Projet TW53

PhantomX controlled by Raspberry

Hakan KAYA- Nicolas MORENTON – Julien PEREIRA

Table des matières

Remerciements

Nous tenons à remercier M. Zhi Yan, porteur du projet, pour son aide et ses conseils lors et entre les revues, et qui a été à notre écoute et a pu nous fournir le matériel nécessaire pour faire avancer notre projet.

Nous tenons également à remercier M. Hassan Ouhamad et M. Sihao Deng pour leurs aides tout au long du projet.

Introduction

Le projet de l'UV TW53 a pour objectif de mettre en place un système contrôlant un robot Phantom X à l'aide d'une carte Raspberry et d'un ordinateur pilote (remote PC).

Notre étude comporte trois parties principales qui sont également les objectifs du projet :

- Pinger le robot avec le remote PC à l'aide d'un routeur. Il s'agit ici d'identifier le Raspberry à partir de son adresse IP local.

- Faire bouger le PhantomX à l'aide de l'interface de commande du Raspberry. C'est la partie principale du projet qui nécessitent l'ensemble des éléments à notre disposition (Robot, Raspberry, PC, routeur)

- Trouver une alternative au système d'alimentation actuel, c'est-à-dire trouver une batterie embarquable pour remplacer l'alimentation actuelle qui se brache sur le secteur.

Enfin, nous terminons en listant les différentes compétences et connaissance acquise tout au long de ce projet.

I- Première étude

Du fait que nous ne connaissions pas les cartes Raspberry, nous avons donc dans un premier temps effectué des recherches afin de connaitre cet outil. Un Raspberry possède son propre système d'exploitation Linux et peut être considéré comme un mini PC. On peut y brancher une souris, un clavier, un écran et d'autres périphériques comme sur un ordinateur classique, on peut aussi l'utiliser pour aller sur internet après avoir configuré le Wi-Fi.

Afin de faire fonctionner le robot PhantomX, il faut utiliser le logiciel ROS Kinetic qui ne marche que sous système d'exploitation linux. Etant donné que nous n'avions pas de PC sous linux disponible, nous avons tenté d'installer les « paquets » ROS Kinetic directement sur le Raspberry. Dans un premier temps, les paquets s'installaient normalement mais ensuite nous avons eu plusieurs messages d'erreurs. En effet, le Raspberry n'est pas assez puissant pour faire tourner le logiciel.

Figure 1 : Carte Raspberry

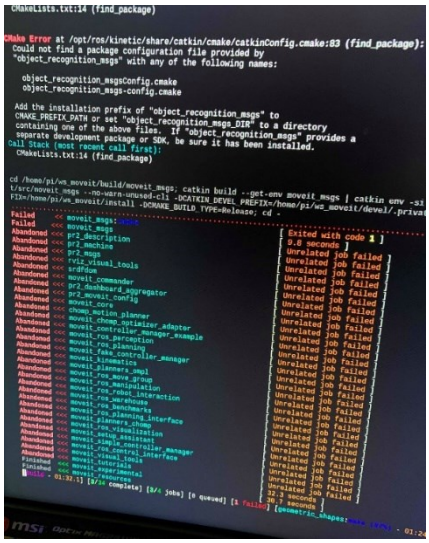

Figure 2 : Installation des paquets ROS Kinetic sur le Raspberry

A la suite de la revue 1, nous avons eu accès à une tour sous Linux à partir du 20 novembre.

II- Ping du robot

Dans un premier temps, nous avons essayé de connecter le robot et le remote PC à un réseau sans fil, d'abord l'internet de l'UTBM, puis via partage de connexion à l'aide d'un téléphone portable. Cependant, nous nous sommes rendu compte que le remote PC ne possédait pas de carte Wi-Fi et ne pouvait donc pas se connecter à un réseau sans fil.

À la suite de cette impossibilité, nous avons eu accès à un routeur en plus du PC. Celui-ci nous permet de faire l'interface entre le robot et le PC, la liaison étant réalisée à l'aide de câbles Ethernet (RJ 45). Il a fallu dans un premier temps identifier les adresses IP du Raspberry et du PC.

L'une des principales caractéristiques du Raspberry que nous ignorions est qu'il possède deux adresse IP distincte, l'une pour les connexions filaires et l'autres pour les connexions sans fil. Celle utilisée pour finir n'était donc pas la même que celle de la première tentative sans fil.

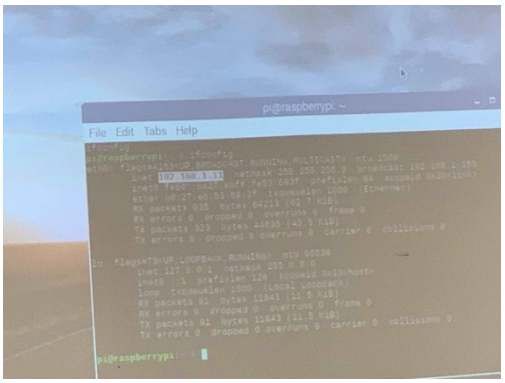

Figure 3 : Raspberry
Adresse IP : 192.168.1.11

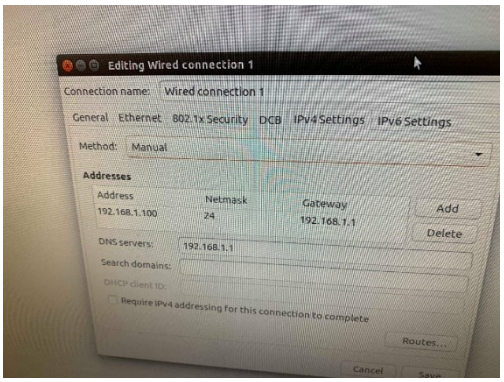

Figure 4 : Remote PC (Linux)
IP adress : 192.168.1.100
Netmask : 24
Gateway : 192.168.1.1

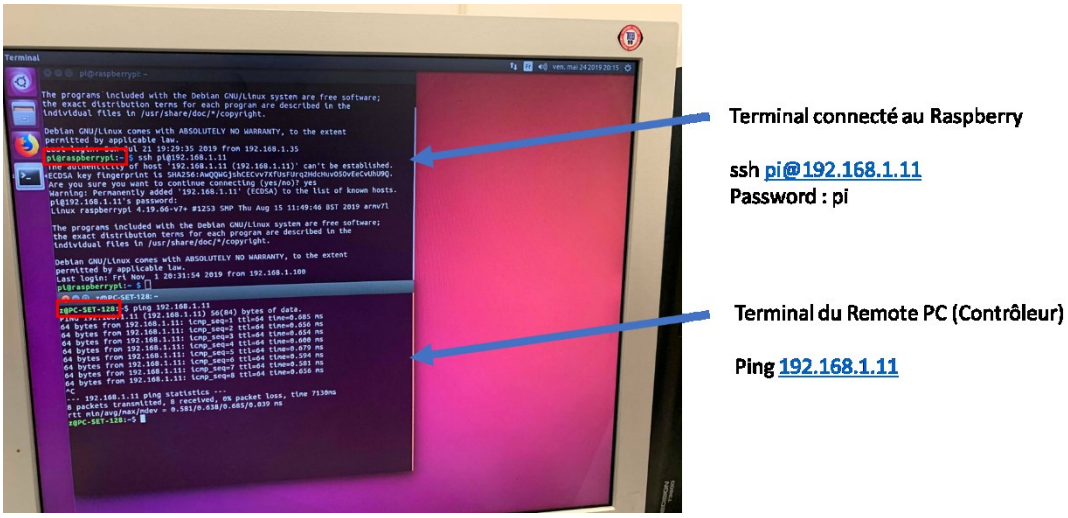

Figure 5 : Connexion du Raspberry au Remote PC à l'aide du routeur

III- Alimentation alternative

L'alimentation du Turtlebot sur lequel est fixé le PhantomX peut être alimenté par une petite batterie et permet au turtlebot de se déplacer sans être branché. En revanche le raspberry et le phantomX on besoin d'une alimentation constante pour pouvoir fonctionner.

L'alimentation actuelle à les caractéristiques suivante :

- Prise secteur
- 12V
- 5 A
- Connectique Jack 2,1 mm

Nous avons donc cherché des batteries ayant la même connectique mais aussi les mêmes tension et intensité en sortie. Nous nous sommes rendu compte que ces caractéristiques sont assez communes et on peut trouver des batteries adaptées pour quelques dizaines d'euros sur plusieurs sites spécialisés ou grand public :

https://euro-makers.com/fr/alimentation/2457-httpeuro-makerscomalimentation-dc-projets2457-batterie-lithium-ion-compacte-12v-6800-mah-rechargeablehtml-3701172913835.html

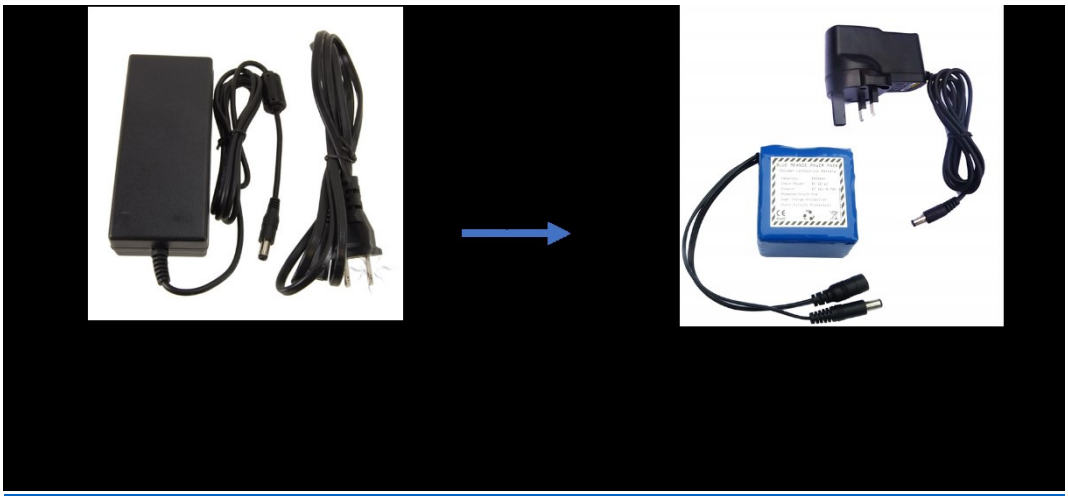

Figure 6 : Exemple de batterie utilisable sur un robot PhantomX

IV- Faire bouger le robot

Pour pouvoir faire bouger notre robot, nous devions apprendre à manipuler Linux Ubuntu et plus précisément le terminal ssh. Ce teminal permet de contrôler l'ordinateur (création de dossiers…), de télécharger des logiciels et de les utiliser. Nous nous sommes aidés de plusieurs des nombreux tutoriels disponibles en ligne.

Les commandes utilisées dans le terminal sont souvent les mêmes :

- Effectuer l'installation et la désinstallation de paquets en provenance d'un dépôt :

```
sudo apt-get install ros-kinetic-
```

- Cloner un dossier depuis GitHub :

```
git clone https://github.com/ROBOTIS-GIT/turtlebot3_msgs.git
```

- Ouvrir un répertoire :

```
cd ~/catkin_ws/src/
```

- Exécuter le script dans l'environnement courant :

```
source ~/.bashrc
```

Pour faire bouger le robot, et donc réaliser la tâche principale de notre projet nous avons dû utiliser bien sur le remote PC, le raspberry mais aussi des périphériques adaptés, notamment concernant les écrans.

En effet, les écrans disponibles dans les salles de l'UTBM possèdent uniquement des entrées DVI, or le remote PC a uniquement une sortie VGA et le Raspberry uniquement une sortie HDMI.

Nous avons eu accès à un écran adapté au VGA peu de temps après avoir récupérer le remote PC mais aussi pour pouvoir afficher le contenu du Raspberry nous avons dû utiliser le projecteur de la salle B421.

Un autre problème rencontré et que le remote PC, équipé d'un Linux Ubuntu, ne peut pas se connecter au réseau de l'UTBM en WI-Fi (il n'a pas de carte réseau), mais ne peut également pas se connecter via un câble Ethernet comme les autres ordinateurs de l'UTBM utilisant Windows.

Nous avons pu avoir accès à une carte réseau externe à partir du 7 janvier, ainsi nous avons pu connecter le remote PC à internet. Nous avons suivi un tutoriel fait pour des utilisateurs débutants de robots et de ROS Kinetic, ce tutoriel consistait surtout à installer des paquets via la console du remote PC. Les différentes étapes sont les suivantes :

- Installation de ROS Kinetic, plateforme open source de développement logicielle pour robot.
- Installation d'un Catkin Workspace, logiciel basé sur ROS
- Compilation avec Cmake, un compilateur open source

- Installation de Move_it, un autre logiciel basé sur ROS. Lors de l'installation, nous avons eu un message d'erreur : en effet le lien du tutoriel utilisé n'était plus à jour. Cependant nous avons réussi à télécharger le paquet nécessaire en allant le télécharger directement sur le site officiel de Move_it et ainsi continuer le tutoriel.
- Lancement de Rviz, un logiciel doté d'une interface graphique permettant entre autres de simuler notre robot.

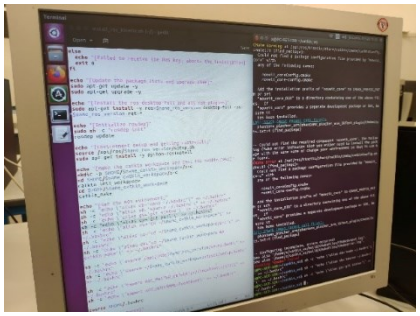
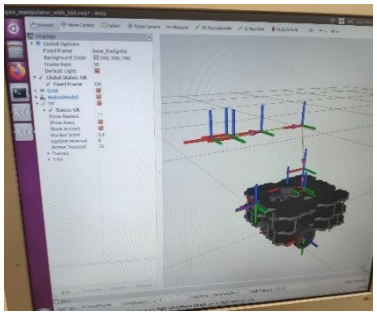
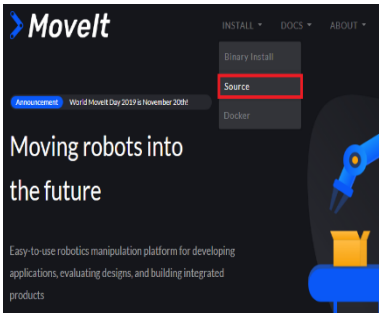

Figure 7, 8, 9 : Installation de ROS, Rviz, et MoveIt

Après l'installation de tous ces paquets, le remote PC peut être considéré comme prêt.

Changement de stratégie :

Le remote PC fut opérationnel environ une semaine avant la revue finale, nous avons estimé que remettre en place le système de connexion avec le routeur entre le raspberry prendrait trop de temps avec un risque important d'erreur lors de la configuration et du ping du raspberry.

Nous avons donc, comme évoqué lors de la deuxième revue, décider de brancher directement le robot au remote PC via câble USB à l'aide de la fonction « lsusb ».

Il a fallu dans un premier temps identifier sur quel port USB du PC nous avions branché :

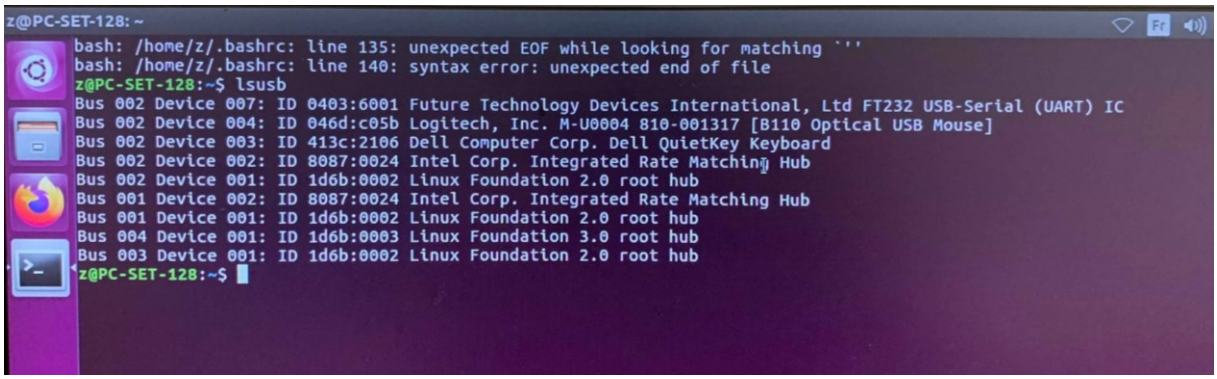

Figure 10 : Identification des ports USB du remote PC

Ici on peut voir que le robot est branché sur le port 007

```
lsusb
```

```
# ls -l /dev/bus/usb/*/*
```

Nous avons après utilisé la fonction « Sudo chmod 777 /dev/ttyUSB0 » pour « ouvrir » le port USB désigné et préciser à la machine que nous voulons orienter les prochaines fonctions vers le périphérique correspondant.

Ensuite la fonction « Arbotix terminal » permet d'identifier tous les servomoteurs connectés à ce port. La fonction « ls » permet de lancer la recherche, qui déclenche la « chorégraphie » d'allumage du PhantomX.

Malheureusement la recherche n'a pas abouti, le remote PC n'arrivant pas à identifier les différents servomoteurs, malgré le fait que le robot effectuait les mouvements. D'après le tutoriel suivi et d'autres guides vus sur internet, des LED sont censées s'allumer et clignoter lors de la recherche, or les servomoteurs de notre robot n'ont pas de LED ou ne s'allumaient pas.

D'après le tutoriel, l'identification des servomoteurs était la dernière étape avant de pouvoir contrôler le robot à l'aide de l'interface Rviz.

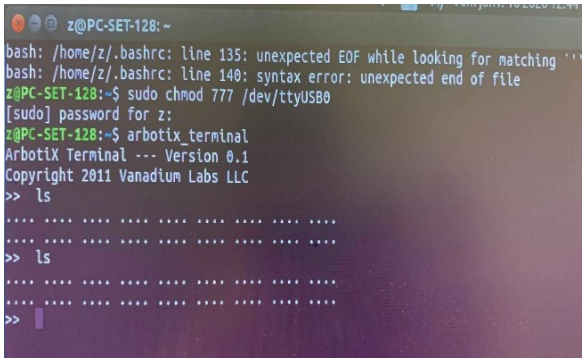

Figure 11 : Tentative d'idendification des servomoteurs

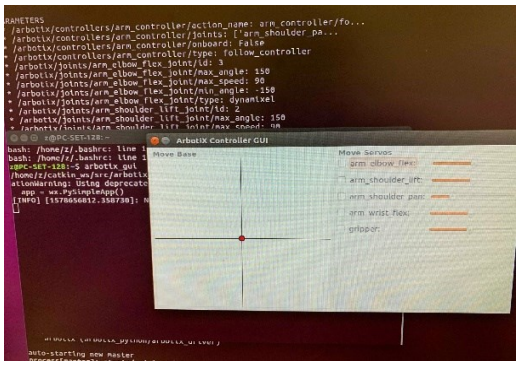

Figure 12 : Interface du logiciel Arbotix

Apports du projet

Ce projet nous a permis de découvrir et de d'utiliser des outils inédits, notamment un Raspberry, un système d'exploitation Linux Ubuntu et le logiciel ROS Kinetic ainsi que des logiciels associés.

- Raspberry

Les Raspberry sont des nano-ordinateurs au prix très réduits (25-45€) mais équipés de nombreuses connectiques fabriqués par la société à but non lucratif éponyme. Ils sont beaucoup utilisés à but pédagogique ou pour initier à la programmation et la robotique, mais ont également de nombreux autres usages :

  - ROS
  - Serveur WEB / NAS
  - Domotique
  - Console de jeu rétrogaming (Recalbox)
  - Mécatronique (Robot, drone,…)
  - Télévision / musique (media center)
  - Interface homme machine dans l'industrie

- Linux Ubuntu

C'est un système d'exploitation, tel que windows 10, libre, gratuit, très flexible avec une grande communauté de créateurs et des logiciels variés. L'une des caractéristiques principales est le terminal ssh : c'est une console qui permet entre autres de télécharger et d'utiliser des logiciels sans passer par un installeur.

- Robot Operating System (ROS)

Nous avons découvert ROS, un ensemble d'outils informatiques, tel que Roscore, Roslauch ou catkin, open-source qui supporte plus de 75 robots différents. Une communauté active propose de nombreux guide et tutoriels sur des sites tels que GitHub ou wikiROS. Il y a plusieurs distributions qui évoluent comme Kinetic, Melodic…

Conclusion

Nous avons réussi la première étape qui consistait à connecter le Raspberry avec l'ordinateur à l'aide d'un routeur (ping).
Nous avons trouvé une batterie qui semble être adapté à notre projet.
Pour la dernière partie, nous n'avons pas réussi à mettre en place le système *remote PC → Raspberry → Robot* de façon fonctionnelle mais nous avons réussi à contourner ce problème en branchant directement le remote PC au robot

Bien que nous n'ayons pas réussi l'objectif final, faire bouger le robot, nous gardons une image positive de ce projet pour plusieurs raisons : tout d'abord la longue durée du projet a permis un approfondissement de nos recherches et de nos essais.
Ce projet est centré sur l'informatique et la robotique, qui ne sont pas nos spécialités. Nous avons rencontré plusieurs difficultés liées à cause de notre manque d'expérience, notamment au début du projet (ping du robot, connexion internet…). Cependant nous avons réussi à relever les différents défis que nous a offert cette étude.
Les apports en connaissances et compétences techniques nous seront surement utiles pour nos stages ou emplois d'ingénieurs, en effet, Linux et Raspberry sont des incontournables du monde technico-industriel.