnement, Sony a conçu un robot chien nommé Aïbo qui est très populaire et capable d'apprendre. La version la plus récente est équipée d'une technologie de pointe et possède des fonctionnalités très avancées telles que : des articulations qui lui permettent de faire bouger ses pattes, sa tête, ses oreilles et sa queue. Il est également équipé de deux caméras, d'un haut-parleur, de microphones, d'accéléromètre, d'un gyromètre et de capteurs tactiles.

Figure I.5. Robot Aïbo

Quant à Boston Dynamics (B.D.), elle a fabriqué récemment « SpotMini » qui sera commercialisé fin 2019 d'après le constructeur. SpotMini est un petit robot à quatre pattes qui convient parfaitement au bureau ou à la maison. Il pèse 25 kg (30 kg en incluant le bras). Il est entièrement électrique et la batterie peut tenir environ 90 minutes avec une charge, en fonction de ses activités **[Web 4].**

Figure I.6. SpotMini qui utilise un bras pour prendre objet sur la table.

I.3.4. Robots militaires

Ils sont principalement utilisés pour la surveillance dans les airs comme dans la mer. A titre d'exemple, les avions sans pilotes qui surveillent, reconnaissent ou identifient même des cibles ennemies **[4].**

Figure I.7. Robots utilisés dans le secteur militaire.

I.3.5. Robots explorateurs

Les robots explorateurs remplacent l'être-humain dans des environnements hostiles. L'exploration de l'un des réacteurs nucléaires de Fukushima a été faite par un robot quadrupède de TOSHIBA. L'exploration de l'espace s'effectue de nos jours le plus souvent par des robots (étude de planètes et d'astres).

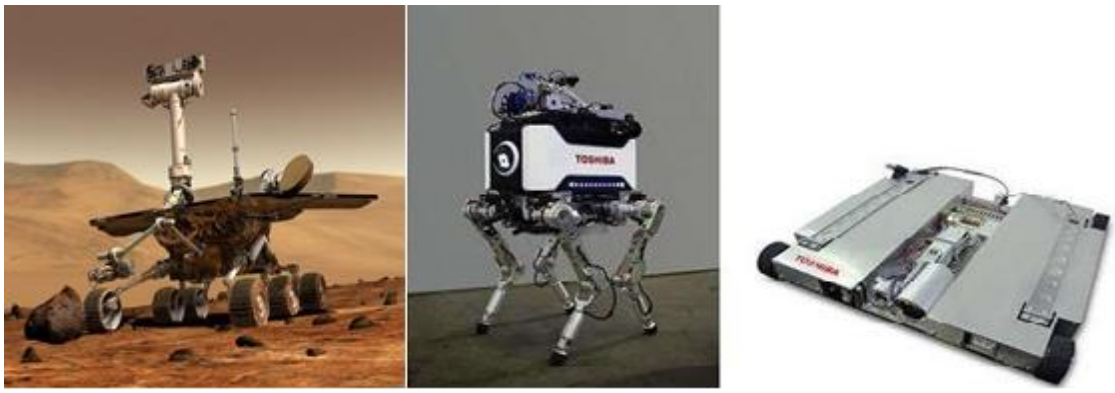


Figure I.8. Illustration de robots explorateurs Spirit (à gauche) et TOSHIBA (à droite).

I.3.6. Robots anthropomorphiques

Ces robots ont l'apparence humaine, ils sont dotés généralement de la bipédie (deux pieds), de deux bras et d'une tête. Ils sont capables de copier quelques gestes que seul l'humain pouvait faire jusqu'à aujourd'hui. La plus récente version de ce type de robot peut courir à 6 Km/h et peut aussi remplir la tâche de réceptionniste ou de guide d'information. C'est une catégorie de robots qui devraient être capable de faire le rôle d'hôtesse d'accueil, de venir en aide aux

personnes handicapées, âgées ou malades, aider des astronautes dans des tâches complexes et peut-être un jour, ils pourront intégrer et interagir avec les sociétés civiles **[5].**

Malgré le fait que les humanoïdes soient capables d'imiter relativement l'être-humain, ils ne sont pas assez développés pour communiquer de façon conviviale avec nous pour l'instant, puisque presque la moitié de la communication n'est ni verbal, ni vocal **[6].**

Figure I.9. Le robot collaboratif Swayer conçu par Rethink Robotics.

Dans ce qui suit, on va citer quelques humanoïdes les plus connus de nos jours.

I.3.6.1 Le robot Atlas

Conçu par Boston Dynamics, une société américaine fondé en 1992, c'est l'un des robots les plus célèbres de notre époque puisqu'il est performant, agile et capable de courir et de réaliser des sauts.

Atlas est le dernier-né d'une gamme de robots humanoïdes avancés que B.D. développe. Le système de commande d'Atlas coordonne les mouvements des bras, du torse et des jambes pour permettre une manipulation mobile du corps entier, élargissant considérablement sa portée et son espace de travail. La capacité d'Atlas à maintenir son équilibre tout en effectuant des tâches lui permet de travailler dans un volume important tout en occupant un faible encombrement.

Le matériel Atlas est fabriqué avec l'imprimante 3D pour gagner du poids et de l'espace. Il en résulte un robot compact remarquable avec un rapport force / poids élevé et un espace de travail extrêmement grand. La vision stéréo, la détection de distance et d'autres capteurs permettent à Atlas de manipuler des objets dans son environnement et de voyager sur des terrains accidentés. Atlas garde son équilibre lorsqu'il est bousculé ou poussé et peut se relever s'il tombe par terre **[Web 5].**

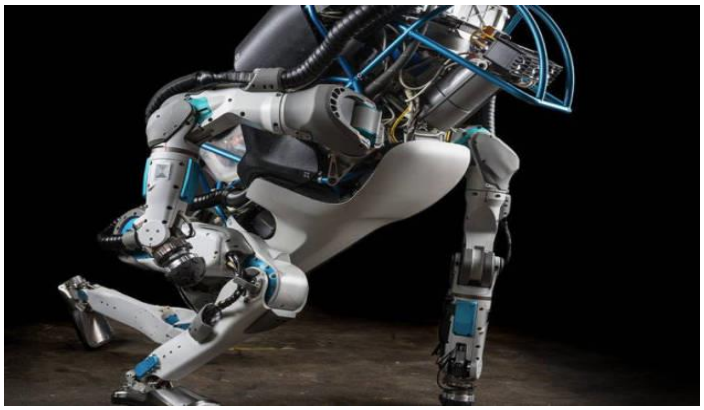


Figure I.10. Robot Atlas.

I.3.6.2 Le robot ASIMO

ASIMO est un robot japonais (plusieurs versions de prototypes) produit par la société Honda. Il n'est pas commercialisé pour l'instant mais il a été développé pour la recherche. La dernière version possède 57 degrés de libertés (ddl), il peut ouvrir des bouteilles, parler quelques langues, chanter, sauter sur un seul pied tout en gardant son équilibre. En 2008, il a joué le rôle de chef d'orchestre dans le Detroit Symphony Orchestra **[7][Web 6].**

Figure I.11. Tâches que ASIMO peut réaliser.

I.3.6.3 Le robot Kengoro

Kengoro est un robot créé par une équipe de chercheur de l'université de Tokyo (successeur de Kenshiro). Il est doté de 116 actionneurs musculaires et de 174 ddl (incluant le visage et les mains) lui attribuant une grande précision de mouvement. L'une des caractéristiques qu'il le distingue des autres humanoïdes, est le fait qu'il possède des bras flexibles et solides, par conséquent, il est capable d'effectuer des pompes comme un être humain et transpirer grâce un système de refroidissement sophistiqué à base d'eau **[8][Web 7].**

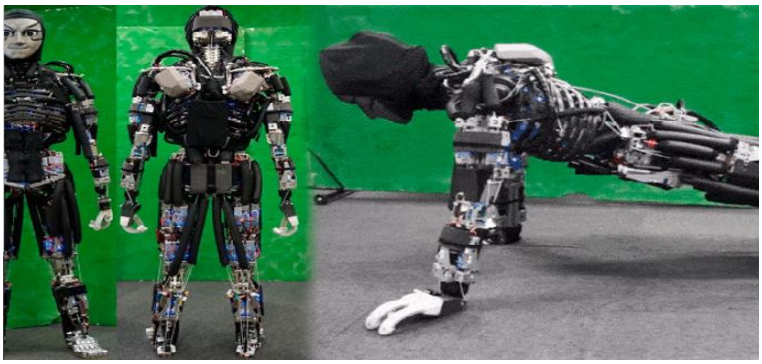


Figure I.12. Illustration de Kengoro en application.

I.3.6.4 Le robot Pepper

Faisant partie du même groupe que B.D., SoftBank Robotics (Aldebaran anciennement) a su s'imposer sur le marché international en concevant des humanoïdes destinés au public qui sont : NAO, Romeo et Pepper. Ils peuvent être utilisé pour l'éducation, la recherche et le divertissement (usage personnel).

Figure I.13. La famille des humanoïdes de la firme Softbank Robotics

Lancé récemment, Pepper coûte environ 20,000 euros, c'est le robot vedette du moment. Ayant une autonomie de batterie de 12 heures, qui est dû notamment à son système de motricité, étant aussi équipé de trois roues omnidirectionnelles, il a la faculté de parler facilement avec des personnes, de les orienter et de leurs donner des informations. Pour l'instant il est destiné pour un secteur professionnel **[9] [Web 8].**

Figure I.14. Illustration du robot Pepper.

I.4. Conclusion

Dans ce premier chapitre du mémoire, on a abordé, de façon générale, la discipline de la robotique et son champ d'application. On a décrit les différents types de robots existants de notre ère. Puis, on a cité quelques humanoïdes les plus fascinants tout en indiquant leurs facultés.

Vu que notre objectif concerne l'étude et le développement d'applications pour le robot humanoïde NAO, il est impératif de décrire son aspect technique et de présenter ses constituants à travers le second chapitre.

Chapitre II : Présentation du robot humanoïde NAO

II.1. Introduction

Dans ce chapitre, nous allons décrire le robot humanoïde ***NAO V5***, ses caractéristiques techniques et mécaniques, ses différentes parties et composants qui le constituent, en ajoutant à cela, leurs principes de fonctionnement et quelques illustrations pour mieux comprendre justement l'utilité de ses constituants et de tout le système de NAO et d'assurer sa mise en marche de manière correcte.

II.2. Historique du concepteur de NAO

Aldebaran Robotics est une entreprise française qui a été créée en 2005 ayant pour objectif la production de robots humanoïdes destinés à évoluer dans la société humaine. En 2008, le robot NAO a vu le jour et plusieurs versions lui ont succédé **[10].**

En 2014 cette entreprise a changé de nom en « Softbank Robotics ». Car, elle a été rachetée par l'entreprise japonaise Softbank. Aujourd'hui, on en est à la $6ème$ version qui fut lancée en 2018. C'est la plus évoluée depuis le début de son aventure, puisque NAO intègre un tout nouveau CPU qui accroît ses performances. Il est développé sur une plateforme Open Source **[Web 9].**

II.3. Description générale de NAO [11]

NAO est un robot humanoïde qui a été conçu dans le but d'interagir avec l'être-humain et l'environnement en parlant plusieurs langues. Pour ce faire, les concepteurs lui ont intégrés différents types de technologies : CPU, capteurs, actionneurs, caméra et LEDs.

Figure II.1. Le robot humanoïde NAO et le logo d'ALDEBARAN (anciennement) [11].

II.4. Caractéristiques techniques [11]

On va regrouper les caractéristiques techniques de NAO dans le Tableau II.1. Puis, on va décrire chaque composant accompagné d'illustrations. La Figure II.2 illustre bien l'architecture de NAO.

Tableau II.1. Composants électronique de NAO.

Éléments	**Types**	**Spécifications**
Carte mère	**ATOM Z530**	CPU 1.6 GHz
	RAM	1 Go
	Mémoire flash	2 Go
	Mémoire SDHC	8 Go
Capteurs	Ultrason	2 Emetteurs /2 Récepteurs
	Infrarouge	2 (Yeux)
	Capteurs Tactiles	Emplacement : Tête, mains et pieds
	Résistance à capteur de force (FSR)	Emplacement : Pieds
	Capteurs de position d'articulation	Capteur à effet de Hall 36 à 2 MRE
	Caméra (capteur d'image SOC)	HD, $1280\text{x}960, 30fps$
LEDs	/	Disposition : Tête, yeux oreilles, mains et pieds
Haut-Parleurs	/	Nombre : 2 Stéréo
Microphones	/	Nombre : 4 sur la Tête Sensibilité : 20mV/Pa +/-3dB à 1 KHz Fréquence : $150Hz$ à $12kHz$
Batterie	Lithium ion	Durée : 60 à $90mn$ Temps de recharge : 3 heures
Connectivité	Wifi	IEEE 802.11 a/b/g/n
	Ethernet	1×RJ45 - 10/100/1000 base T
	USB	2.0

Pour bien illustrer l'architecture de NAO, nous avons choisi la figure en dessous.

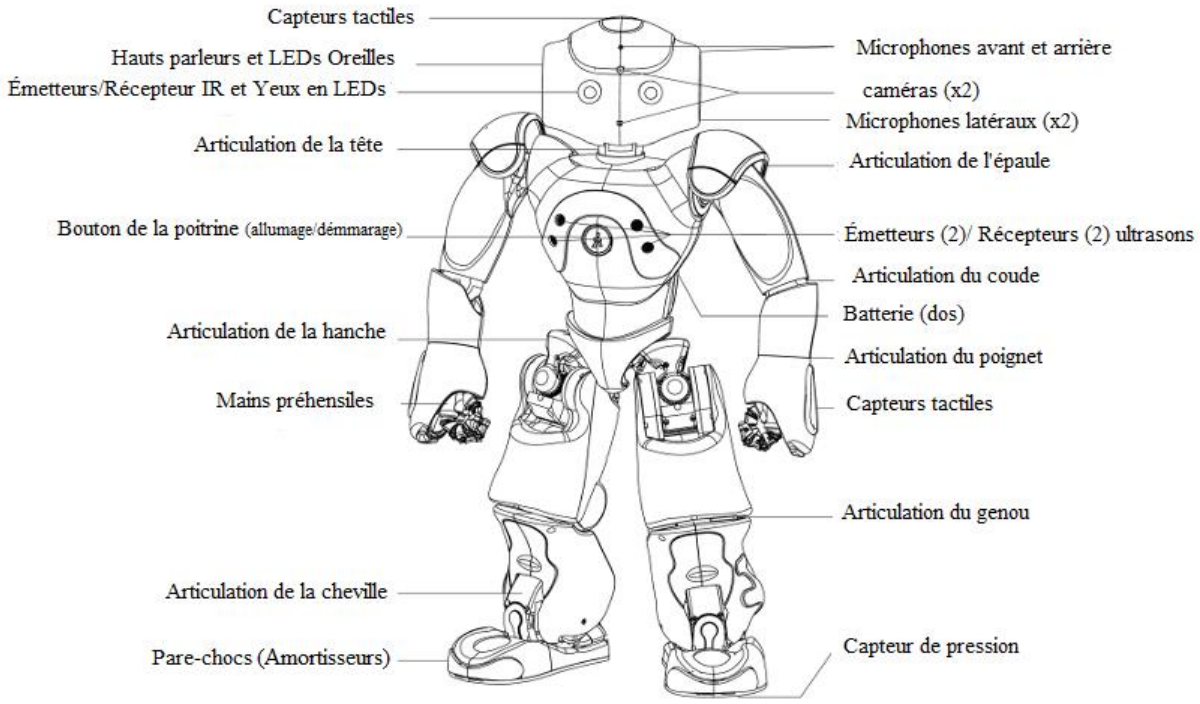


Figure II.2. Illustration qui détaille l'ensemble des éléments qui sont inclus dans NAO [11].

II.5. Description mécanique [11]

Il mesure 58 cm environ et pèse 5.4 Kg. Son centre d'inertie se situe dans son torse possédant son propre processeur, un gyromètre dont sa vitesse angulaire qui $\approx 500°/s$ et un accéléromètre ($2g$), ayant tout les deux 3 axes chacun.

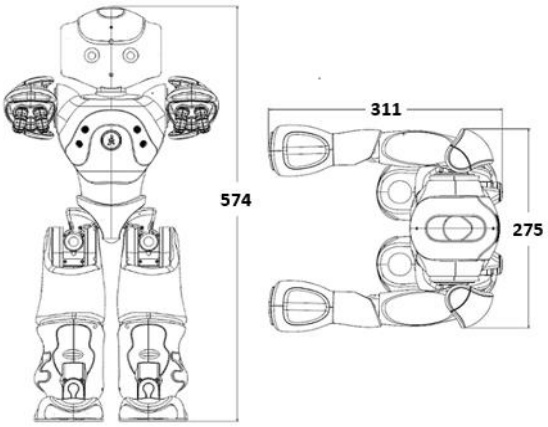


Figure II.3. Vue en perspective des dimensions de NAO (unité en mm) [11].

II.6. Description des capteurs [11]

La cinquième version de NAO est constituée de différents capteurs ayants comme fonction de lui donner une perception précise et optimale dans l'espace où il évolue.

II.6.1 Disposition des capteurs ultrasons

Comme il est indiqué dans le premier tableau, NAO possède quatre capteurs ultrasons, dont deux sont des émetteurs et les deux autres sont des récepteurs.

➢ Spécifications :

- **Fréquence :** 40 kHz.
- **Résolution :** 1 cm à 4 cm.
- **Portée de détection :** 0.20 m à 0.80 m. Au-dessous de 20 cm, il n'y a pas d'information sur distance, le robot sait seulement qu'un objet est présent. Au-dessus de 80 cm, la valeur renvoyée est une estimation.
- **Cône d'environ :** 60°.

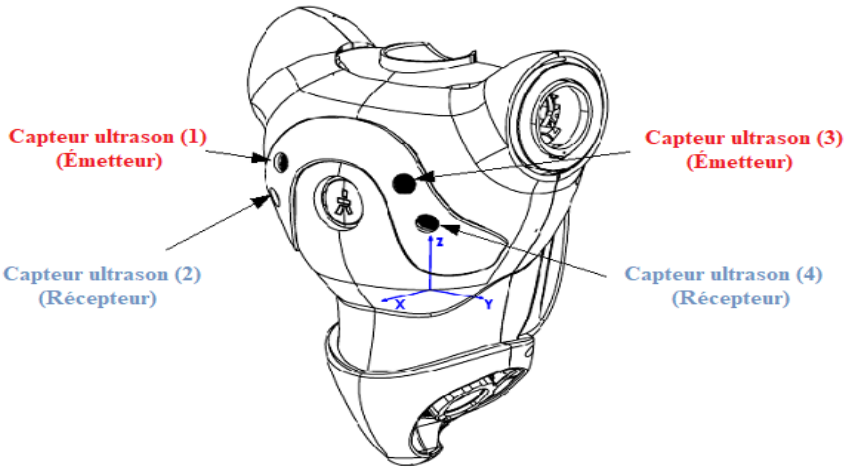


Figure II.4. Emplacement des capteurs ultrasons ainsi que leur ordre de numérotation [11].

II.6.2 Disposition des capteurs infrarouges

Les deux capteurs infrarouges (IR) de NAO se situent au niveau de ses yeux.

➢ Spécifications :

- **Longueur d'onde :** 940 nm.
- **Angle d'émission :** $+/-$ 60°.
- **Puissance :** 8 mW/sr.

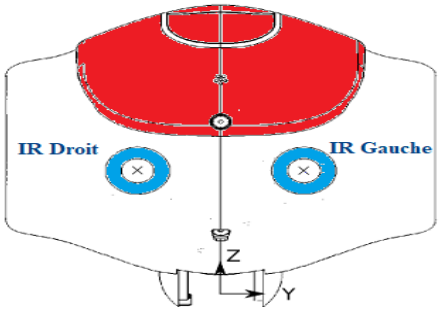


Figure II.5. Les capteurs sont indiqués par les deux croix [11].

II.6.3. Capteurs tactiles

NAO est constitué de capteurs tactiles dont l'utilité est d'offrir une autre possibilité de communication avec l'utilisateur. Ces capteurs sont mis dans différentes zones du robot.

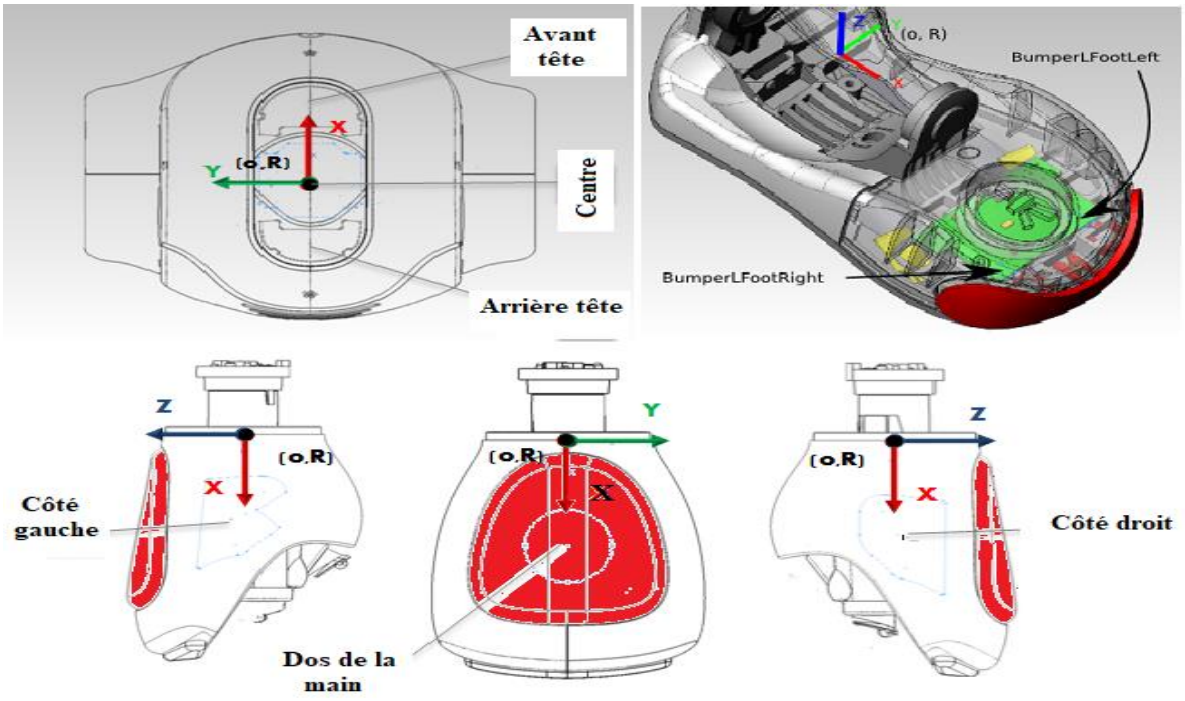


Figure II.6. Cette illustration indique la disposition des capteurs tactiles de NAO [11].

II.6.4. Capteur de force résistif (FSR)

Les capteurs FSR (Force Sensitive Resistors) sont placés en dessous des pieds de l'humanoïde pour assurer son équilibre. Ce type de capteurs mesurent la variation de la résistance selon la pression appliquée. L'intervalle de fonctionnement est de [0 à 25 N].

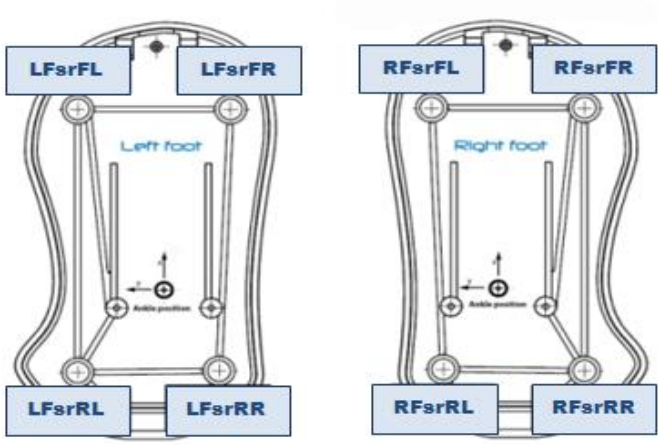


Figure II.7. Vue d'en haut de la position des capteurs Fsr (**LFsr** : capteur pied gauche, **RFsr** : capteur pied droit, **FL** : Front Left, **FR** : Front Right, **RL** : Rear Left, **RR** : Rear Right) [11].

II.6.5. Capteurs de position d'articulation

Ces capteurs fonctionnent à base d'encodeurs rotatifs utilisant la technologie d'effet de Hall qui déterminent la position angulaire. Leur précision est de 12 bits, c'est-à-dire 4096 valeurs par rotation, correspondant à 0.1° de précision **[11].**

II.6.6. Caméra

NAO possède deux caméra HD sur la face (au niveau du front et de la bouche) ayant une résolution de 1280x960 à 30fps qui sont utilisées pour l'identification d'objets présents dans son champ de vision tel qu'un ballon ou une carte de jeu.

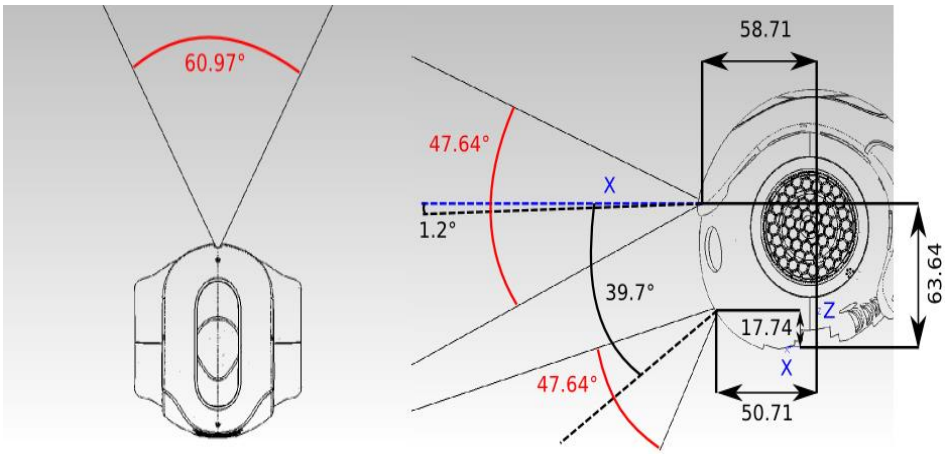


Figure II.8. Champ de vision de NAO sur l'horizontale et la verticale (unité distance : mm, angle) [11].

II.7. Les moteurs (Actionneurs) [11]

Comme son nom l'indique, NAO est un humanoïde qui a la faculté de reproduire certains mouvements de l'être-humain. C'est pour cela que les concepteurs l'ont équipé de moteurs faisant le rôle d'articulations pour lui attribuer 25 degrés de libertés **[12].**

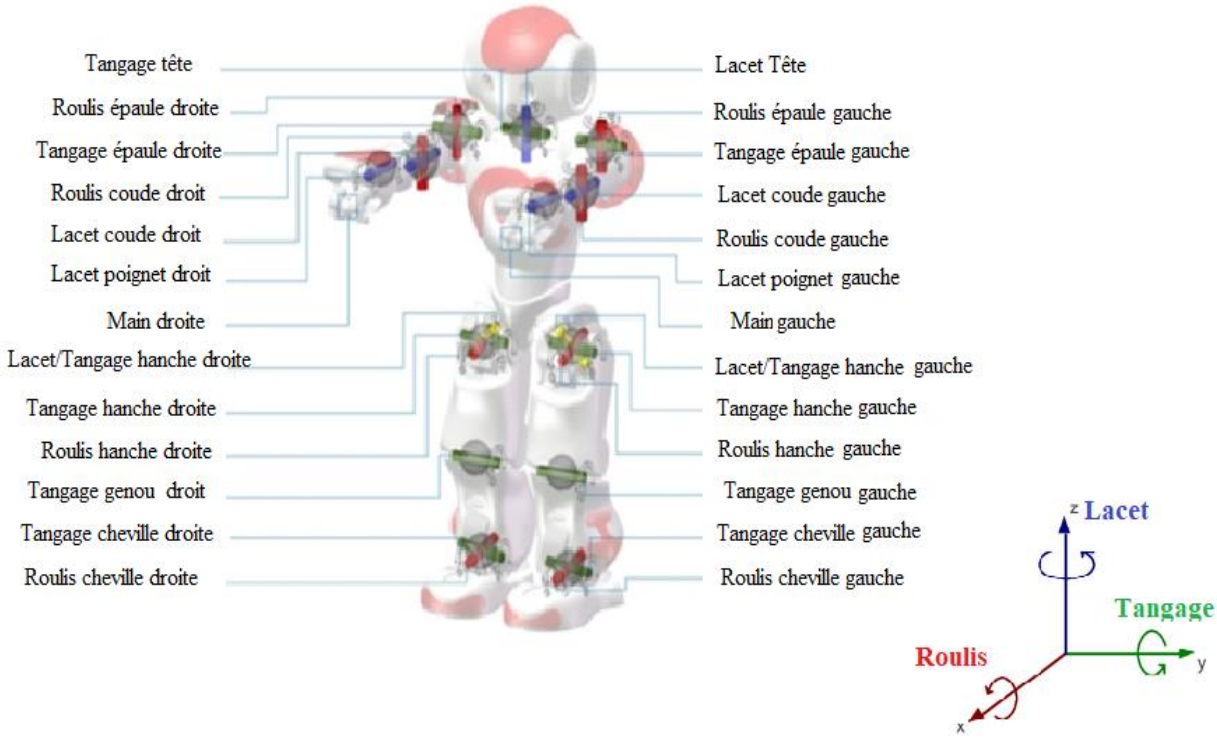


Figure II.9. Ensemble des articulations incluses dans NAO qui fonctionnent selon le repère indiqué dans la figure.

II.8. Autres composants [11]

II.8.1. LEDs

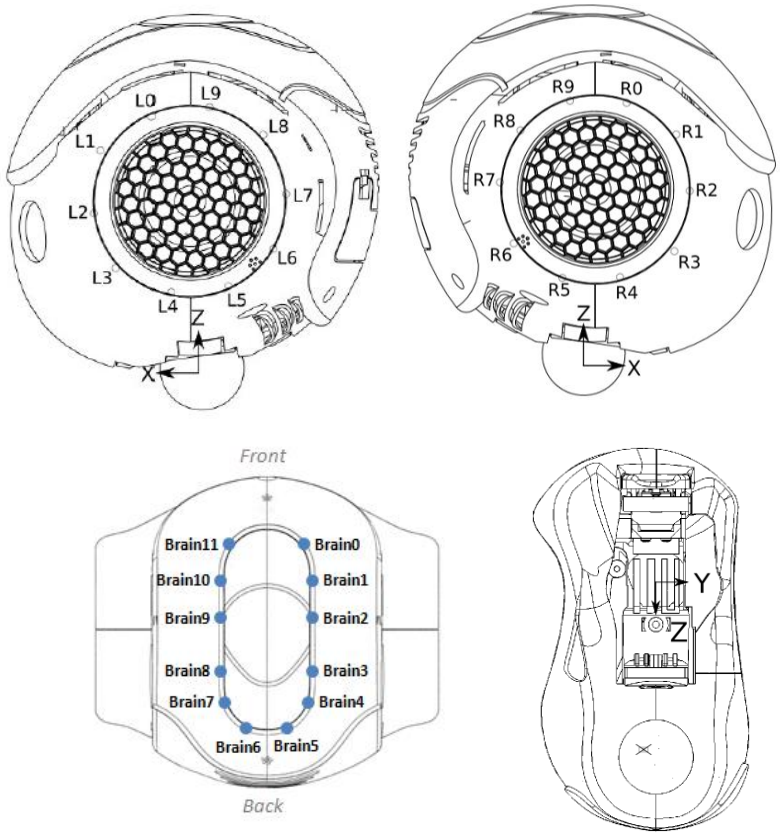


Figure II.10. Répartition des LEDs selon leurs numéros (Tête, oreilles, pied droit et gauche).

Des LEDs sont placées un peu partout dans NAO pour lui donner, non seulement un aspect esthétique, mais également, selon la couleur d'allumage de ces LEDs, des informations sur l'état de NAO que nous allons traiter dans la partie **« Mise en marche »**.

II.8.2. Haut-parleurs

Le haut-parleur stéréo est un transducteur électroacoustique qui produit des sons à partir d'un signal électrique. Situés au niveau de ses oreilles, ils permettent à NAO de communiquer un message ou de jouer de la musique.

II.8.3. Microphones

Un microphone est un autre transducteur électroacoustique, ce qui signifie qui peut convertir un signal acoustique en signal électrique (l'inverse du haut-parleur). NAO possède quatre microphones situés dans sa tête.

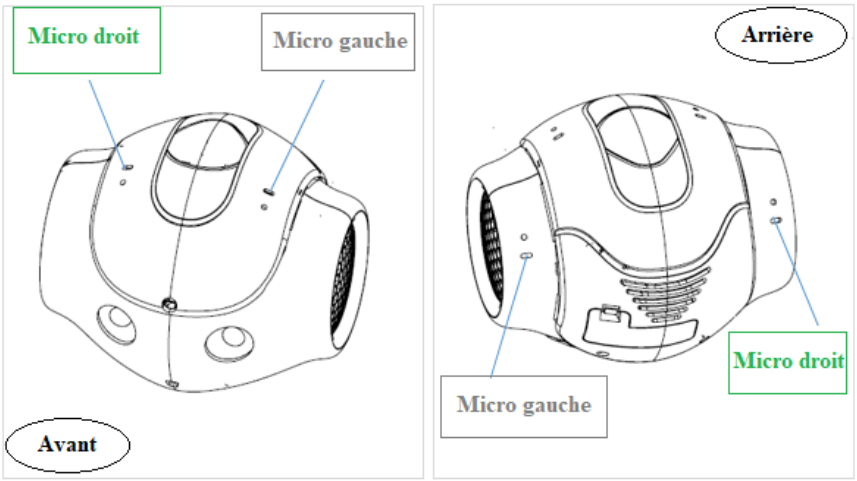


Figure II.11. Microphones intégrés dans NAO.

II.8.4. Batterie

NAO est équipé d'une batterie Li-ion, ayant une autonomie de 60 à 90 mn et un temps de recharge d'environ 3 heures.

II.9. Mise en marche (connectivité) [11] [13]

Ce qu'il faut bien retenir avant d'utiliser le robot NAO, c'est qu'il doit être manipulé avec précautions. Donc pour bien démarrer, il est impératif de suivre quelques consignes fournies par le constructeur. Voici les étapes à suivre.

II.9.1. Connexion et initialisation de NAO

Pour une toute première utilisation, il faut d'abord mettre le robot en position de sécurité et ne pas mettre les mains sur ses articulations. Puis, brancher l'alimentation avec le chargeur distribué. Ensuite, connecter NAO au PC via un câble Ethernet (ou bien vers un routeur auquel est connecté le PC). Cette étape permet de créer et de configurer un réseau Wifi entre le robot et le PC et offrir une meilleure expérience d'utilisation, en se débarrassant des câbles encombrants et dangereux durant les déplacements du robot. D'après les tests effectués, il n'est pas exigé que le routeur (Modem) soit connecté à Internet afin d'établir une connexion. Par contre, si on veut télécharger une application ou ajouter une nouvelle langue, la connexion Internet est obligatoire.

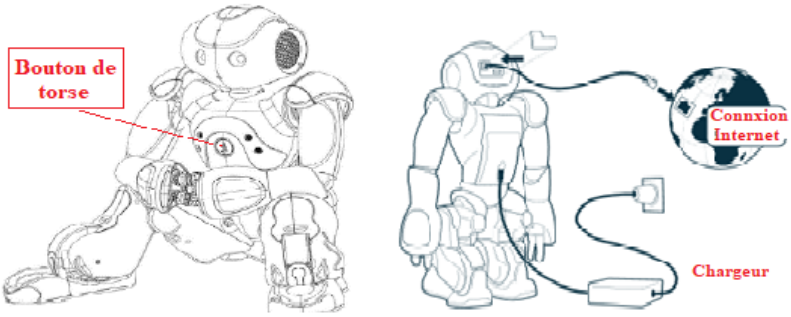


Figure II.12. Position de sécurité et système de connexion de NAO.

Une fois la liaison établie, on peut procéder à la mise en marche de l'humanoïde et cela en appuyant sur le bouton du torse (indiqué par la figure au-dessus). Le temps de démarrage peut prendre de 3 à 5 mn. On peut suivre la progression de démarrage à travers les LEDs qui entourent les haut-parleurs. Après un moment, le robot prononcera l'expression « **OGNAK GNOUK** », puis, il va commencer à regarder autour de lui et reconnaitre des visages, c'est un signe que NAO est prêt à l'utilisation.

À la fin de l'utilisation, la procédure de l'extinction se fait ainsi : il faut appuyer sur le même bouton de démarrage pendant cinq secondes. Les lumières des yeux, du torse et de la tête seront toutes éteintes quand le processus d'extinction sera terminé.

Lorsqu'on communique avec NAO, ses yeux s'illuminent avec différentes couleurs :

- Jaune, signifie qu'il vous écoute et traite les sons.
- Bleu, en attente de réception de messages.
- Vert, signifie que le message est reconnu.
- Il alterne entre le blanc et le bleu lorsqu'il parle **[13].**

II.9.2. Configuration Wifi

D'abord il est nécessaire de créer un compte sur Aldebaran Community en raison de la synchronisation. Pour configurer le Wifi pour la première fois, il faut garder le câble Ethernet branché. Par la suite, on appuie sur le bouton de mise en marche après que NAO soit démarré afin d'obtenir son adresse IP (Internet Protocol). Si on n'a pas bien retenu les chiffres, on appuie une nouvelle fois pour qu'il répète le message moins vite. A présent qu'on a récupéré l'adresse, on l'insère dans la barre d'adresse d'un navigateur web (Firefox ou Chrome) et on valide la recherche. Une boite de dialogue d'authentification va apparaître qui demande de saisir le nom d'utilisateur et son mot de passe qui sont par défaut « nao, nao » respectivement **[11][13].**

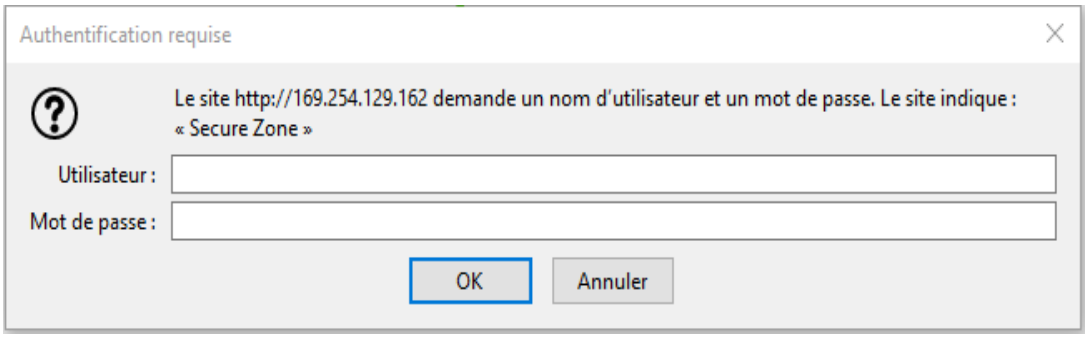


Figure II.13. Boite de dialogue d'authentification.

Après cela, une fenêtre d'assistance « Getting Started » va apparaître pour orienter l'utilisateur.

Figure II.14. Première étape de configuration.

Dans l'étape qui suit, c'est la fenêtre de configuration Wi-Fi qui apparaitra, on peut alors en sélectionner un puis en saisissant son code d'accès, ou bien cliquer sur "Skip" ou "Ignore" pour laisser NAO sans Wi-Fi pour l'instant **[10].**

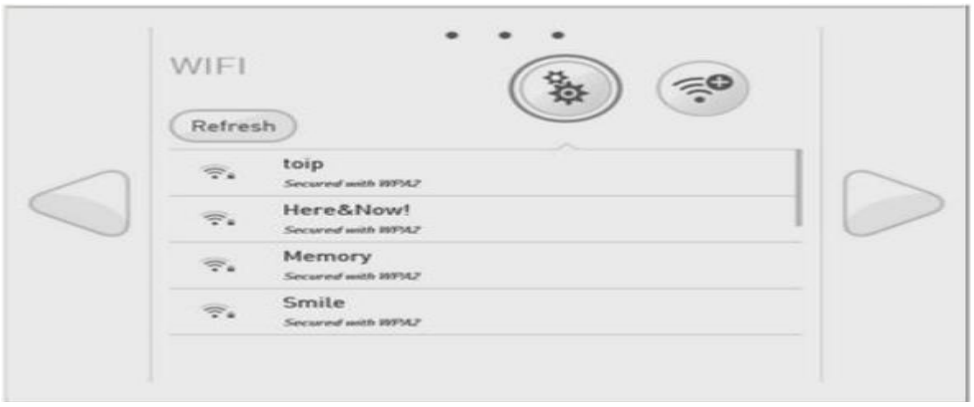


Figure II.15. Réseaux Wifi détectés à proximité.

On a la possibilité de changer le nom du robot ainsi que le mot de passe. (Par-contre, il faut noter précisément les changements, ils seront demandés à chaque authentification).

Les fenêtres vont s'afficher dans l'ordre indiqué par l'illustration en dessous. Pour passer à la prochaine étape, on clique sur suivant.

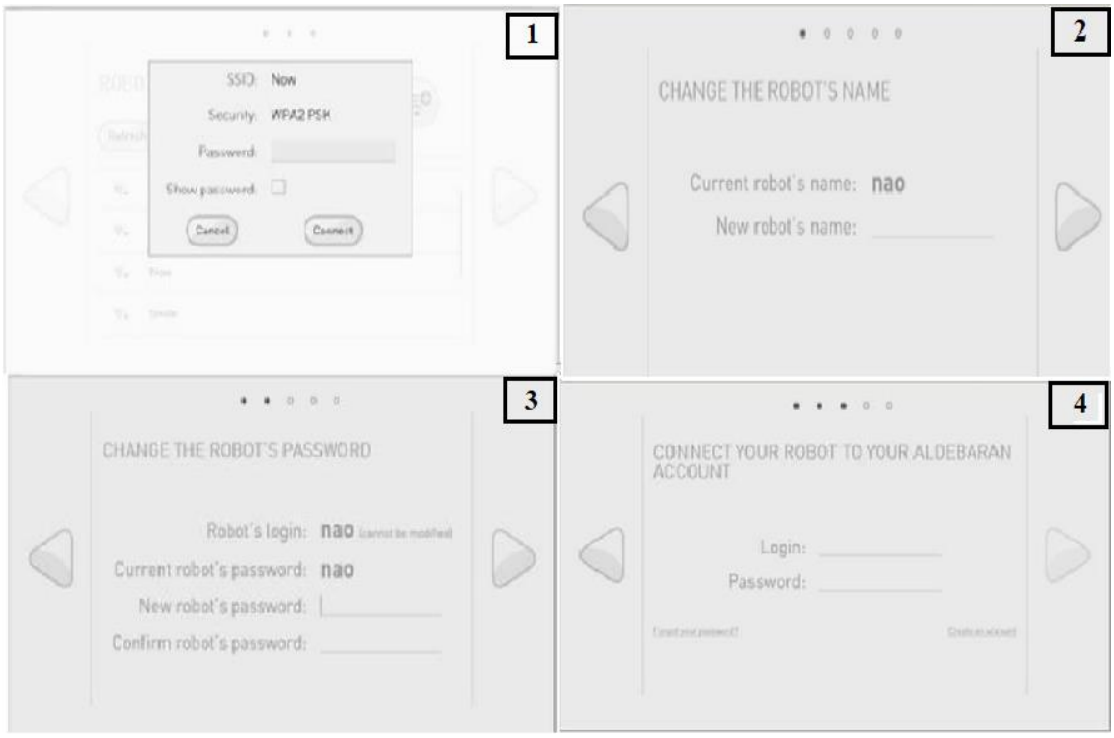


Figure II.16. Fenêtres de configuration de nom, de mot de passe et de synchronisation du compte.

On a besoin de redémarrer NAO si on modifie son nom. On vérifie que la batterie soit pleine ou encore mieux, que le chargeur soit branché, et ce n'est qu'à ce moment-là que l'on pourrait cliquer sur le bouton "Reboot now" pour redémarrer NAO.

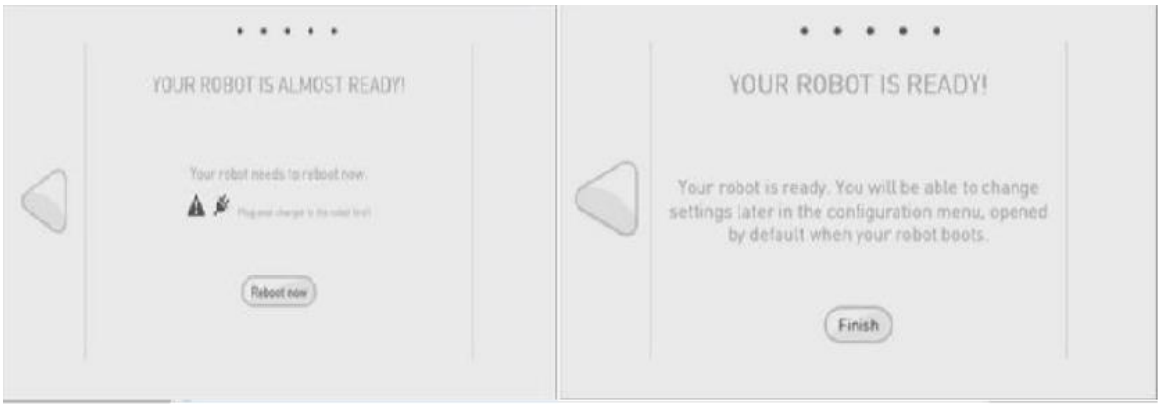


Figure II.17. NAO est prêt pour le redémarrage.

En cliquant sur « Finish », on obtient l'interface suivante :

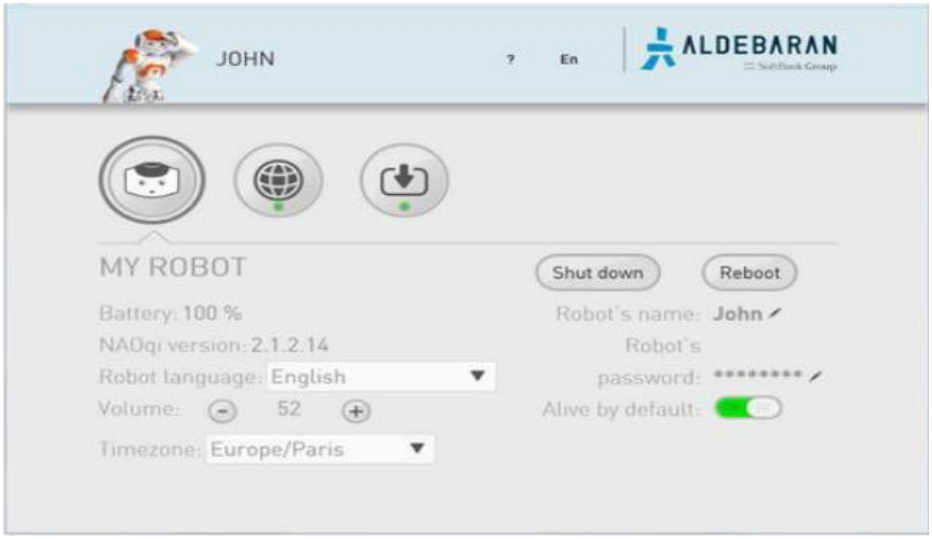


Figure II.18. Interface Web de NAO.

Une fois qu'on a terminé la configuration, on lance l'application Choregraphe et on introduit l'adresse IP de NAO. Car, cette fois-ci, on peut débrancher le câble Ethernet et le chargeur de batterie (après la charge) et donner plus de liberté de mouvements à NAO et une certaine autonomie.

II.10 Procédure d'installation d'une nouvelle langue [13]

La robot NAO est fourni avec une seule langue (Anglais). Mais, il nous donne le choix d'installer une deuxième langue. Celle-ci peut servir pour la communication et la configuration

ultérieure du robot ou comme une langue d'échange et de dialogue dans l'implémentation des applications concernant l'analyse et la synthèse vocale.

La première étape consiste à créer un compte dans le site Aldebaran (https://community.aldebaran.com/en/content/your-nao-his-way). Dans ce lien, on vous demandera des informations que vous devez remplir (nom, prénom, mot de passe, e-mail…). D'après notre expérience, aucun e-mail de confirmation de création de la boite nous a été envoyé, mais notre compte a été validé.

L'étape suivante consiste à synchroniser le compte créé et le robot. Pour ce faire, il faut se connecter à l'adresse IP du robot via un navigateur web (il vaut mieux connecter le robot au PC avec le câble Ethernet). Puis, il faut s'orienter vers le téléchargement de nouveaux modules (ou de mise à jour). Après cela, il faut cliquer sur le bouton « **Modifier le compte** ». Une boite de dialogue va apparaitre et vous demande d'introduire l'e-mail et le mot de passe du compte Aldebaran. Ce n'est qu'après avoir synchronisé le compte que l'on pourra accéder au « Store » du robot NAO, car depuis ce site vous pouvez télécharger des langues ou des applications : https://cloud.aldebaran-robotics.com/languages/

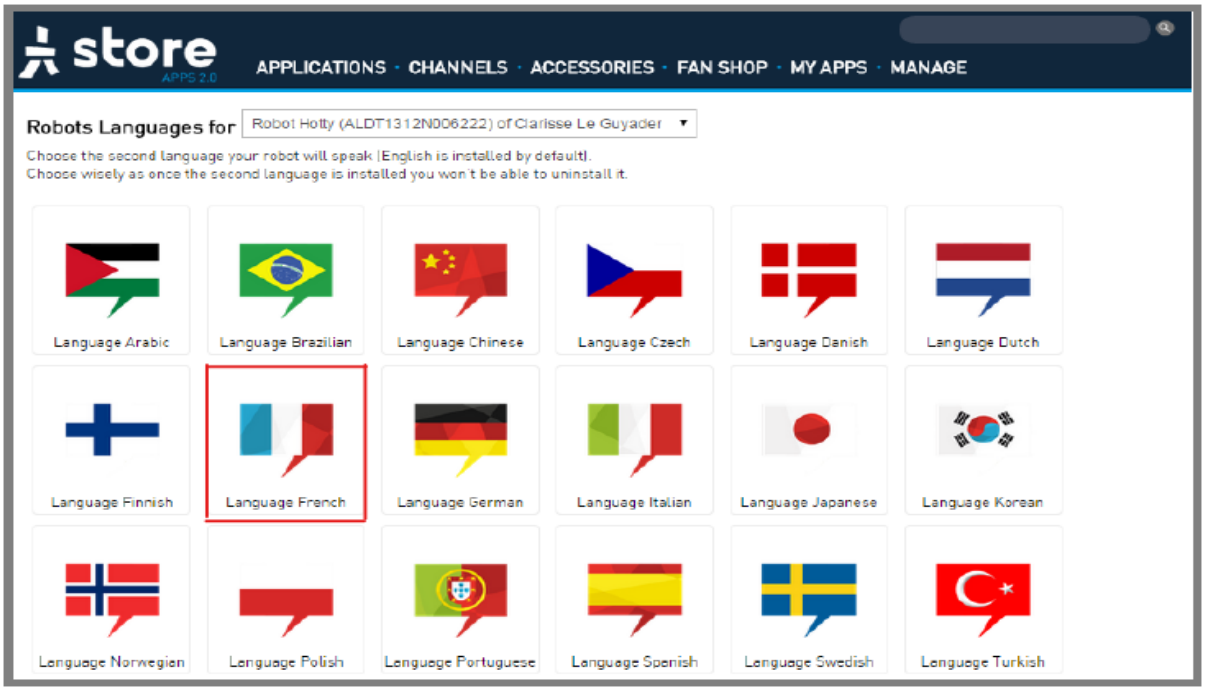


Figure II.19. Store officiel pour ajouter une langue.

On choisit la langue à installer en cliquant dessus (langue française dans notre cas). Une boite de dialogue vous demandera de valider votre choix (1). Si une langue est installée (2), on ne pourra plus la supprimer.

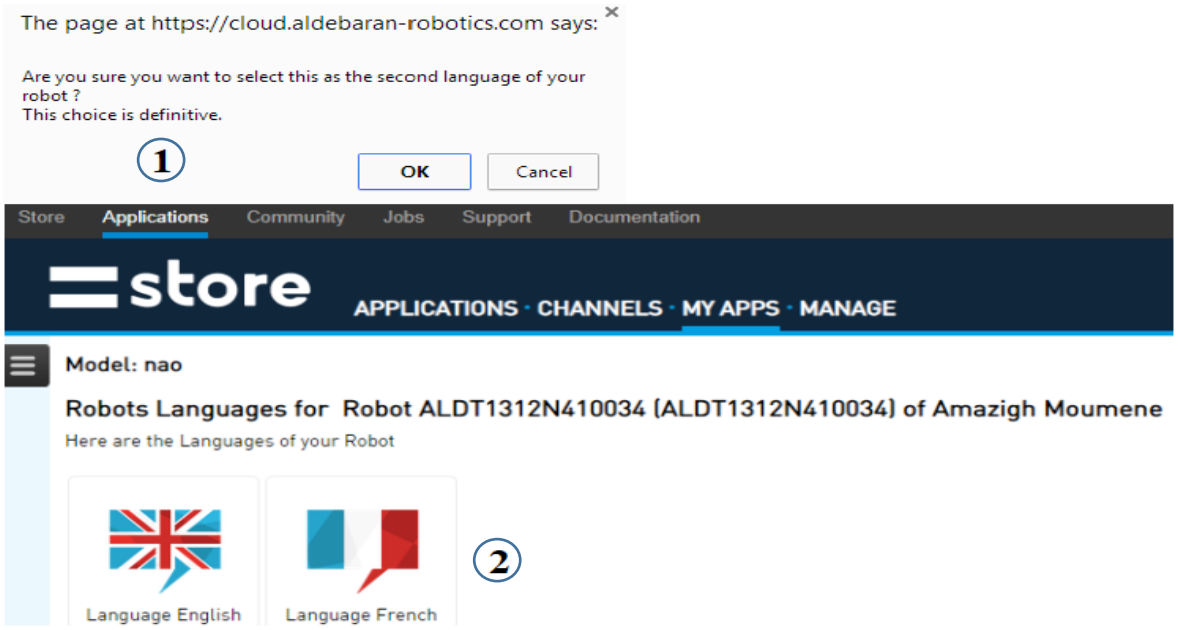


Figure II.20. Boite de dialogue demandant la validation.

Après la validation de la seconde langue, il est nécessaire de redémarrer NAO, et cela, en appuyant sur le bouton du torse pendant 3 secondes jusqu'à ce que NAO dise « GNUK GNUK ».

On revient maintenant sur la page web de NAO et on clique sur l'onglet de téléchargement. Il faut alors cliquer sur le bouton « **Tout mettre à jour** » pour finaliser l'installation. On peut suivre la progression de l'installation grâce une barre en couleur orange qui va se charger en vert.

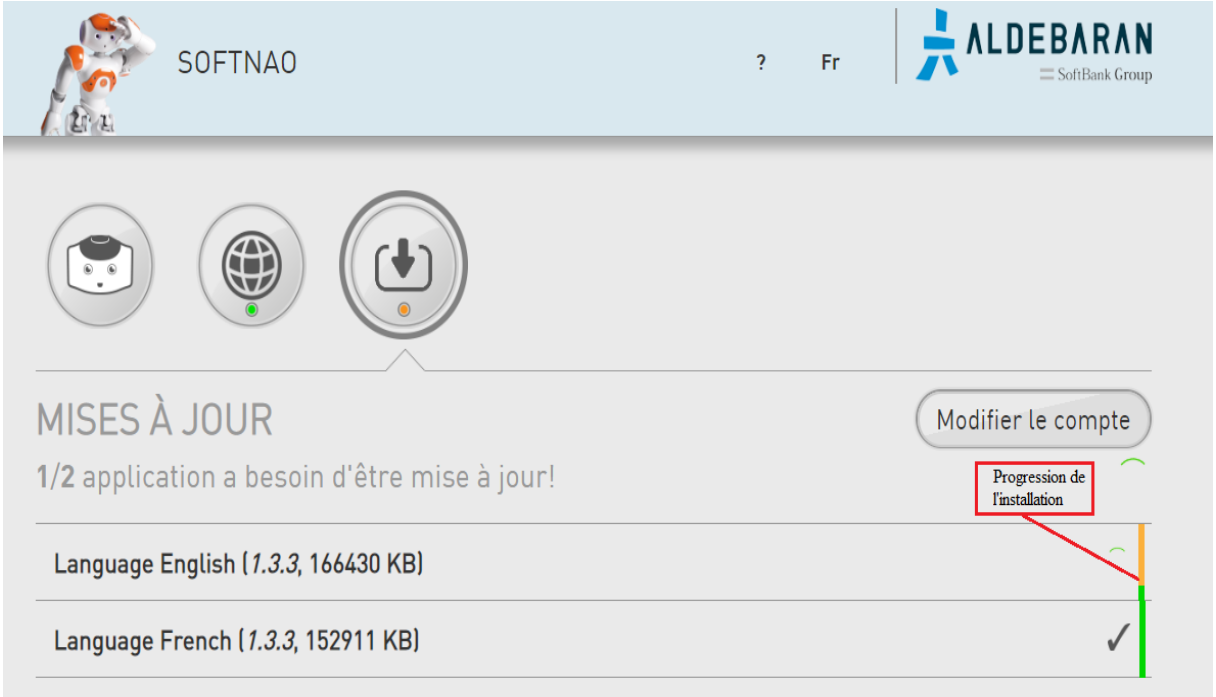


Figure II.21. Installation des mises à jours des langues disponibles.

Dans notre cas, nous avons installé le module de la langue française après avoir développé les premières applications en anglais. Le but était d'abord de réaliser l'essentiel du travail demandé, avant de s'engager dans la reconfiguration du robot. Car, même si on a réussi à installer ce module, on a tout de même fait face à quelques difficultés et quelques frayeurs, tels que la survenue des bugs (blocage du robot, perte de la voix après redémarrage et, par conséquent, difficulté d'avoir l'adresse IP dictée par le robot au démarrage) et la perte de connexion lors de l'opération de la mise à jour.

II.11. Conclusion

Maintenant, que nous avions présenté l'architecture de cet humanoïde, ses caractéristiques techniques et mécaniques, ses capteurs et moteurs et, enfin, sa configuration, il est évident pour nous de passer à la présentation de son logiciel (Choregraphe) qui va nous donner la possibilité de nous servir des ressources offertes par ce robot et de le manipuler.

Chapitre III : Présentation du logiciel Choregraphe

III.1. Introduction

Dans ce 3ème chapitre, on va répondre à la question suivante : « Comment se servir du logiciel Choregraphe et de ses fonctionnalités afin de créer des applications pertinentes et signifiantes ? ».

Choregraphe est un logiciel multiplateforme distribué avec le robot humanoïde. Il nous permet de réaliser des animations, des comportements et des dialogues et bien d'autres choses plus ou moins complexes.

III.2. Présentation de l'interface de Choregraphe [11]

Après l'installation du logiciel et de son exécution, on obtient cette fenêtre d'accueil dont la fonctionnalité de chaque sous-fenêtre est décrite dans ce qui suit.

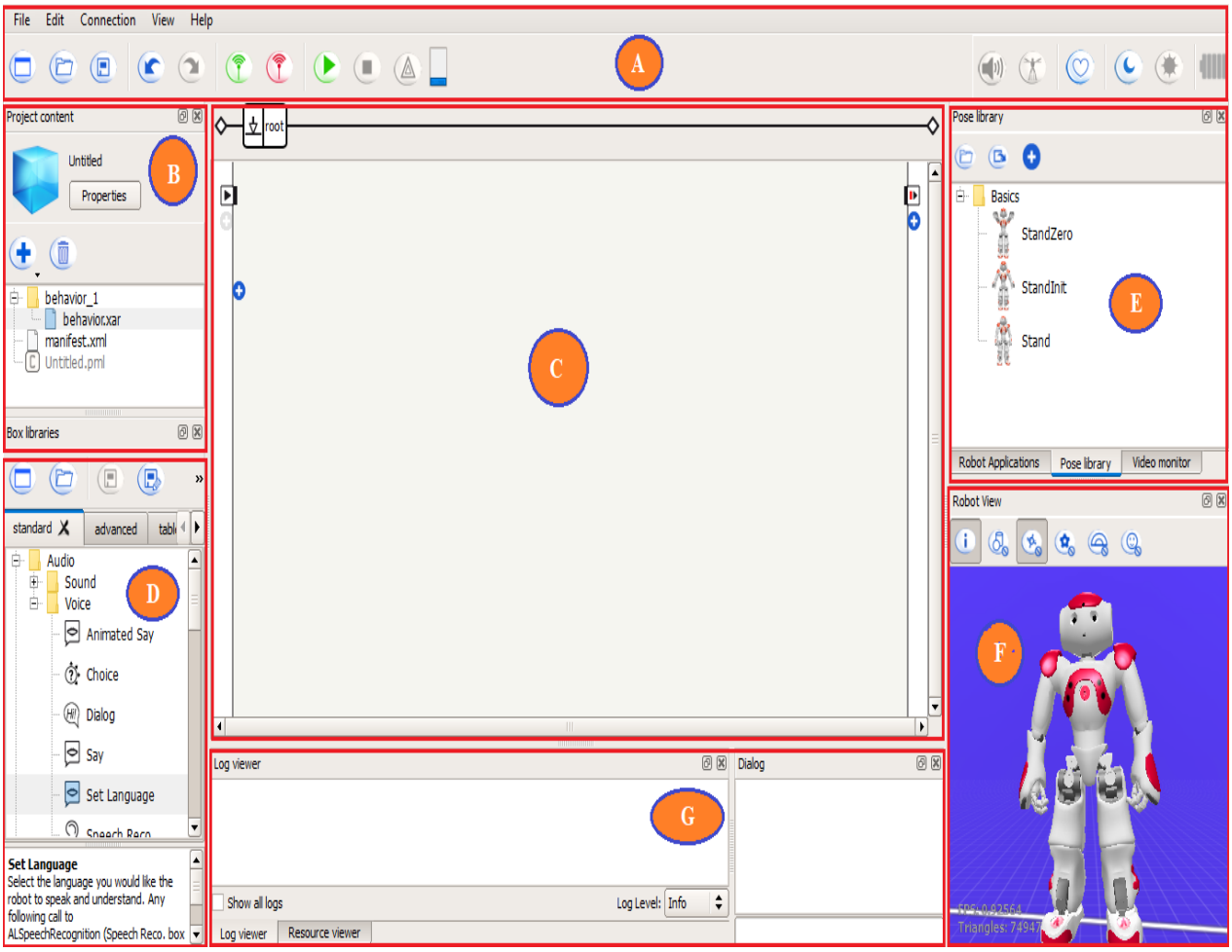


Figure III.1. Présentation graphique des différentes parties de Choregraphe.

III.2.1. Barre d'outils et de menus (A)

Les boutons d'actions et d'exécutions sont décrits dans Tableau III.1.

Tableau III.1. Description des fonctions des boutons de la barre d'outils.

Boutons	Fonctions
	Créer un nouveau projet, ouvrir ou sauvegarder un projet.
	Annuler et refaire une action ou une opération.
	Connecter ou déconnecter le robot réel ou virtuel. Connexion à **NAOqi** perdue et tentative de reconnexion.
	Bouton d'exécution et d'arrêt du programme.
	Bouton signalant les alertes et les erreurs.
	Barre indiquant la progression de chargement du programme. *Presque gris :* démarrage du chargement. *Monté et descente :* entrain de charger. *Bleu :* Programme chargé.
	Volume des haut-parleurs du robot.
	Activer / Désactiver « **Animation Mode** » permet d'enregistrer des mouvements de NAO facilement. Il s'allume en 3 couleurs : *Vert :* Animation Mode est désactivé. *Orange :* État intermédiaire soit en chargement ou déchargement. *Rouge :* Animation Mode activé.
	Bouton pour activer ou désactiver l'autonomie « **Autonomous Life** » de NAO.
	Bouton de mise en repos de NAO. Le mode **Stifness** (rigidité des articulations) est désactivé.
	« Mode Réveil » des moteurs. Le mode **Stifness** est activé
	État de charge de la batterie de NAO. *Vert :* la batterie est pleine. *Orange :* batterie à moitié chargée. *Rouge :* la batterie est faible.

III.2.2. Panneau de contenus du projet (B)

Lorsqu'on crée un nouveau projet dans Choregraphe, on y trouve systématiquement des détails concernant le projet créé. Il donne l'information sur le nom et l'identificateur (code ID), le type de fichiers utilisé et une option de configuration.

On peut importer depuis ce panneau des applications téléchargées via Internet et des musiques en cliquant sur l'icône  (Figure III.2).

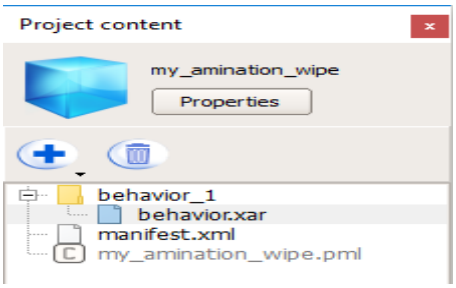


Figure III.2. Illustration du contenu du projet créé.

La figure III.2 nous donne quelques informations qui sont :

- **Behavior.xar :** C'est un dossier dans lequel sont stockés les comportements créés.
- **Manifest.xml :** En double-cliquant dessus, la même fenêtre que celle des propriétés s'ouvre pour effectuer une configuration sur le projet.
- **Titre.pml :** C'est un fichier portant le titre du projet.

III.2.3. Espace de travail (C)

C'est la plateforme où l'on modélise le comportement de NAO en glissant et en déposant dessus des **Boxes** (boites) et en les reliant sous forme d'un codage graphique (diagramme).

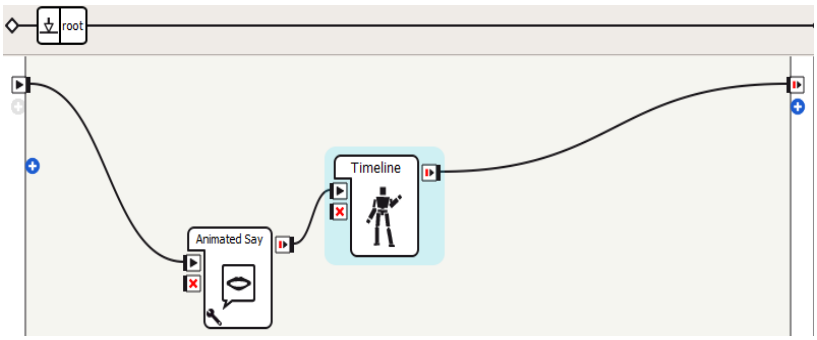


Figure III.3. Interface de travail de Choregraphe.

III.2.4. Bibliothèque des Box (D)

Elle contient différents types de boites répertoriées dans leurs dossiers respectifs. Il existe non seulement des boites permettant de réaliser des actions souhaitées, telles que la reconnaissance d'images et de voix, mais également, on peut réaliser des boucles d'exécutions à travers les boites adéquates. Si on clique une seule fois sur une boite, une petite description de la fonctionnalité de celle-ci s'affiche.

Afin de créer une nouvelle boite, on clique sur le bouton droit de la souris / Pavé tactile dans l'espace de travail \ ***Create a new box*** \ ***(Diagram, Timeline, Python, Dialog)***.

On a aussi la possibilité de rechercher manuellement les boites à travers les onglets de la bibliothèque **[14]**.

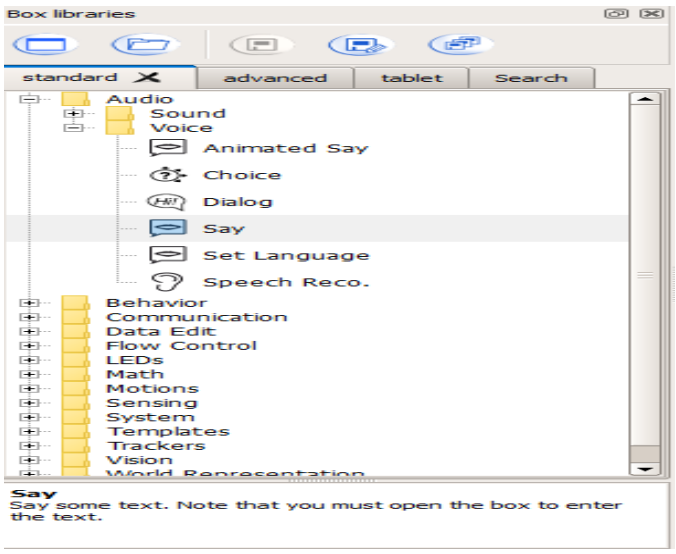


Figure III.4. Onglet de la bibliothèque des Boxes.

III.2.5. Bibliothèques des postures (E)

Elle contient des postures prédéfinies de NAO (StandZero, StandInit et Stand), comme l'on peut aussi enregistrer de nouvelles postures. On peut afficher la perception en temps réel des caméras à travers l'onglet « Video monitor » et on peut pour installer des applications sur NAO à travers l'onglet « Robot Applications ».

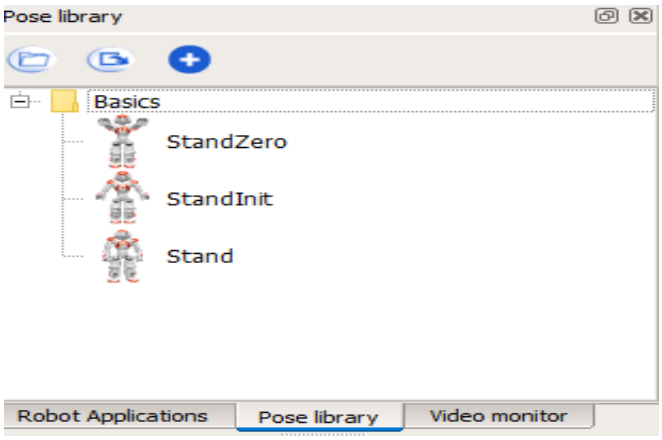


Figure III.5. Options disponibles dans la bibliothèque.

III.2.6. Vue du robot (réel ou virtuel) (F)

C'est une partie qui nous offre diverses options. On peut poursuivre le comportement de NAO selon différents angles de vue. L'autre option disponible, c'est la manipulation des parties du corps du robot avec la boite *Timeline*.

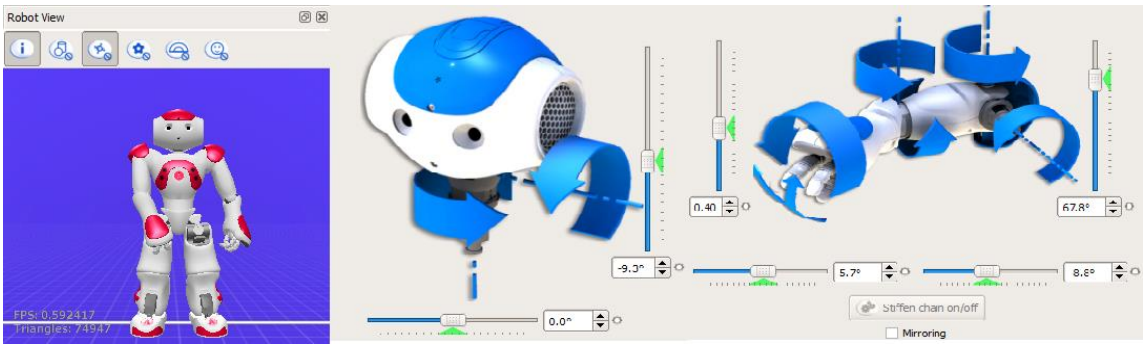


Figure III.6. Vue d'ensemble de la partie simulation et de celle de la maniabilité du corps.

III.2.7. Fenêtres de diagnostic et de dialogue (G)

La fenêtre de diagnostic fournie des informations sur le déroulement et l'évolution du programme exécuté pour signaler des avertissements et des erreurs. Quant à la fenêtre de dialogue, elle affiche la communication entre l'utilisateur et NAO.

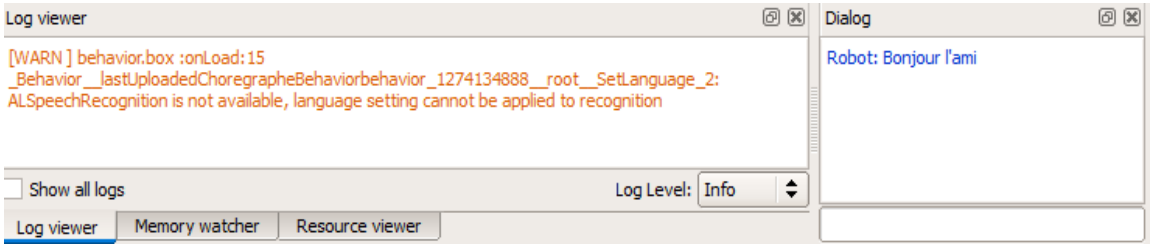


Figure III.7. Exemple de messages affichés après l'exécution d'un code.

III.3. Description des types de boites [10] [14]

Choregraphe propose quatre modèles de boites modifiables et qui sont : *Diagram*, *Timeline*, *Python* et *Dialog*. En effet, ces boites partagent quelques points en communs : des entrées et des sorties de différents types. Si on veut éditer une boite, il faut la sélectionner puis cliquer sur le bouton droit de la souris et cliquer sur **Edit Box** ou directement sur **Ctrl+E**.

III.3.1. Diagram Box

Il a pour fonction de contenir d'autres diagrammes afin de mieux structurer son programme. Voir la figure III.8.

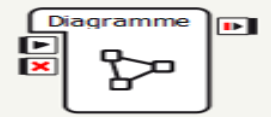


Figure III.8. Boite de Diagramme

III.3.2. Timeline

Si on souhaite synchroniser des mouvements avec d'autres tâches, il faut utiliser ce type de boite. Il affiche une sorte de blocs d'échantillonnages (samplers) dans lequel il est possible de modifier le temps d'échantillonnage.

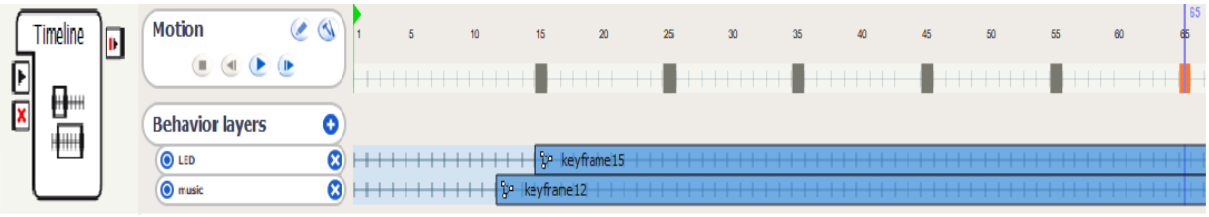


Figure III.9. Panneau de modélisation de mouvements de NAO intégré dans Timeline Box.

Deux méthodes d'enregistrement sont possibles : soit on introduit manuellement la valeur des angles d'articulations à partir de la fenêtre de simulation (**Motion**) de Choregraphe, soit à partir du robot NAO.

Pour pouvoir enregistrer directement à partir de NAO, on procède de cette manière :

- Dans la barre d'échantillonnage, on clique sur le bouton droit de la souris / Store joints in keyframe / Choisir entre les différentes parties de NAO proposées (**Whole Body** par exemple).
- En double-cliquant sur une partie du corps de NAO, la fenêtre **Motion** appraît.
- On clique sur **Stiffen chain on/off** (la couleur de l'icône devient rouge), ceci signifie que la rigidité de NAO est désactivée afin de pouvoir manipuler facilement une partie de son corps.
- Une fois terminé, on réactive la rigidité en cliquant encore une fois sur **Stiffen chain on/off** (la couleur de l'icône devient verte).

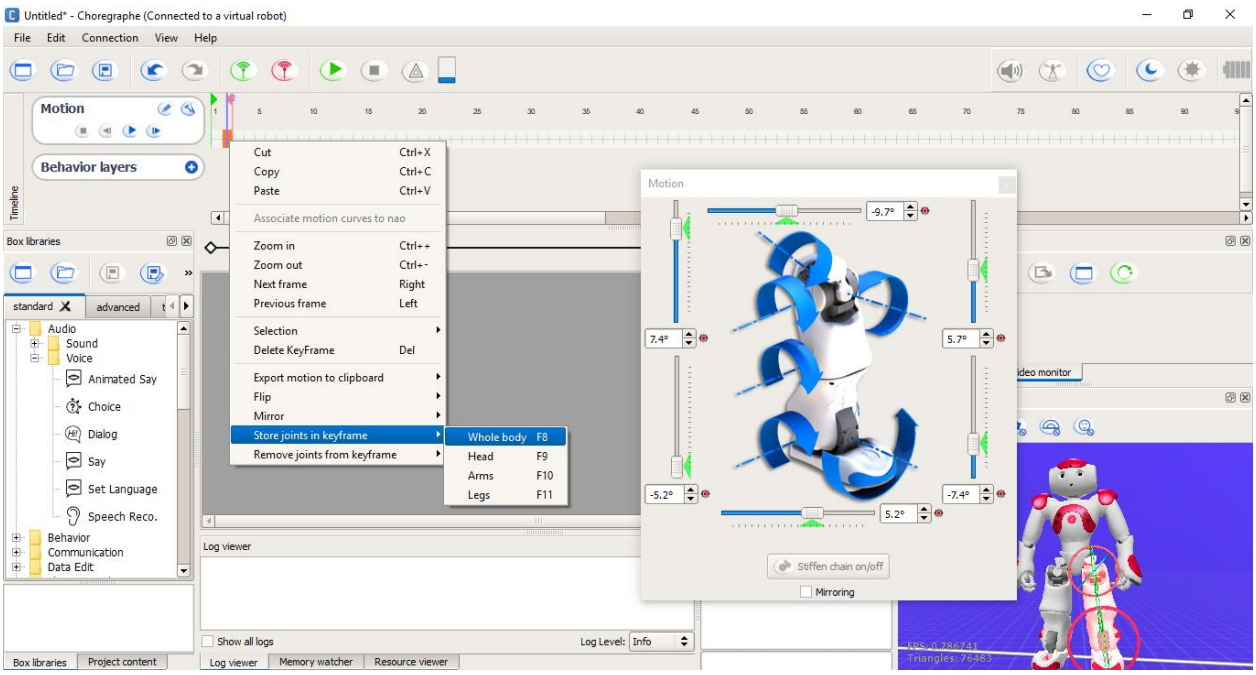


Figure III.10. Procédure d'enregistrement de comportements dans la boite Timeline.

II.3.3. Python Script

C'est un code écrit en langage python par défaut contenant des classes et des méthodes. Par conséquent, on peut instancier des objets ou des fonctions d'une classe principale (modules).

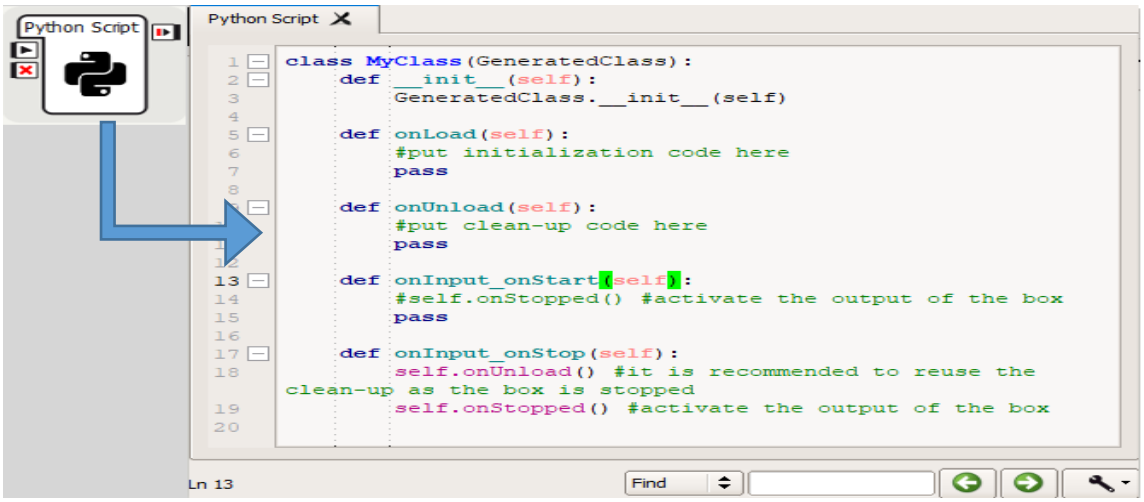


Figure III.11. Syntaxe générée par la boite Python Script.

III.3.4. Boite Dialogue

Cette boite propose le choix de langues avec laquelle on va communiquer avec NAO. Si la boite de dialogue n'est pas configurée, elle s'affiche en pointillés (Figure III.12). Elle s'affiche en traits continus une fois elle est configurée.

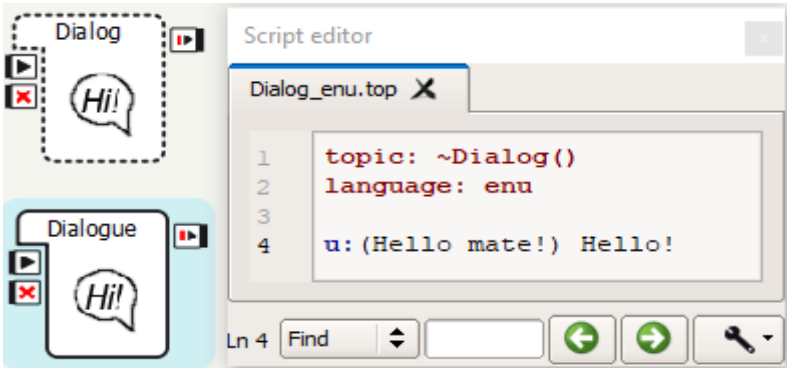
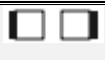
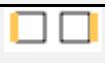
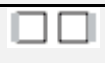


Figure III.12. Boites de dialogue (à gauche) et l'espace de programmation de la conversation (à droite).

III.4. Description des entrées / sorties [10] [13]

Il est important de savoir comment relier les boites entre elles, grâce aux entrées et sorties, pour que le programme réalisé fonctionne correctement. On résume les paramètres de ces points de connexions dans le Tableau III.2.

Tableau III.2. Description des symboles d'entrées / sorties.

Symbole	Nature	Description
Entrées		
	onStart	Si l'entrée est stimulée, la boite est démarrée.
	onStop	Si cette entrée est exécutée, la boite est arrêtée.
	onEvent	C'est une entrée qui n'a pas d'effet spécifique sur la boîte, il ne démarre et ne stoppe rien. Quand il est stimulé : La fonction *onInput_<input_name>* du script de la boîte est appelée. Le signal reçu sur l'entrée est transmis au diagramme de la boîte.
	ALMemory Input	C'est une entrée qui n'est visible que dans le schéma. Donc, vous ne pouvez pas la stimuler de l'extérieur de la boîte. Elle est stimulée à chaque fois que la valeur des données stockées dans *ALMemory* correspondant à cette entrée est mise à jour et qu'un événement est déclenché (mis à jour).
	onLoad	C'est une entrée qui n'est visible que dans le schéma et quand la boîte est de type *Timeline*. Donc, on ne peut pas la stimuler de l'extérieur de la boîte. Elle est seulement stimulée lorsque le diagramme a été chargé.
Sorties		
	onStopped	Lorsque cette sortie est stimulée (à partir du *Timeline*,du *Dialog* ou à partir du diagramme), la boîte s'arrête (désactivée). Par contre elle n'a aucun effet sur la boite *Python*.
	punctual	Cette sortie transmet seulement le signal entre les diagrammes et les sous-diagrammes.
Types d'entrées / sorties		
	Bang	Représente un évènement simple et ne porte que l'information stimulée.
	Number	Représente un évènement portant des informations de types float (réel), int ou bien sous forme d'un tableau de nombres.
	String	Représente aussi un évènement portant des informations mais de type *String* ou d'un tableau de type chaines de caractères.
	Dynamic	Il représente les deux types d'évènements : simple ou porteur d'informations.

III.5. Conclusion

Selon le contenu de ce chapitre, on comprend que Choregraphe est un logiciel qui offre diverses options à l'utilisateur pour qu'il puisse créer des animations en se référant à une certaine logique de connectivité et de synchronisation tout en respectant la structure exigée.

On a décrit aussi, globalement, ses fonctionnalités avec lesquelles on va mettre en œuvres des applications suggérées dans le prochain chapitre.

Chapitre IV : Développement d'applications

IV.1. Introduction

Notre objectif principal dans ce chapitre est de décrire les applications créées lors des essais effectués sur NAO. Dans chaque application, on a imaginé des situations où le robot va reproduire les tâches programmées. Bien évidemment, le robot à la capacité de faire des tâches complexes et remarquables, ainsi le choix revient à l'utilisateur de faire ce qu'il souhaite.

De notre côté, on a choisi à travers les premières applications de tester la réactivité des capteurs de NAO et en même temps, de comprendre et d'expliquer leurs modes de fonctionnement. Dans le cas où l'on souhaite réaliser des applications qui ne nécessitent pas la mise en marche des moteurs du robot, il est conseillé d'activer le mode repos **(Rest).** Car, cette option économise l'autonomie de la batterie **[14]**.

IV.2. Application 1 : Réponse de NAO aux sensations tactiles

Cette application consiste à tester les capteurs tactiles de NAO : tête, main gauche et amortisseurs des pieds. Pour ce faire, on a associé à chaque sortie du capteur une phrase que le robot va prononcer en guise de bon fonctionnement.

Au démarrage, NAO va prononcer : « *I'm ready for test* ». Puis, en touchant une partie de ses capteurs, il prononcera l'une des trois phrases suivantes : « *Hi, do not touch my Bumper* », « *Hi, do not touch my Head*» et « *Hi, do not touch my hand*».

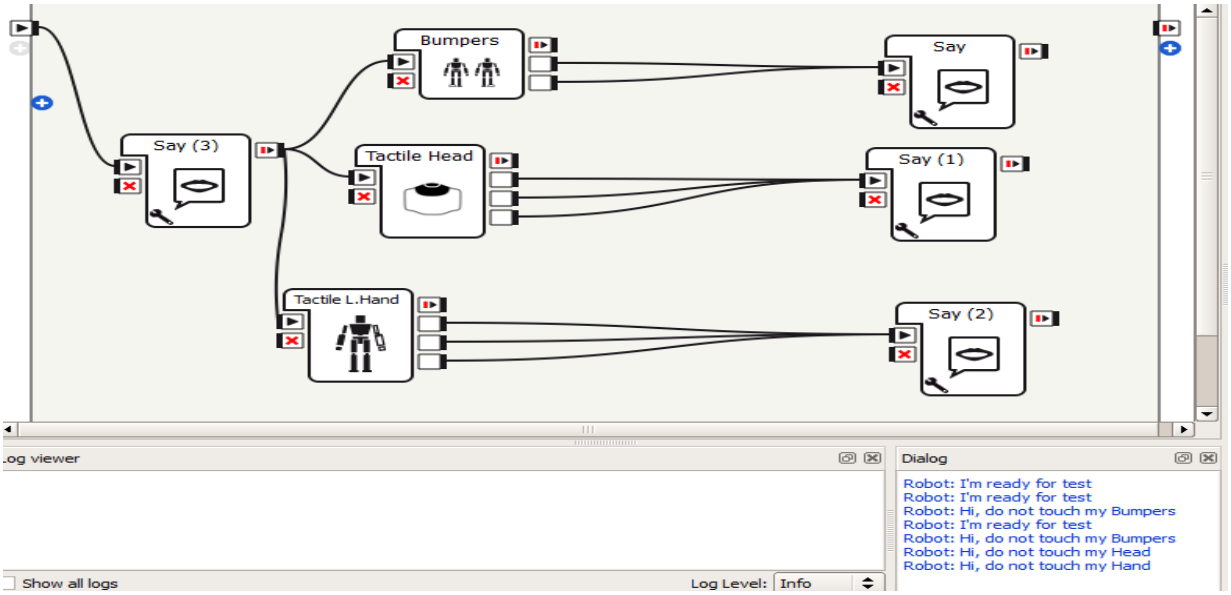


Figure IV.1. Essai effectué sur les capteurs tactiles.

Nous avons illustré le contenu des boites « Say » dans la partie des annexes (Voir Annexe A.1).

IV.3. Application 2 : Capacités de communication et de dialogue verbal de NAO

Comme nous l'avions mentionné au préalable, on peut tenir une conversation avec NAO. C'est pour cela que nous avions choisi de concevoir une application simple contenant quelques syntaxes permettant d'animer une conversation afin de pouvoir imiter le concept du dialogue de l'être humain.

Dans cette application nous avons choisi la langue anglaise comme moyen de communication qui est disponible par défaut sur le robot réel. Toutefois, différentes langues sont disponibles sur Choregraphe (Robot virtuel).

Les boites utilisées sont : **Set Language** et **Dialog**. Le bouton apparent en bas et à gauche de la boite **Set Language** permet d'afficher la fenêtre **Set parameters** pour le choix de la langue. La figure IV.2 montre comment ajouter une nouvelle conversation et paramétrer la boite de dialogue et son type d'entrées et de sorties.

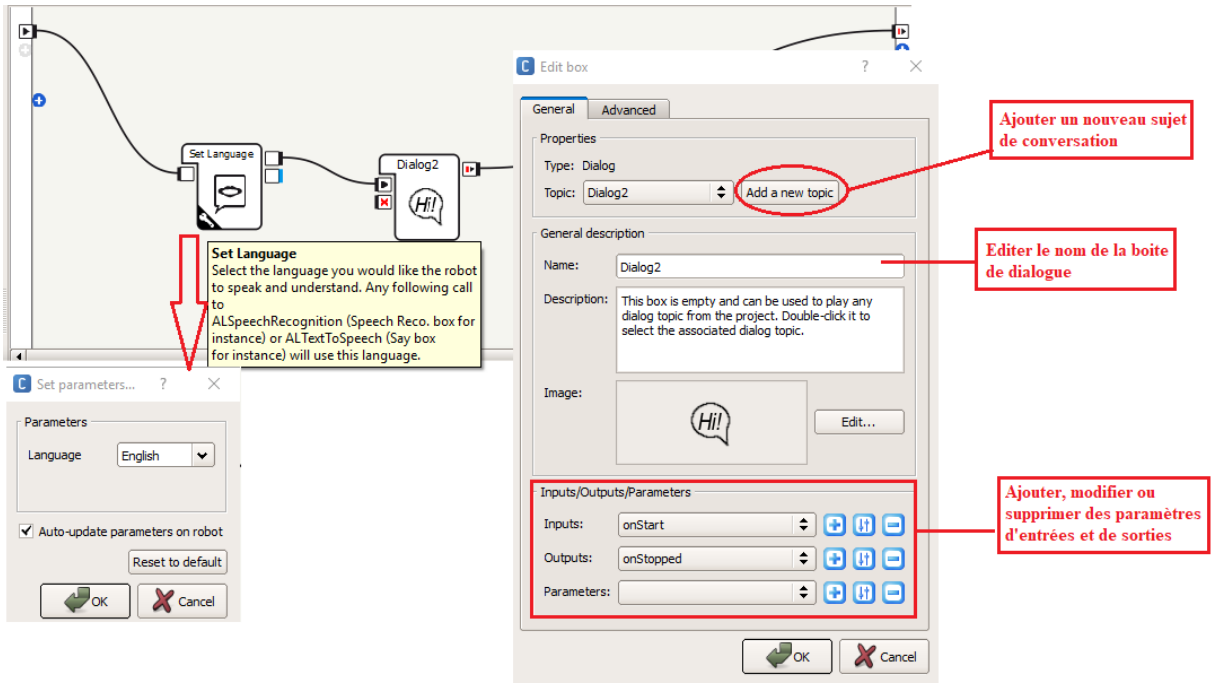


Figure IV.2. Description de la configuration des boites utilisées pour créer un dialogue.

A la fin de la configuration, il faut afficher la fenêtre **Script editor** afin de pouvoir programmer une conversation. La procédure est simple : dans la fenêtre **Project content** on trouve dans le dossier portant le nom de la boite de dialogue. Dans ce dossier on trouve deux

fichiers. Le fichier qui se termine par l'extension : **.dlg** permet de choisir la langue. On double-clique sur le fichier qui se termine par l'extension : **_enu.top** (c'est une abréviation de **english.topic**), permet d'accéder à la fenêtre Script editor.

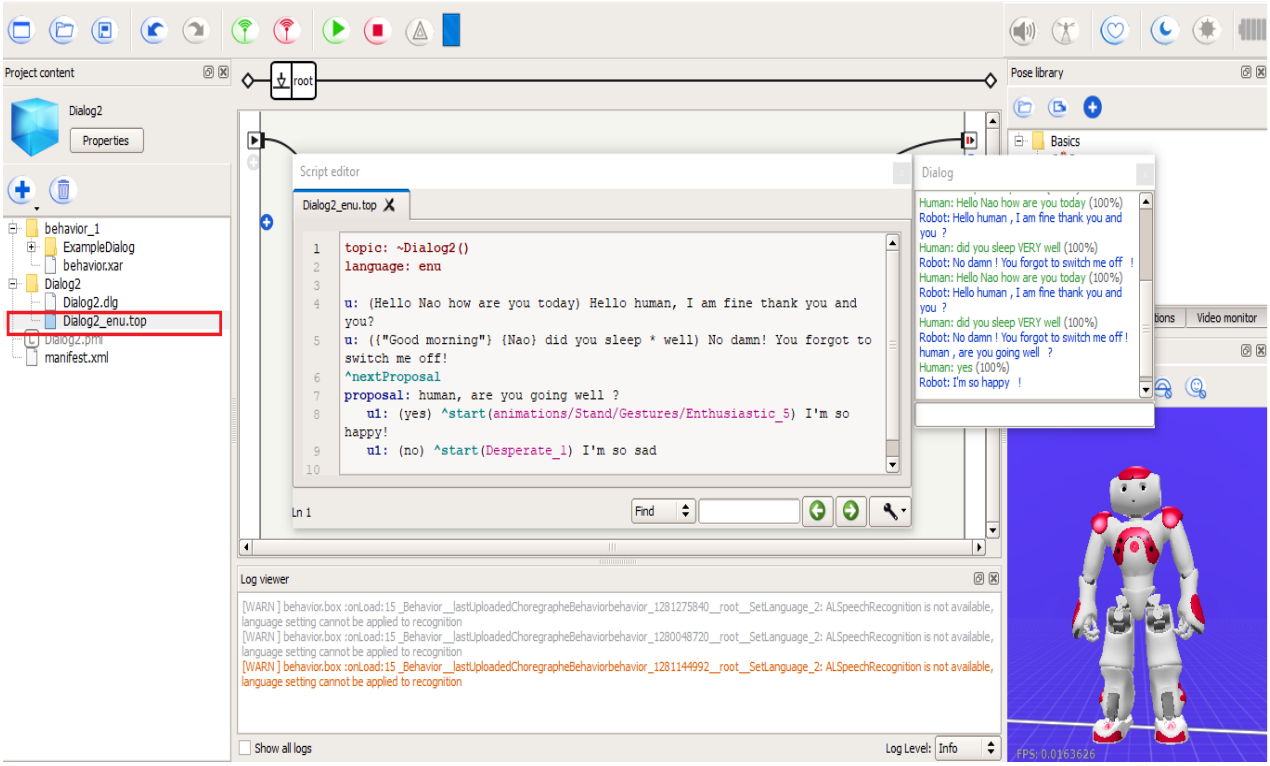


Figure IV.3. Programmation d'une conversation et test de réceptivité par NAO.

La syntaxe générale nommée **User rule** est la suivante : « **u:** *(Entrée humaine) Réponse du robot* ».

Il existe un très grand nombre de syntaxes disponibles dans la bibliothèque **QiChat** qui se trouve dans la documentation installée avec Choregraphe.

On cite quelques fonctions qu'on peut utiliser dans le script.

Délimiteur " " : Donne la possibilité d'écrire une phrase entière.

Accolades { } : Ce qui est écrit à l'intérieur sera considéré comme une entrée facultative.

Choix [ ] : Attribue le choix à l'utilisateur et/ou à NAO de dire ce qui est programmé.

Wildcar * : Plusieurs mots ou phrases que l'on peut insérer.

concept : C'est une liste qui peut contenir des mots et/ou des expressions. L'entrée peut être humaine comme elle peut celle de NAO. Pour appeler un concept, on écrit : ~(nom_concept).

proposal : Cette fonction permet au robot de dire quelque chose.

^nextProposal / ^previousProposal : Elles sont reliées à la fonction **proposal** car elles permettent de défiler entre les propositions déclarées.

^start / ^stop : Démarrer ou arrêter une animation ou un comportement du robot.

^wait : Attend jusqu'à ce qu'une animation est achevée **[11].**

On peut suivre le déroulement de la conversation entre l'humain et le robot à partir de la fenêtre de dialogue (Figure IV.3).

On a pu écrire un autre dialogue en langue française dans lequel on a ajouté d'autres syntaxes qui permettent à NAO d'accompagner son message avec des animations (Voir Annexe A.2).

IV.4. Application 3 : Capacité de reconnaissance visuelle de NAO

NAO est doté d'un système d'apprentissage de nouvelles figures, d'objets et de voix. On a conçu une application démonstrative des procédures à suivre pour permettre à NAO d'acquérir et d'ajouter à sa base de données une image d'une boite d'emballage de marque **Skymax**, qu'il va nommer par la suite.

1) Il faut s'assurer que Choregraphe soit connecté à NAO.
2) Cliquer sur l'onglet **Video Monitor,** puis cliquer sur **Play** pour afficher la vision du robot.
3) Il faut bien placer un objet devant le champ de vision NAO.
4) En cliquant sur le bouton **Learn** , un décomptage de 4 à 0 secondes va commencer pour qu'une image soit capturée.
5) En cliquant sur l'image, le logiciel propose de dessiner des contours de l'objet par la technique de segmentation.

Dès qu'on termine la segmentation, la fenêtre de dialogue **Object tags** s'ouvre automatiquement et demande des renseignements sur l'image capturée (nom et côté). Une fois terminé, on clique sur OK.

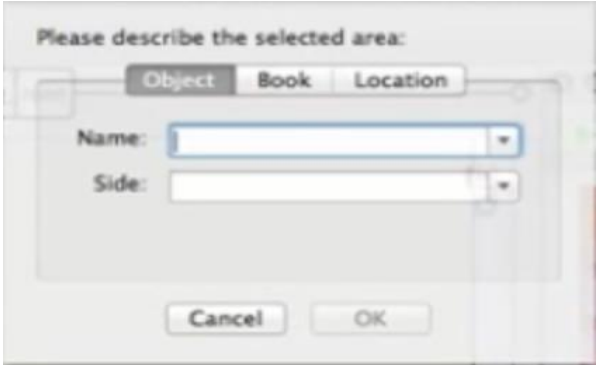


Figure IV.4. Fenêtre Object tags.

A présent, on a le choix entre sauvegarder la donnée dans l'ordinateur et cela en cliquant sur (**Export Vision Recognition Database**), ou bien de l'envoyer directement vers le robot en cliquant sur (**Send current vision recognition database to NAO**) **[12][13]**.

Lorsque nous avons achevé cette opération, nous devions vérifier si le robot allait reconnaitre la boite.

Nous avons utilisé deux boites : **Vision Reco.** et **Say Text**. La boite « Say Text » se trouve à l'intérieur de la boite « Say » qu'on va copier et connecter à « Vision Reco. » tel qu'il est montré dans la figure IV.5.

Dès qu'on exécute le programme, il faut mettre l'objet devant les caméras de NAO. Si l'objet en question est reconnu, alors NAO prononcera son nom. Dans notre cas, le nom attribué à la boite est « Front Box ».

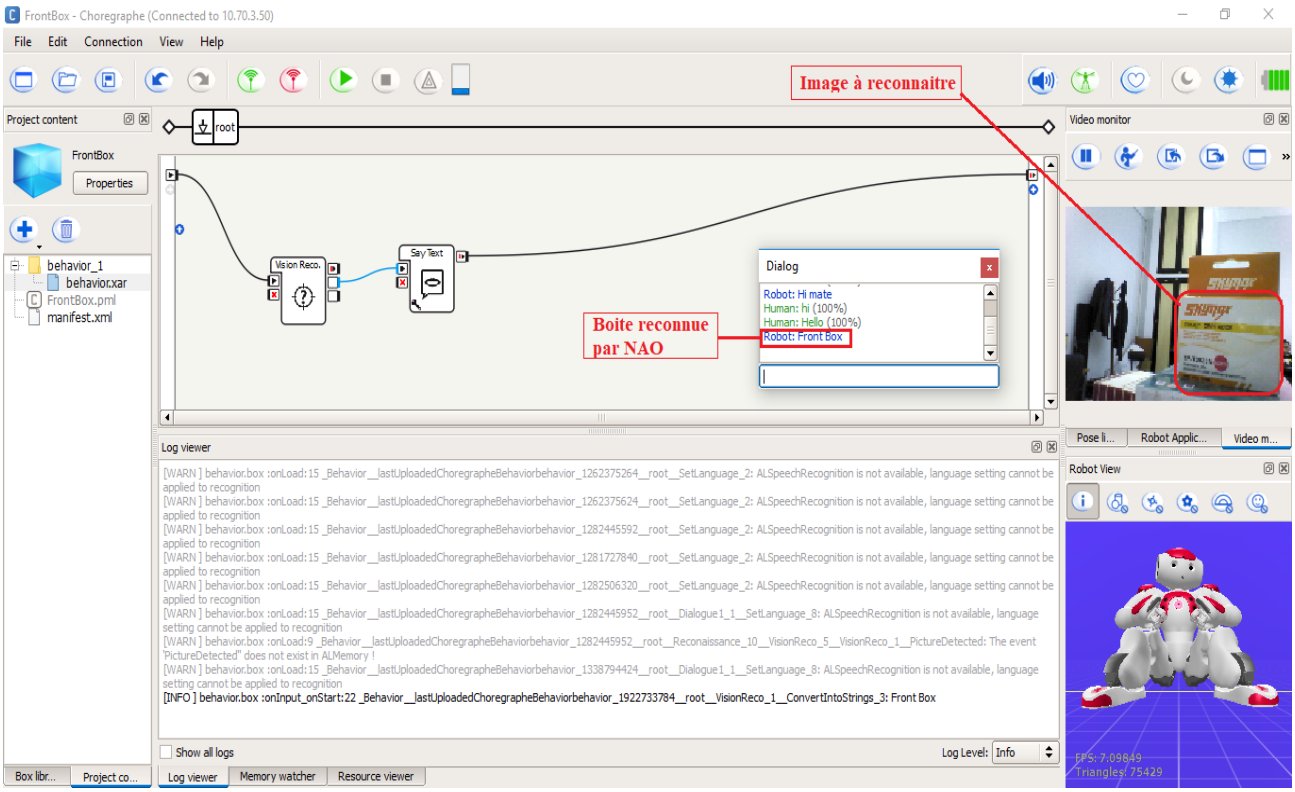


Figure IV.5. Programme de reconnaissance d'objet.

IV.5. Application 4 : Mobilité et réaction de NAO

Nous allons montrer dans cette application comment NAO se déplace dans l'environnement et quelles sont les boites à utiliser pour générer ce mouvement.

Pour faire avancer le robot droit devant, il faut insérer un valeur positive dans la variable X et respectivement, une valeur négative pour le faire reculer, sachant que la valeur de $Y = 0$.

Dans le cas où l'on insère une valeur positive dans la variable Y, le robot avancera latéralement vers la gauche, et respectivement à droite si cette valeur est négative, sachant que $X = 0$.

Si on attribue des valeurs à X et Y différentes de 0 en même temps, alors le robot avancera en diagonale.

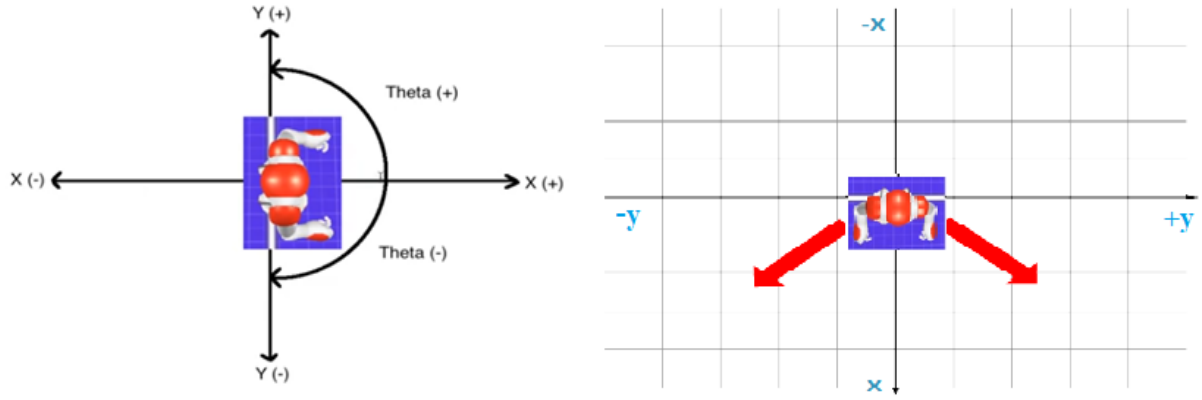


Figure IV.6. Déplacement relatif de NAO [11].

Il existe deux boites pour le déplacement de NAO :

- La boite **Move To** : Déplacer le robot vers un point configuré par rapport à son emplacement initial.
- La boite **Move Toward** : Déplacer le robot dans la direction définie dans les paramètres, sauf que le robot ne s'arrêtera pas de se déplacer tout seul. Soit on définit x, y et thêta sur 0, soit on arrête directement la boîte.

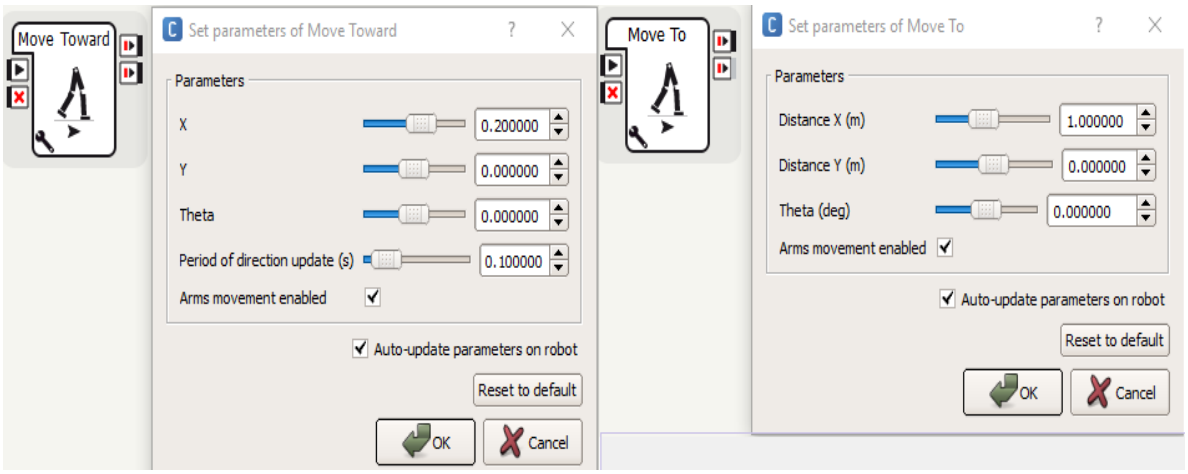


Figure IV.7. Paramètres des boites pour le déplacement.

En se basant sur ce modèle, nous avons pu développer une application où le robot va chercher un objet et le distribuer. Nous avons intégré une petite conversation où le robot va demander un stylo à une personne, le prendre, revenir à la position initiale pour enfin le distribuer. Cela sous-entend systématiquement d'ajouter quelques animations grâce à la boite *Timeline*.

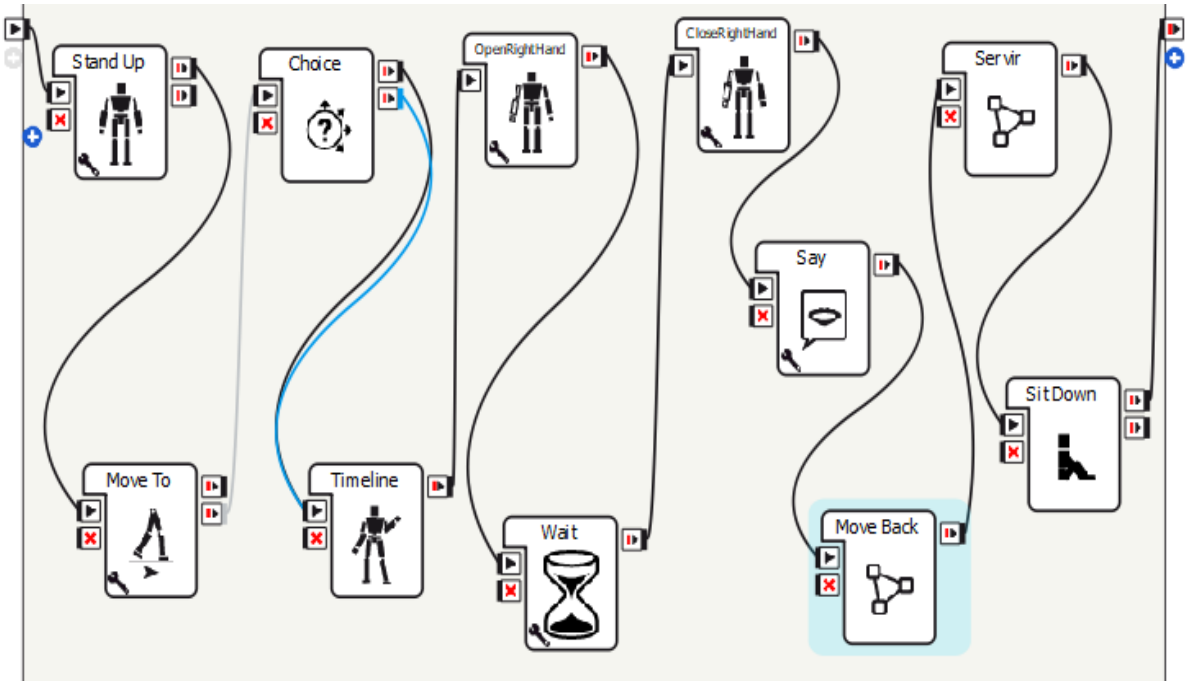


Figure IV.8. Programme de distribution d'objets.

Afin de rendre le programme plus lisible, on a regroupé les détails concernant les sous-diagrammes dans l'Annexe A.3.

IV.6. Application 5 : Evitement d'obstacles

L'application est assez simple à réaliser, puisque c'est la boite « **Obstacle Avoidance** » qui prend en charge la navigation de NAO sans se heurter à des obstacles via ses capteurs ultrasons. Par conséquent, son champ de détection, de portée égale à 0.5 m, est limité que vers l'avant du robot (pas de capteurs derrière).

D'après le test qu'on a effectué, on a compris que la procédure fonctionne ainsi :

D'abord, le robot avance droit devant. Si un obstacle est présent de 20 à 50 cm, le robot va reculer rapidement et tourne à droite pour essayer de marcher à nouveau. Par contre si la distance de détection est inférieure à 20 cm , le robot recule de manière progressive (séquentielle, petits pas) et plus lente (Figure IV.9).

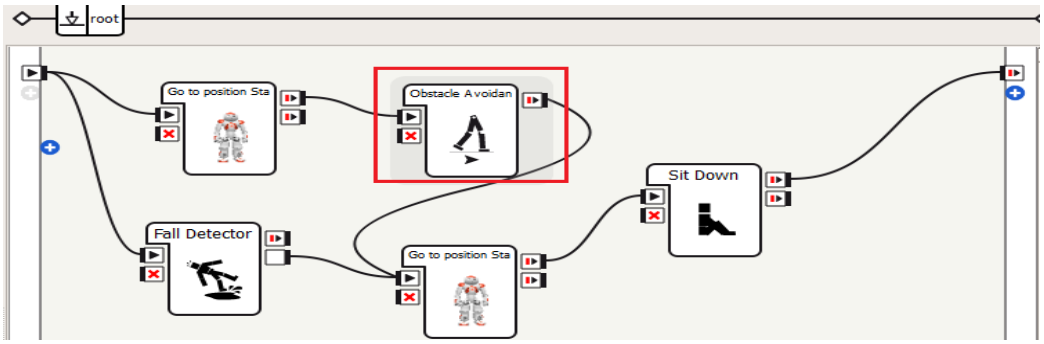


Figure IV.9. Programme d'évitement d'obstacles.

IV.7. Application 6 : Commande vocale

Cette application a été développée dans le but de contrôler de simples mouvements de NAO avec la voix. Des blocs de temporisation (Wait) ont été introduit pour permettre au robot de comprendre avant d'agir. Au début, lorsqu'on a attribué des valeurs de temporisation entre 1 et $5s$, le robot fusionne plusieurs comportements en même temps et finit par perdre son équilibre. Pour résoudre ce problème, on a introduit une temporisation (de $10s$ dans notre essai) avant d'affecter la commande récupérée depuis la boite de dialogue. On a remarqué que même avec cette temporisation, le robot exécute toujours une animation imitant le comportement humain en posture d'attente. On a alors ajouté des blocs « Stand up » (posture debout) après les boites « Wait ». Ceci permet au robot de transiter, en toute sécurité, entre le comportement d'attente et l'exécution des mouvements demandés verbalement.

Il faut également que le nom de chaque sortie de la boite de dialogue (à configurer manuellement) soit similaire à celui utiliser dans le script editor (Voir Annexe A.4).

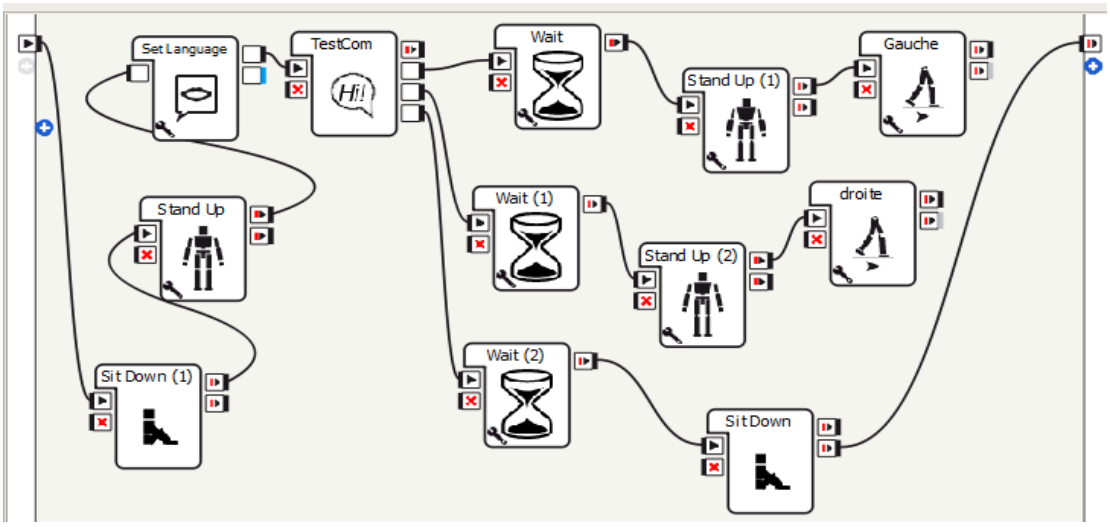


Figure IV.10. Application de la commande vocale.

IV.8. Application 7 : Programmation de NAO avec Python

Dans le chapitre trois, nous avons mentionné qu'on pouvait programmer NAO avec la boite *Python Script*, mais il existe tout de même une autre alternative, c'est la programmation avec Python SDK (Software Development Kit) dans lequel on va installer le module NAOqi.

Installation :

- Télécharger et installer Python $2.7 - 32\ bits$ depuis le site http://python.org/download/.
- Télécharger et installer le module « `pynaoqi-python-2.7-naoqi-x.x-win32.exe` » depuis le site Aldebaran Community.

Pour vérifier qu'on a bien effectué la procédure d'installation, on exécute « IDLE (Python GUI) » ou « Python (Command line) » ensuite, on introduit la syntaxe suivante : « *from naoqi import ALProxy* ». S'il n'y a aucun message d'erreur qui apparait, alors tout fonctionne correctement **[11]**.

Nous avons écrit un code simple en langage python qui permet à NAO de parler, d'affirmer une réponse en faisant un geste avec sa tête et enfin, marcher vers une position donnée. (Voir Annexe A.5).

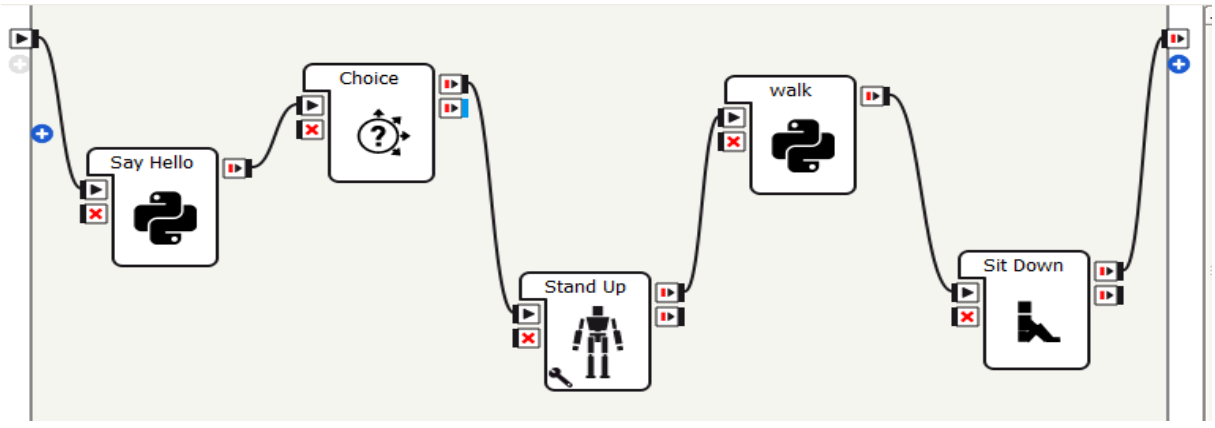


Figure IV.11. Diagramme constitué de programmes écrits en Python Script.

La figure IV.12 indique les deux possibilités de programmation de NAO.

Si on utilise directement la boite Python Script, on écrit la syntaxe dans la section **A** : `self.tts = ALProxy('ALTextToSpeech')`, puis on fait appel à la fonction **self.tts** dans la section **B** de la figure IV.12 en instanciant l'action **say** contenant le message. Par la suite, on ajoute la

syntaxe self.onStopped() dans le but de stopper l'exécution de la boite et passer la consigne à la boite suivante.

Généralement dans Python, le premier argument d'une méthode est appelé « **self** ». Ce n'est qu'une convention, puisque ce nom est un signe pour que le programme soit lisible pour les autres programmeurs en langage Python **[15].**

Dans le cas où l'on choisit de coder avec IDLE Python, on doit d'abord importer la fonction **ALProxy** depuis le module **naoqi** installé auparavant. Les concepteurs ont créé un « *Broker* » qui est un programme qui s'exécute à l'intérieur du robot NAO. Il joue le rôle d'un coordinateur ou bien d'un intermédiaire. Il a accès à toutes les informations concernant les modules et les méthodes et les mets à disposition de l'utilisateur. La particularité du Broker, c'est qu'il ne donne pas un accès direct vers les modules (ou méthodes), mais relaie l'information via un Proxy vers ces modules. Le Proxy dans ce cas, est une sorte de clones ou bien d'une copie qui permet de manipuler le robot depuis le PC.

Ensuite, on introduit l'adresse IP de NAO et le port par défaut 9559 pour assurer la communication.

Ainsi, on demande au Broker avec l'adresse IP et le port de nous donner le proxy du module de « AlTextToSpeech », puis on affecte (stocke) cette donnée dans la variable « **tts** » (figure IV.13).

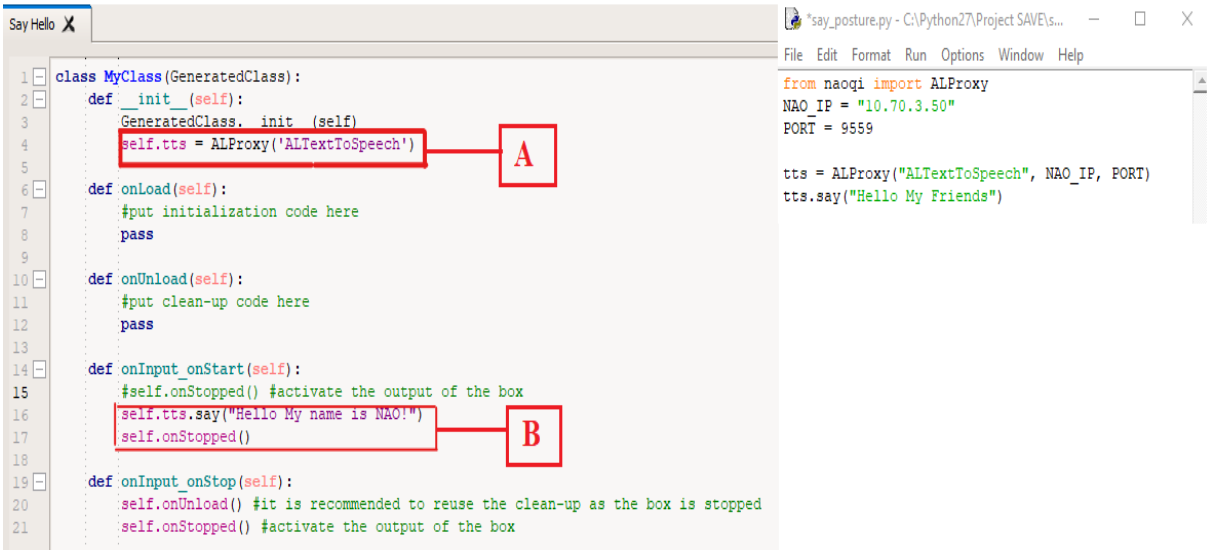


Figure IV.12. Programme écrit avec Python Script (Choregraphe) à gauche et Python IDLE à droite.

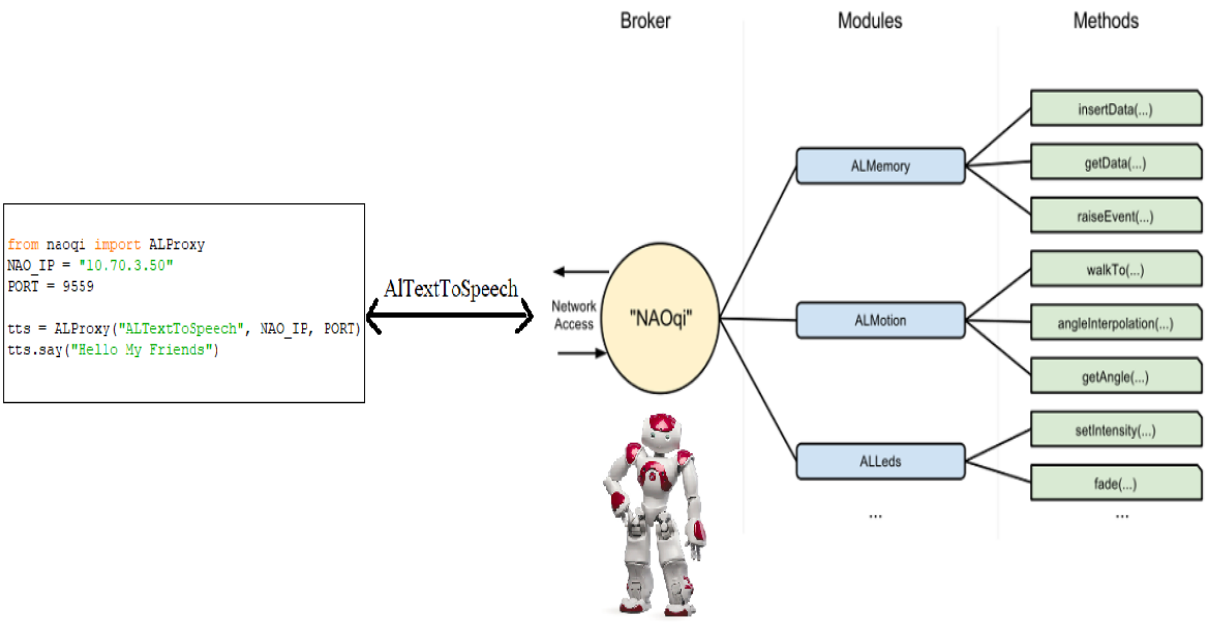


Figure IV.13. Coordination entre NAOqi et Python [11] [12].

IV.9. Conclusion

Dans ce dernier chapitre nous avons exploré les différentes techniques de programmation de NAO avec le logiciel Choregraphe, et ce, en réalisant diverses applications qui se sont déroulées avec succès.

Au début, on a testé la sensibilité de ses capteurs tactiles. Puis, on a testé sa capacité de communication avec le script. Ensuite, on a développé une application simple qui a permis de voir la fonctionnalité de la reconnaissance visuelle d'objets de NAO. Le seul inconvénient, c'est que le système présente quelques latences afin de reconnaitre l'image enregistrée au préalable.

Après, on a démontré à travers un ensemble de comportements (dans la 4ème application), sa mobilité et sa réaction dans son environnement. Par la suite, on a testé ses capacités d'évitement d'obstacles. On a aussi démontré comment NAO exécute des tâches en se servant de la communication vocale. Enfin, on s'est intéressé à la programmation avec le langage Python, qui s'est avéré être un outil flexible et puissant dans le cas où l'on souhaite contrôler le robot sans avoir recours à Choregraphe.

Conclusion générale

Conclusion générale

Ce mémoire de Master présente une étude sur le robot humanoïde NAO. Après une exploration générale du domaine de la robotique, on s'est intéressé aux robots humanoïdes en décrivant leurs caractéristiques générales. La suite du travail concerne l'étude des ressources matérielles et logicielles du robot NAO, en particulier le logiciel Choregraphe.

Le fond du travail a été entamé par l'étude de la configuration et les protocoles de communications du robot. Après la mise en marche, on a examiné les compétences du robot à travers une étude approfondie des fonctionnalités du logiciel Choregraphe.

La dernière partie de notre travail est dédiée au développement de quelques applications démonstratives permettant de tester et de mettre en évidence les compétences du robot, et ce en utilisant des logigrammes à base des fonctions (boites) du logiciel, ou une programmation en langage Python.

Ce travail nous a permis de consolider des connaissances acquises durant notre cursus de formation et de découvrir et d'acquérir de nouvelles connaissances (les actionneurs, les capteurs, interfaçage et protocoles de communications, langages de programmation des robots).

Il reste tout de même beaucoup d'éléments à étudier et à apprendre avec NAO, du moment que l'on peut le programmer avec d'autres langages de programmation tels que le C++, Java et JavaScript. Il offre également l'opportunité de tester ses propres programmes dans un monde virtuel en installant le logiciel Webots qui est une plateforme Open Source.

Références bibliographiques

Références bibliographiques

[1] Ichbiah Daniel. "*Robots, genèse d'un peuple artificiel*". Minerva, 2005. 540p.

[2] Baddoura-Gaugler, Rita. *"L'homme et le robot humanoïde: Transmission, Résistance et Subjectivation*." Thèse de doctorat, Université Paul Valéry-Montpellier III, 2013.

[3] Russell H., et al. *"Medical robotics and computer-integrated surgery.*" *Springer handbook of robotics*. Springer, Cham, 2016. 1657-1684.

[4] Paul J. "*Military robots and drones: a reference handbook*. " Springer., ABC-CLIO, 2013.

[5] Sanchez-Riera, Jordi. "*Developing Audio-Visual capabilities of humanoid robot NAO*." Thèse de doctorat. Université de Grenoble, 2013.

[6] Salotti, Jean-Marc. "La robotique humanoïde." *Questions internationales,* (2018): 90-92.

[7] Cruz, Luis. "*Humanoid Robot Nao: Developing Behaviours for Soccer Humanoid Robots*." LAP Lambert Academic Publishing, 2013.

[8] Asano, Yuki, Kei Okada, and Masayuki Inaba. "*Design principles of a human mimetic humanoid: Humanoid platform to study human intelligence and internal body system*." Science Robotics 2.13 (2017).

[9] Giaretta, Alberto, Michele De Donno, and Nicola Dragoni. "*Adding salt to pepper: A structured security assessment over a humanoid robot*." Proceedings of the 13th International Conference on Availability, Reliability and Security. ACM, 2018.

[10] Joachim.A, Rémi.C, Guillaume.G, Chunhui.L, Wei.G, Benjamin.P, Sébastien.S, Cedric.V. : "*Guide Utilisateur du robot humanoïde NAO. Projet d'Ingénierie du Logiciel en Informatique*". École Polytechnique de l'Université de Tours. France, 2012.

[11] Documentation du robot NAO *(Aldebaran documentation)* installé avec le logiciel Choregraphe, version : 2.1

[12] Wang, Chaoyue. "*Behaviour Design of NAO Humanoid Robot Playing Tic Tac Toe Game.*", Technology and Communication, Vaasan Ammattikorkeakoulu University of Applied Sciences, Finland. 2015.

[13] Guide : Démarrer avec NAO version 2.1.4

[14] Karsenti, T., Bugmann, J. et Parent, S. "*Le robot NAO en éducation. Guide de l'élève*". Montréal. CRIFPE. 2017.

[15] Python 2.7.16 documentation.

[Web 1]: https://www.oxfordlearnersdictionaries.com/definition/american_english/robot, consulté le 12 avril 2019.

[Web 2]: **h**ttp://www.atilf.fr/ressources/grand-public/mots-de-la-science/cadres/mots/tlfi/robot.htm, consulté le 12 avril 2019.

[Web 3] : Étienne DOMBRE, « **ROBOTIQUE CHIRURGICALE** », *Encyclopædia Universalis* [en ligne], consulté le 13 avril 2019. URL : http://www.universalis.fr/encyclopedie/robotique-chirurgicale/

[Web 4]: https://www.bostondynamics.com/spot-mini, consulté le 14 avril 2019.

[Web 5]: https://www.bostondynamics.com/atlas, consulté le 14 avril 2019.

[Web 6]: https://asimo.honda.com/asimo-specs/, consulté le 14 avril 2019.

[Web 7]: https://www.sciencesetavenir.fr/high-tech/kengoro-un-robot-fait-de-108-moteurs-et-d-un-peu-de-sueur_119604, consulté le 14 avril 2019.

[Web 8] : https://www.softbankrobotics.com/emea/fr/pepper, consulté le 15 avril 2019.

[Web 9] : https://www.softbankrobotics.com/emea/fr/nao, consulté le 15 avril 2019.

Annexes

Annexe A.1. Dialogue et sensation tactile

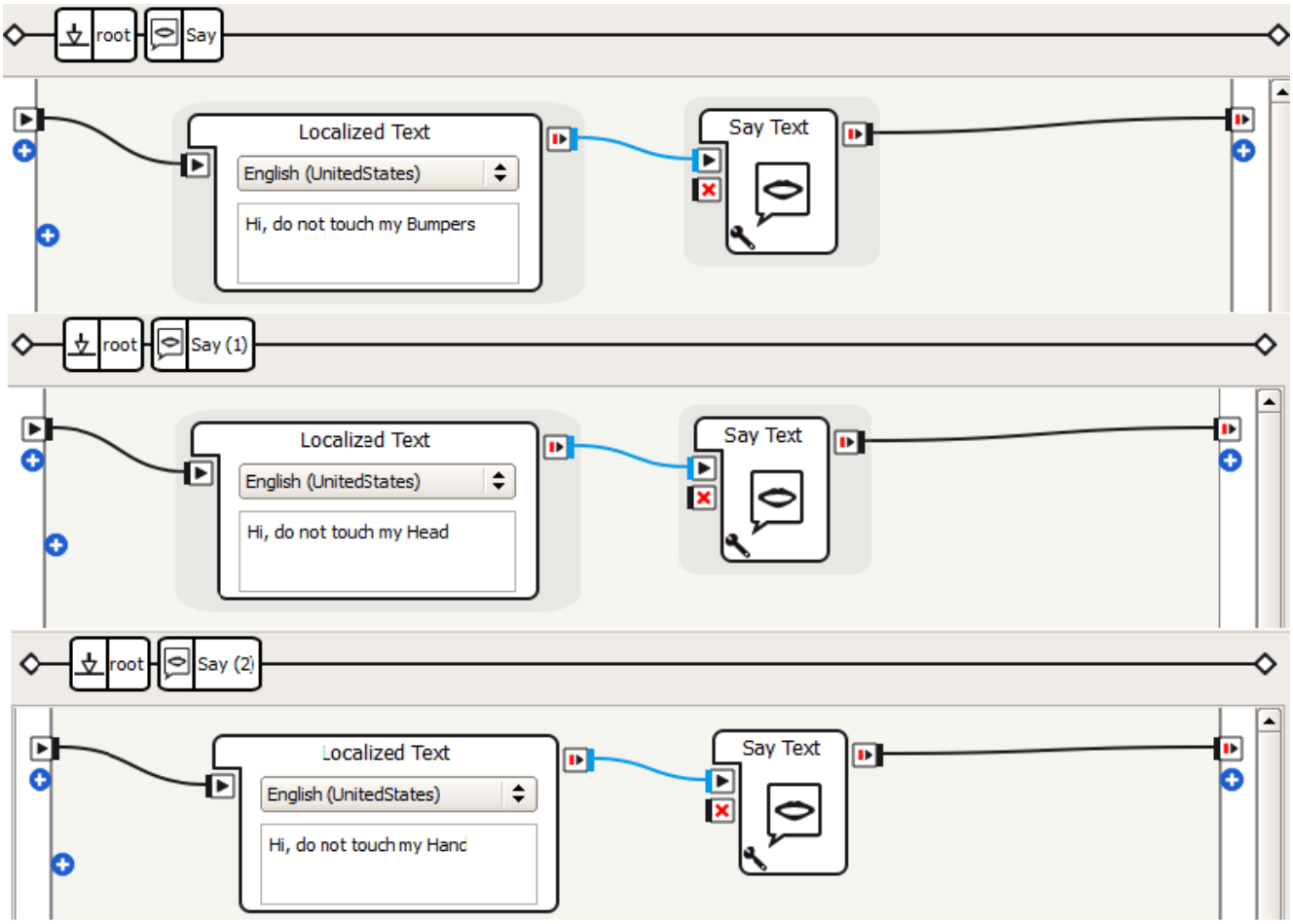


Figure A.1. Illustration du contenu des boites « Say ».

Annexe A.2. Script d'un dialogue animé écrit en langue française.

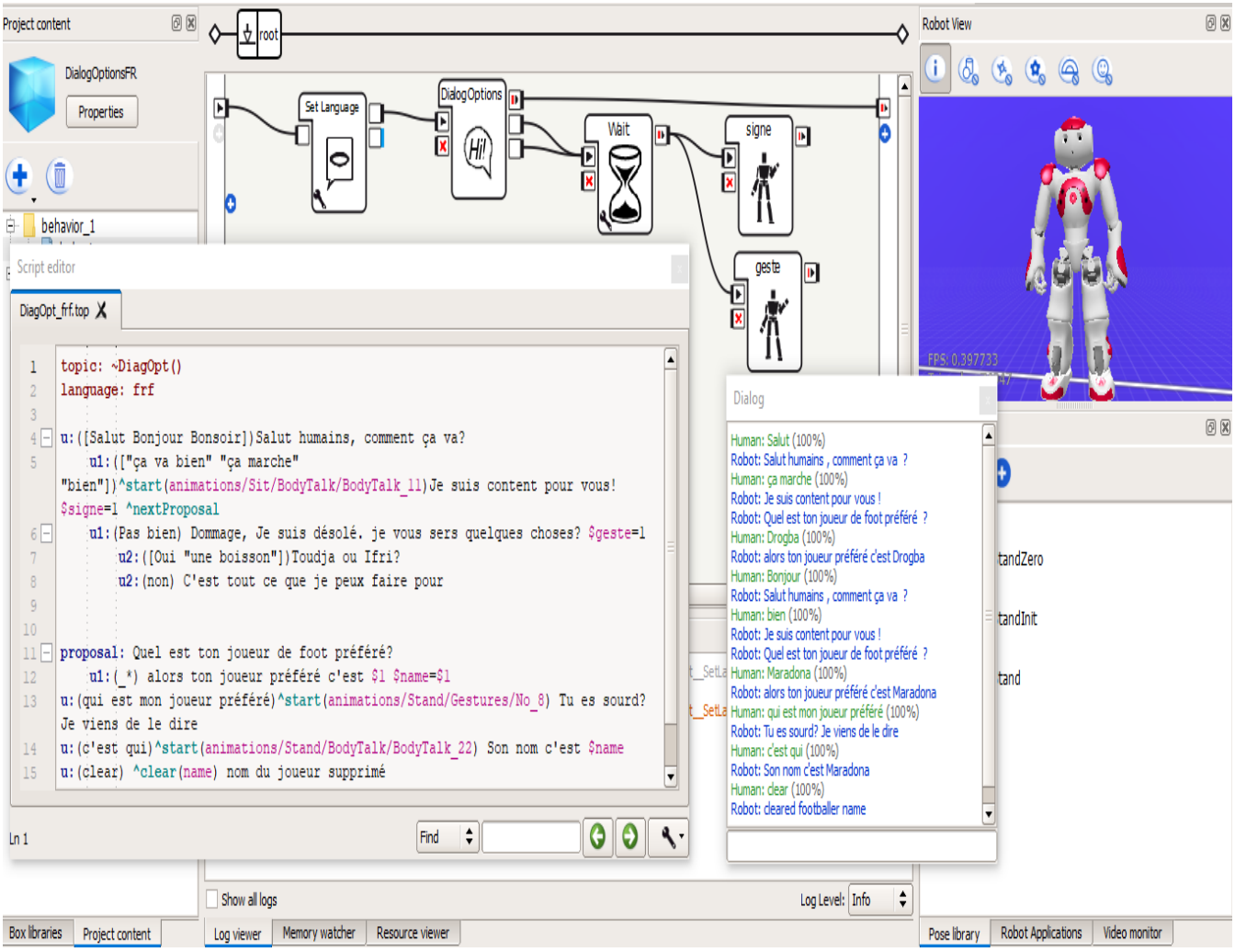


Figure A.2. Logigramme et le script de dialogue.

Annexe A.3. Sous-diagrammes du programme de distribution d'objets

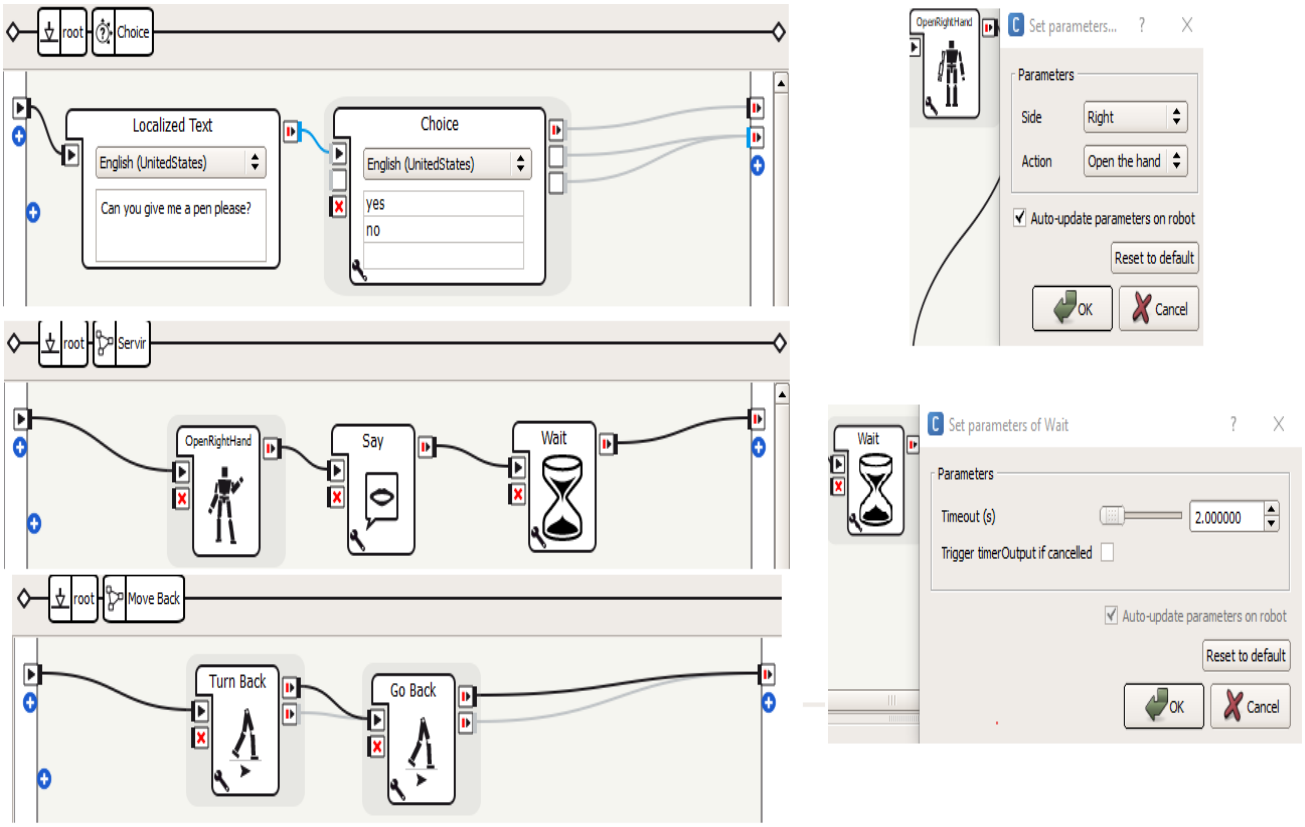


Figure A.3. Description du contenu de quelques boites du logigramme de distribution d'objets.

Annexe A.4. Script de la commande vocale

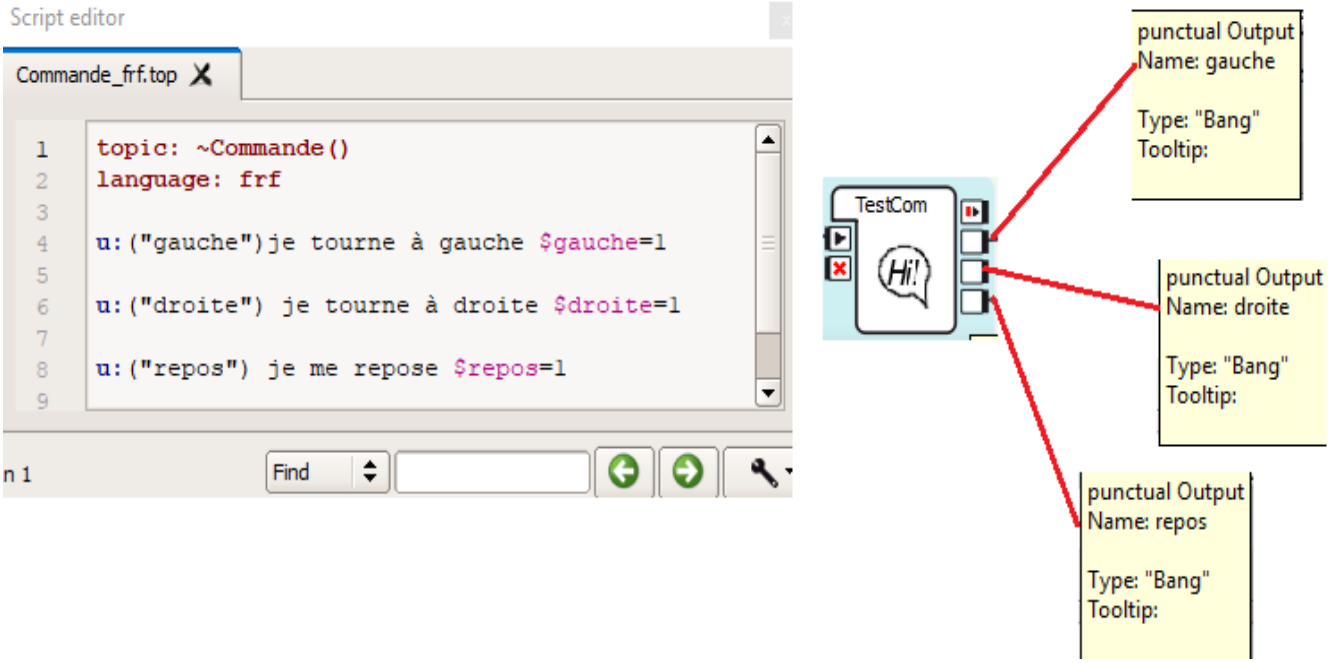


Figure A.4. Illustration du script de dialogue (commandes vocales) produisant des sorties de type Bang pour activer les animations associées.

Annexe A.5. Sous-diagramme de la boite « Choice » et la programmation sous Python

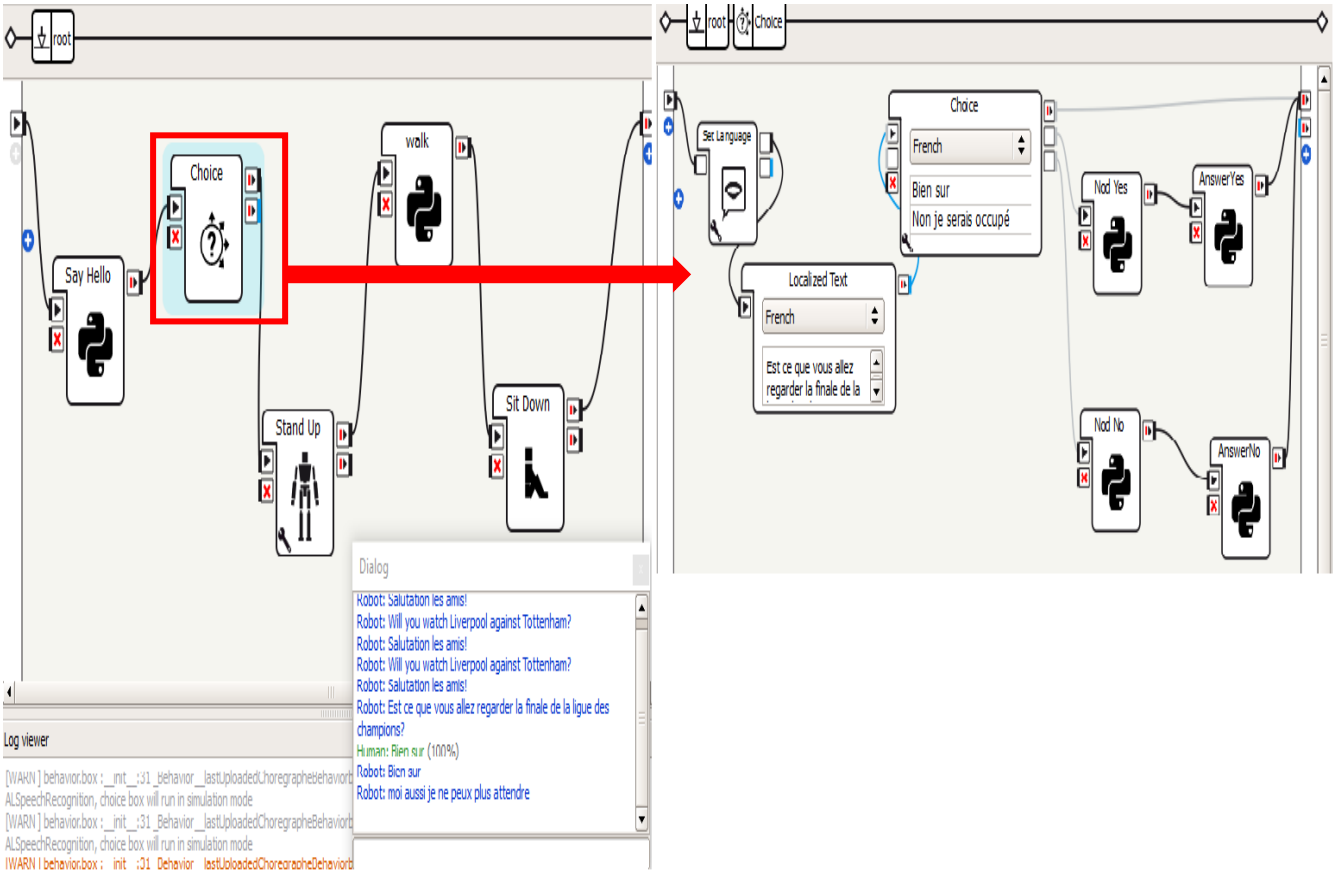


Figure A.5. Illustration du contenu de la boite Choice combinant des dialogues et des actions programmées sous Python de l'annexe A.6.

Annexe A.6. Sous-programme d'une animation en langage Python

L'algorithme A.6. décrit un exemple d'imitation de l'expression gestuelle de l'être-humain. Il décrit le passage de la commande du mouvement tangage (rotation de l'actionneur associé autour de l'axe Y) de la tête à l'expression de l'acceptation habituelle (balancement haut /bas de la tête).

Algorithme A.6. Exemple de l'expression gestuelle.

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

    def onLoad(self):
        #put initialization code here
        pass

    def onUnload(self):
        #put clean-up code here
        pass

    def onInput_onStart(self):
        motionProxy = ALProxy("ALMotion")
        names = ['HeadYaw','HeadPitch']
        times = [[1.0], [1.0]]
        motionProxy.angleInterpolation(names,[0.0, 0.0], times, True)

        for i in range(1):
            motionProxy.angleInterpolation(names,[0.0, 1.0], times, True)
            motionProxy.angleInterpolation(names,[0.0, -1.0], times, True)

            motionProxy.angleInterpolation(names,[0.0, 0.0], times, True)

            self.onStopped()

    def onInput_onStop(self):
        self.onUnload() #it is recommended to reuse the clean-up as the box is stopped
        self.onStopped() #activate the output of the box
```

Résumé :

L'objectif de ce travail de Master concerne la familiarisation et le développement de quelques applications permettant d'explorer les compétences du robot humanoïde NAO. Le travail a été abordé par une exploration générale de la robotique et en particulier le domaine des humanoïdes. Puis, on a étudié les aspects matériels et logiciels du robot NAO. La dernière partie concerne le fond du travail, à savoir, la configuration et la communication avec le robot, l'exploration du logiciel Choregraphe et l'élaboration de quelques applications démonstratives, en utilisant les logigrammes et le langage Python.

Mots-clés : NAO, Choregraphe, Robotique, Humanoïde, Programmation, Python.

Abstract :

The aim of Master project is to understand the NAO humanoid robot and to develop some applications to discover its skills. First, the field of robotics, in particular, humanoids has been explored. Then, the hardware and software aspects of NAO robot have been studied. Finally, we investigated the configuration and the communication protocols of the robot, discovered the software Choregraphe, and developed some demonstrative applications.

Keywords : NAO, Choregraphe, Robotic, Humanoid, Programmation, Python.

ملخص:

الهدف من عمل الماجستير هو التعرف على بعض التطبيقات وتطويرها لاستكشاف مهارات روبوت **NAO**. تمت معالجة العمل من خلال استكشاف عام للروبوتات. بعد ذلك، درسنا جوانب الأجهزة والبرامج الخاصة بروبوت **NAO**. الجزء الأخير يتعلق بخلفية العمل، أي التكوين والتواصل مع الروبوت، واستكشاف برنامج **«Choregraphe»** وتطوير بعض التطبيقات الإيضاحية، وذلك باستخدام **«logigrammes»** ولغة البرمجة **«Python»**.

Agzul :

Iswi n umahil-a n Master, yerza asifses akked tneflit n kra n yisnas yessurufen anadi ɣef tzemmar n urubu azunles NAO. Amahil-a nebda-t s unadi deg tussna n yirubuten ladɣa taɣult n yizunlas. S yin, nezrew timezra tingawiyin akked tseɣzanin n urubu NAO. Ahric aneggaru, d win I d ul numahil, yerza tawila, ameslay akked urubu, anadi s telqey deg useɣzen **"Choregraphe"** akked usbeddi n kra n yisnas imeskanen s useqdec n yiseɣzan iseldanen d umeslay n **"Python"**.