

Enseignants Référents : VERNIER Flavien & LEPLUS François

Étudiants : CHAPELET Ludovic & MIGNOTTE Arnaud

Année : 2017/2018

Date de remise du rapport (15/02/2018)

RAPPORT PRD

Réhabilitation du robot Pekee



Figure 1 : Le robot Pekee avec son alimentation et son Joystick de pilotage

Polytech Annecy-Chambéry
5 Chemin de Bellevue, 79940 Annecy-le-Vieux
04 50 09 66 00

Remerciements

Dans un premier temps, nous tenions à remercier nos référents de projet, M VERNIER Flavien et M LEPLUS François, pour leur confiance sur ce projet, leur accompagnement, leur aide, en informatique pour M VERNIER et en électronique pour M LEPLUS, et le temps qu'ils nous ont consacrés pour nous permettre d'avancer.

Dans un second temps, nous souhaitions remercier vivement les personnes suivantes :

- M CHAMBON Laurent pour son expertise en électronique (fabrication des cartes électroniques, soudures délicates, matériel donné et/ou prêté...), ses nombreux conseils dans cette partie et sa disponibilité.
- M BLAIZE Gérard pour son expertise en mécanique (impression 3D, restructuration du robot Pekee, assemblage des différentes parties du robot...), ses précieux conseils et sa patience face à nos nombreuses demandes.
- M MANDALLAZ Patrick pour ses différents dépannages en informatique.
- M MONNET Sébastien pour son expertise en informatique et plus particulièrement dans la configuration de la Raspberry.

Enfin, nos remerciements vont également aux élèves de mécanique, M PORTIER Corentin & M RATEL Corentin, pour leur précieuse aide dans la conception des supports 3D de Pekee ainsi qu'à M DECAUT Clément – élève en IAI – pour son expertise en C et son aide lors des soudures.

Nous terminerons cette partie en exprimant notre gratitude envers toutes les personnes qui ont pu nous aider de près ou de loin à notre projet et qui nous ont accordés de leur temps pour répondre à nos problématiques, nous pensons tout particulièrement à M MOUILLE, M BADEL, M CURTELIN et Mme PASSARD. Il se peut que nous ayons oublié de mentionner certaines personnes, nous tenons à nous en excuser et nous leur adressons nos sincères remerciements.

Sommaire

	N° Page
<i>Remerciements</i>	1
<i>Sommaire</i>	2&3
<i>I. Introduction</i>	4
<i>II. Description de Pekee & Reprise du projet</i>	5
1) <u>Description du robot Pekee I : Ancienne génération</u>	5
2) <u>Reprise du projet : État des lieux</u>	7
<i>III. Pekee 2.0 : Les objectifs et les ajouts</i>	9
1) <u>Objectifs du projet</u>	9
2) <u>Les ajouts matériels par rapport à la version I</u>	10
➤ <u>Raspberry PI 3</u>	10
➤ <u>Arduino Mega</u>	10
➤ <u>Pont en H</u>	11
➤ <u>Les batteries 5V et 12V</u>	12
➤ <u>Le bouton ON/OFF et une carte de relais</u>	13
➤ <u>La roue folle</u>	14
➤ <u>Le joystick</u>	14
➤ <u>Les capteurs infrarouges et MinIMU 9 axes</u>	14
3) <u>Gestion de projet</u>	15
<i>IV. Conception électronique & Étude des capteurs</i>	16
1) <u>Structure de Pekee</u>	16
➤ <u>La base de Pekee</u>	16
➤ <u>La tête de Pekee</u>	19
➤ <u>Le cerveau de Pekee</u>	21
❖ <u>La 1^{ère} couche : Batterie 5V</u>	21
❖ <u>La 2^{ème} couche : La gestion de l'alimentation</u>	22
❖ <u>La 3^{ème} couche : La partie intelligente</u>	24
➤ <u>Résumé des connexions</u>	25
❖ <u>Pins utilisés sur l'Arduino MEGA & Pins de la nappe</u>	25
❖ <u>Schéma électronique du système d'alimentation</u>	26
2) <u>Étude des capteurs</u>	27

Sommaire (suite)

N°	Page
<i>V. Conception logiciel Arduino & Raspberry</i>	29
1) <u>Arduino</u>	29
➤ <u>Programmes Arduino permettant l'étude des capteurs</u>	30
2) <u>Raspberry</u>	32
<i>VI. Pistes d'amélioration</i>	34
1) <u>Partie électronique</u>	34
2) <u>Partie logicielle</u>	35
<i>VII. Conclusion</i>	36
<i>Annexe : Webographie et sources du rapport</i>	37
➤ <u>Sources des images</u>	37
➤ <u>Sources du rapport</u>	38

I. Introduction

Dans le cadre de nos études d'ingénieur de 5^{ème} année, nous avons été amenés à continuer la réhabilitation d'un robot – nommé Pekee (que l'on peut voir en *Figure 1*) – pour notre projet de fin d'études. Ce projet nommé « **Projet de Recherche et Développement** » se fait soit en partenariat avec une entreprise soit directement avec notre école. Avec un total de 6 semaines de projet et en complète autonomie, l'objectif est de répondre à une problématique ou un cahier des charges donné. Mettant à l'épreuve nos connaissances et compétences apprises au cours de notre cursus cette expérience est une des plus enrichissantes et valorisantes en vue de notre entrée dans le monde du travail.

Les travaux de notre projet consistaient à reprendre le projet de l'année dernière qui n'avait pas abouti et de le faire évoluer. L'objectif initial du projet est de réhabiliter un robot des années 2000 qui n'a plus été utilisé depuis une quinzaine d'années afin de permettre aux futurs étudiants d'**APP Robotique de Service** d'avoir un nouveau robot. Nos objectifs que nous étions fixés au début de celui-ci étaient de reprendre le travail de l'an passé, revoir entièrement la conception électronique du robot, d'intégrer de nouveaux capteurs et enfin de prévoir une partie logicielle permettant le pilotage du robot.

Pour ce projet, encadré par M VERNIER Flavien et M LEPLUS François, nous avions carte blanche, nous permettant de prendre quelques initiatives notamment au niveau de l'arrangement matériel et de la coque de Pekee.

La structure de ce rapport se décompose en sept grandes parties, dont la première est l'introduction générale.

La deuxième partie présentera le robot Pekee ancienne génération et le projet de l'année dernière afin d'avoir une comparaison avec l'évolution de notre projet.

La troisième partie sera consacrée à nos objectifs, notre vision du robot version 2.0 avec la présentation du matériel et notre gestion de projet

Les quatrième et cinquième partie seront le corps du rapport où nous aborderons le travail que nous avons pu effectuer au cours de notre projet. Dans un premier temps, nous nous focaliserons sur la conception électronique et mécanique du robot, nos choix de conception des différentes cartes électroniques, l'étude des différents capteurs ainsi que la disposition des différents éléments matériels sur le robot. Dans un second temps, nous détaillerons la partie logicielle avec notamment la gestion de l'alimentation de la Raspberry et le pilotage du robot Pekee.

La sixième partie vous présentera les pistes d'évolutions pour continuer le projet l'année prochaine.

Enfin, vous trouverez en dernière partie, la conclusion générale et les enseignements que nous avons pu tirer de ce projet.

II. Description de Pekee & Reprise du projet

1) Description du robot Pekee I : Ancienne génération

Cette section décrit brièvement l'historique du robot Pekee ancienne génération ainsi que son architecture, tel qu'il l'était lors de sa sortie d'usine dans les années 2000.

Ce robot est issu d'un projet de deux étudiants en informatique et en électronique, Erwann LAVAREC et Laurent TREMEL. C'est le cabinet d'étude WANY ROBOTICS – dont le PDG est M LAVAREC – qui a sorti ce robot dans les années 2000.

Pekee est une sorte de plate-forme standard de la robotique qui se voulait grand public. En effet, ce robot pouvait être modifié et chacun pouvait apporter ses propres développements soit en concevant soi-même les cartes électroniques de commande, soit en les achetant (comme on peut le voir sur la figure ci-contre) : ces cartes s'enfichent dans le robot.



Figure 2 : Cartes électroniques enfichables

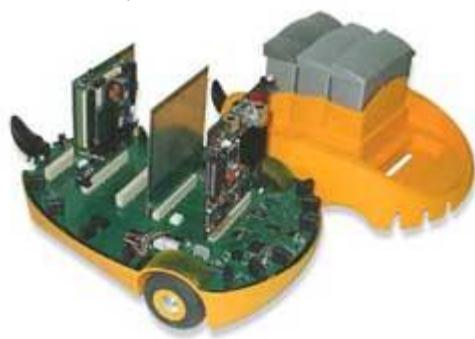


Figure 3 : Vue éclatée de l'ancien robot

À l'époque Pekee se voulait être un prototype réservé essentiellement au monde de la recherche pour tester les applications destinées à la robotique pour ensuite devenir une technologie indispensable chez les particuliers, que ce soit pour des applications comme robot aspirateur, apporteur de café, enregistreur de messages, arroseur de plantes... Une application qui a pu être développée par le CNRS consistait à lui faire apprendre des chemins de ronde. Cela était possible grâce à une phase d'apprentissage où l'utilisateur téléopérait l'étape de reconnaissance à distance grâce à une communication radio.



Figure 4 : Exemple d'application :
Apprentissage d'un chemin de ronde

Il s'agissait d'un réel système embarqué car il embarquait de nombreux capteurs : infrarouge, température, luminosité, odomètres, détecteur de choc, gyromètre... Cela lui permettait de surveiller les éléments critiques dans l'environnement où il opérait. Il embarquait un PC et une cartouche vidéo. Il communiquait de manière sans fil grâce au protocole WiFi 802.11. Un logiciel de simulation Maëva 2004 a été développé pour permettre l'écriture d'applications personnalisées et d'avoir une simulation 3D dans Microsoft Windows ainsi qu'un logiciel, nommé Environnement, permettant de contrôler à distance le robot.

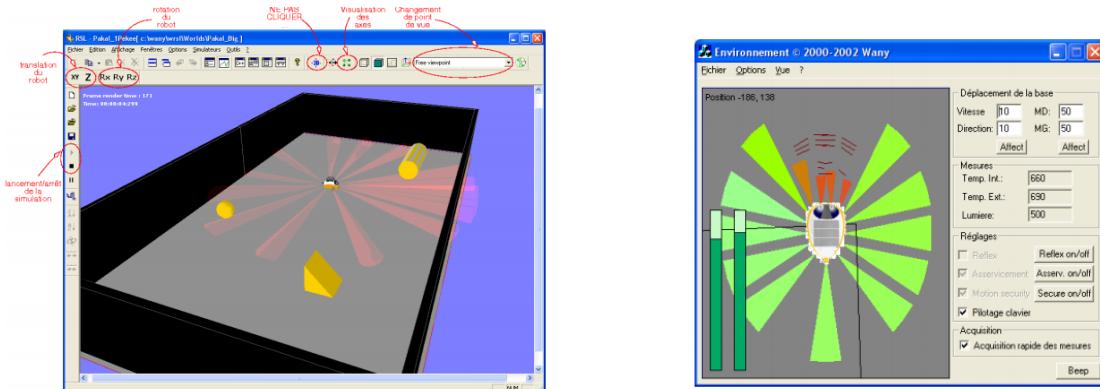


Figure 5 : Logiciels Maëva et Environnement permettant d'avoir une visualisation 3D de l'environnement et de suivre l'état des capteurs et de piloter à distance le robot.

Pekee intégrait deux cartes filles enfichables sur la carte mère (comme vue en *Figure 2*) pour permettre aux passionnés de programmer le robot comme il le désirait :

- Une carte fille pour PC Windows 98 comprenant un port Ethernet et des ports USB ainsi que différents connecteurs standards pour le clavier, la souris et l'affichage. Cette carte comprenait également une caméra vidéo couleur. Cette carte intègre un calculateur Intel i486.
- Une carte fille pour contrôler à distance le robot Pekee, grâce à une liaison sans fil basée sur le protocole WiFi 802.11, diffuser des données et des vidéos directement sur un ordinateur en temps réel.

Toutefois ses cartes filles se pluggent sur le bus breveté OPP (Open Parallel Platform) de la carte mère de Pekee. La carte mère peut accueillir jusqu'à 5 cartes permettant de développer ses propres cartes électroniques. Cette carte mère (carte verte disposée horizontalement sur la *Figure 6*) est dotée d'un microcontrôleur, qui intègre un micro-processeur, de la mémoire et des fonctionnalités d'entrée-sortie. Ce microcontrôleur prend en charge l'actionnement des moteurs ainsi que la gestion de plusieurs modalités sensorielles, dont les codeurs disposés sur les roues motrices et les capteurs proximétriques infra-rouges.

Voici quelques caractéristiques techniques du robot :

- Il est doté de 3 roues dont 2 roues motrices indépendantes et une roue folle.
- Il possède 2 moteurs à courant continu à entraînement différentiel avec codeurs incrémentaux et suspension intégrée.
- Une vitesse maximale de l'ordre de 1 mètre par seconde.
- De nombreuses modalités sensorielles :
 - o Capteurs dits *proprioceptifs* (renseignant le robot sur son état interne) : Des codeurs sont disposés sur les axes de ses roues motrices permettant de mesurer le nombre de tours de celles-ci. De même, l'orientation du robot peut être repérée au moyen de gyromètres.
 - o Capteurs dits *extéroceptifs* (renseignant le robot sur son environnement) : Les deux plus importants sont une ceinture de capteurs de proximité de type infrarouges (au nombre de 15), qui permettent de mesurer la distance du robot avec les obstacles situés dans son voisinage immédiat, ainsi qu'une caméra vidéo couleur, qui lui permet de disposer d'images du monde dans lequel il évolue. Des capteurs de choc, de lumière ou de température sont également disponibles.
- Enfin 2 batteries de 12V (NiMH) sont embarquées dans la base mobile de Pekee pouvant être rechargées par un chargeur intelligent intégré que l'on connectait aux batteries via la queue de Pekee.

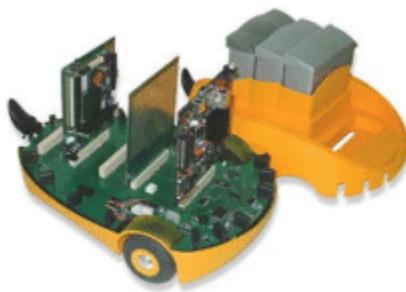


Figure 6 : Carte mère de Pekee avec ses cartes filles et ses différents capteurs

Son architecture, les technologies qu'il embarquait et les différents services qu'il proposait étaient à l'époque des prouesses technologiques.

Malheureusement pour la société et ce robot, il a vite été dépassé par l'arrivée des nouveaux robots comme Nao, Pepper, robot aspirateur... Néanmoins nous pouvons le considérer comme le précurseur de l'arrivée des robots dans notre quotidien qui tend encore à se développer dans les années à venir.

C'est pourquoi, l'école Polytech Annecy-Chambéry a décidé de donner une seconde vie à ce robot et de le réhabiliter. La réhabilitation de ce robot – resté plus de 15 ans dans un placard – a débuté durant l'année universitaire 2016/2017 et une première version a été proposée par les étudiants de cette promotion. Nous avons récupéré cette version que nous allons vous présenter dans la seconde partie afin d'avoir un état des lieux au commencement de notre projet.

2) Reprise du projet : État des lieux

Comme nous avons pu le dire précédemment le projet n'a pas commencé cette année. Cette partie va donc vous présenter l'état du projet tel que nous l'avons repris.

Les étudiants précédents ont principalement travaillé sur l'étude des différents composants de Pekee – afin de savoir ceux qui pouvaient être réutilisés – ainsi que sur l'asservissement de la commande des moteurs. Pour cela, ils ont fait une étude détaillée des moteurs et également des encodeurs. Ils ont pu conclure que la précision des encodeurs était peu précise pour permettre un asservissement correct. Ils ont donc programmé quelques fonctions pour piloter le robot telles que les fonctions avancer, tourner, réaliser un triangle... Ils se sont également aperçus que la roue folle entraînait une forte dérive au démarrage selon sa position initiale. Ils ont alors décidé de la supprimer pour réaliser leurs différents tests de pilotage.

Au niveau des composants originaux, seuls les moteurs ont été gardé car le reste était inutilisable car les capteurs et les cartes sont une technologie propriétaire à WANY Robotics. Ils ont alors acheté un pont en H pour pouvoir piloter les moteurs, un Arduino et un Raspberry pour la partie intelligente de Pekee. Ils ont également commandé pour la poursuite de projet des capteurs infrarouges pour permettre à Pekee de se repérer dans son environnement et un capteur 9 axes – gyromètre, accéléromètre, magnétomètre – pour avoir des mesures plus précises et ainsi réaliser un meilleur pilotage du robot. L'étude de ces capteurs n'a pas été faite par ces étudiants.

Enfin ils ont mis en place une communication I2C entre un Raspberry et un Arduino. Ils ont également réalisé un support 3D pour regrouper le Raspberry, l'Arduino et le pont en H à un seul et même endroit. Malheureusement ce support n'a pas été intégré à Pekee, il ne pouvait pas être mis sur le robot en raison de ses mauvaises dimensions. Par ailleurs, ils ont réalisé une carte électronique faisant office de shield pour la carte Arduino où l'on retrouve les connexions allant sur les moteurs, le pont en H et le Rasberry.

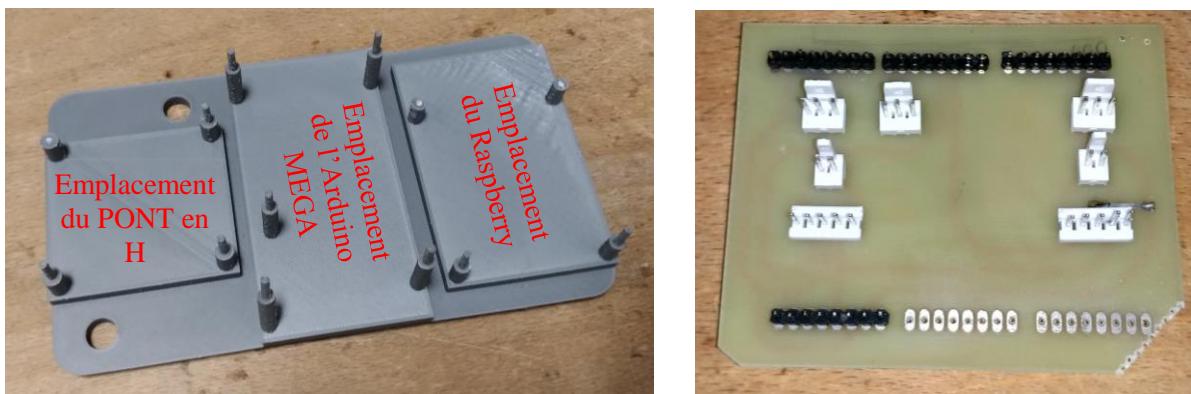


Figure 7 : Support 3D et carte électronique réalisés par les anciens étudiants

Vous avez donc ici un bref aperçu du projet tel que nous l'avons débuté. Par la suite, vous verrez que nous avons fait d'autres choix que nous justifierons, notamment au niveau de la communication Arduino/Raspberry, du support 3D et de la carte électronique.

III. Pekee 2.0 : Les objectifs et les ajouts

Cette partie s'intéresse aux objectifs généraux du projet ainsi qu'aux objectifs que nous nous sommes fixés au début du projet. Nous détaillerons également les ajouts matériels que nous avons pu faire sur Pekee. Et enfin nous consacrerons une courte partie sur notre gestion de projet.

1) Objectifs du projet

Comme nous avons pu le dire en introduction, la réhabilitation de ce robot permettra aux futures promotions IAI d'intégrer Pekee aux autres robots de l'APP Robotique de Service. On voit donc ici que l'objectif principal de ce projet est de pouvoir utiliser le robot à des fins éducatives. La vision que nous avons de ce projet, une fois finalisé, peut se résumer par le schéma synoptique suivant.

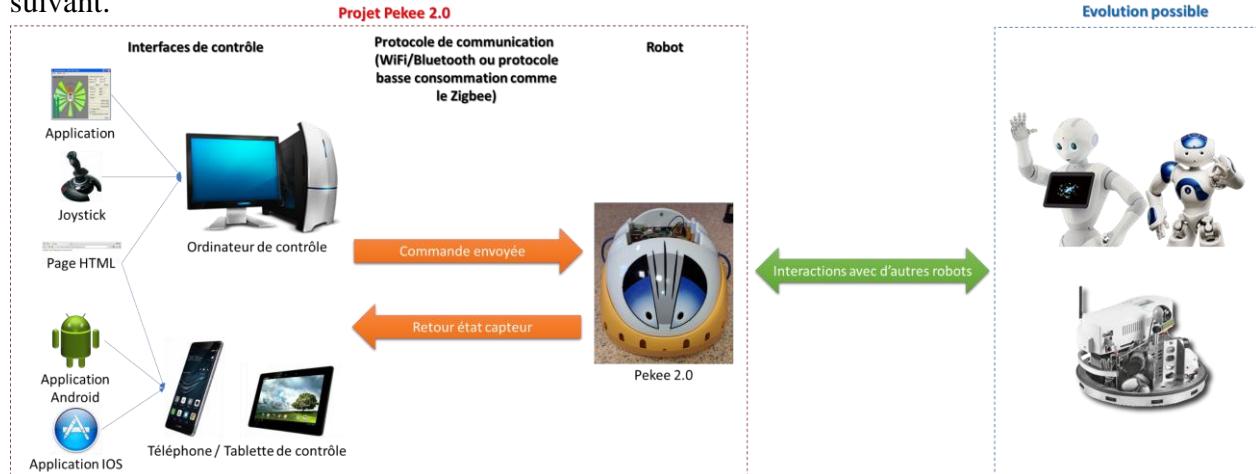


Figure 8 : Schéma synoptique de notre vision du projet

De notre point de vue, lorsque Pekee sera complètement réhabilité, nous imaginons qu'il pourra être piloté à distance soit par un ordinateur de contrôle au travers d'une application Windows qui aura pu être développée ou par un joystick ou encore depuis une page web. Il pourra également être commandé à distance depuis une application Android/IOS sur portable ou tablette. Cela nécessite donc un protocole de communication entre les API et le robot. Nous imaginons qu'intuitivement le WiFi pourrait être utilisé du fait que le robot intègre un Raspberry connecté. Mais nous pouvons très bien imaginer que ce soit par un autre protocole de communication comme le Bluetooth ou le ZigBee. Ce protocole sera utilisé pour échanger des informations de manière bidirectionnelle. Les informations transmises seraient la commande de pilotage et l'état des capteurs.

Enfin une évolution possible que nous voyons est l'interaction du robot Pekee avec d'autres robots comme Robotino, Pepper et Nao, envisageable grâce au Raspberry intégré sur Pekee.

Bien évidemment cette vision n'est pas notre objectif à la fin du projet car d'une part nous n'aurions pas le temps et d'autre part, d'un point de vue matériel, nous n'avons pas tous les éléments nécessaires pour le réaliser. C'est pourquoi, le projet de recherche et développement de l'année 2017/2018 s'est accentué sur les objectifs suivants :

- Étudier le travail réalisé par les anciens étudiants
- Faire l'étude des capteurs 9 axes et infrarouges & Intégration des capteurs sur le robot
- Prévoir des fonctions de pilotage simples et intuitives afin d'avoir un robot autonome facilement pilotable
- Proposer et concevoir une nouvelle architecture matérielle de Pekee

L'objectif final de notre projet est de pouvoir livrer un robot complètement autonome embarquant les différents capteurs, les batteries et l'intelligence du robot.

2) Les ajouts matériels par rapport à la version I

Pour nous permettre de réaliser les objectifs fixés précédemment, par rapport à l'ancienne version de Pekee que nous avons pu vous détailler lors de la seconde partie, le robot Pekee 2.0 intègre de nouveaux composants que nous vous présentons succinctement dans cette partie (certains composants étaient déjà présents lors de la reprise du projet).

➤ Raspberry PI 3

Afin d'avoir un ordinateur embarqué sur le robot, il a été décidé d'embarquer un Raspberry PI 3.



Figure 9 : Raspberry PI 3

Les principales caractéristiques de celui-ci peuvent être résumées sur la figure ci-contre :

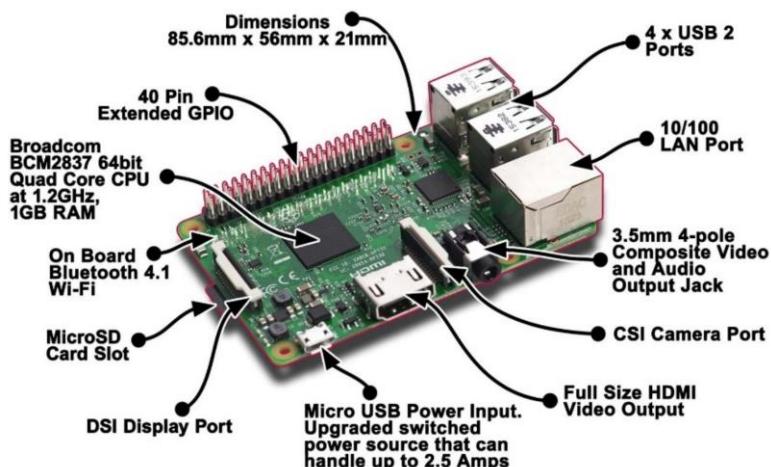


Figure 10 : Caractéristiques techniques du Raspberry Pi 3

Pour plus d'informations nous vous conseillons de vous rendre à ce lien :

<https://www.framboise314.fr/raspberry-pi-3-quoi-de-neuf-docteur/>

Cette carte sera la dernière couche du robot. Elle permettra d'avoir une connexion à distance et d'envoyer les commandes de pilotage au moteur via la carte Arduino. C'est un peu le cerveau de Pekee.

➤ Arduino Mega

Pour nous permettre une interaction plus facile avec les différents capteurs et les moteurs du robot, il a été choisi d'utiliser un Arduino Mega pour envoyer la commande reçue aux moteurs mais également pour récupérer l'état des capteurs.

Cela nous permet d'éviter les contraintes du GPIO de la Raspberry



Figure 11 : Arduino MEGA

Comme pour le Raspberry, ses principales caractéristiques peuvent être synthétisées sur l'image ci-contre :

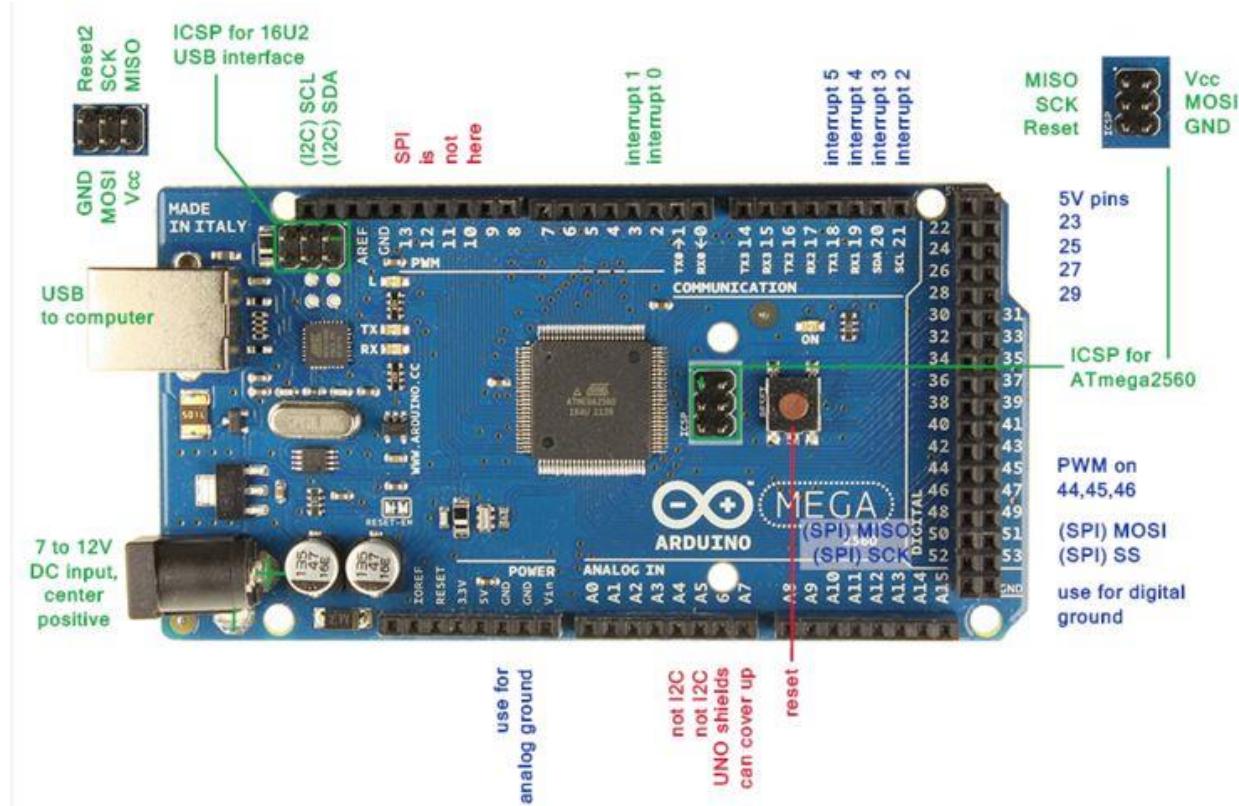


Figure 12 : Caractéristiques techniques de l'Arduino MEGA

Pour plus d'informations nous vous conseillons de vous rendre à ces liens :

<http://www.redohm.fr/2014/12/arduino/>

<https://www.arduino.cc/>

➤ Pont en H

Ce pont en H, dont la référence est L298 dual h_bridge, permet d'appliquer la tension nécessaire sur les moteurs pour permettre au robot de faire tourner ses moteurs en fonction de la commande envoyée.

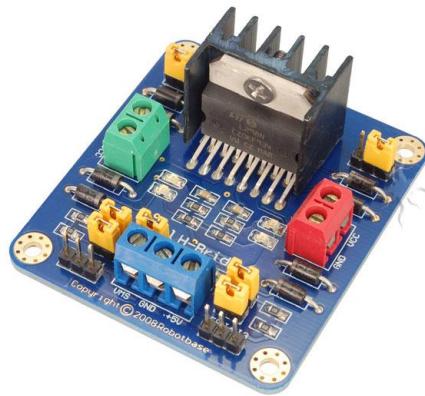


Figure 13 : Pont en H

Les caractéristiques techniques de ce pont seront détaillées un peu plus tard mais nous pouvons les résumer sur cette figure :

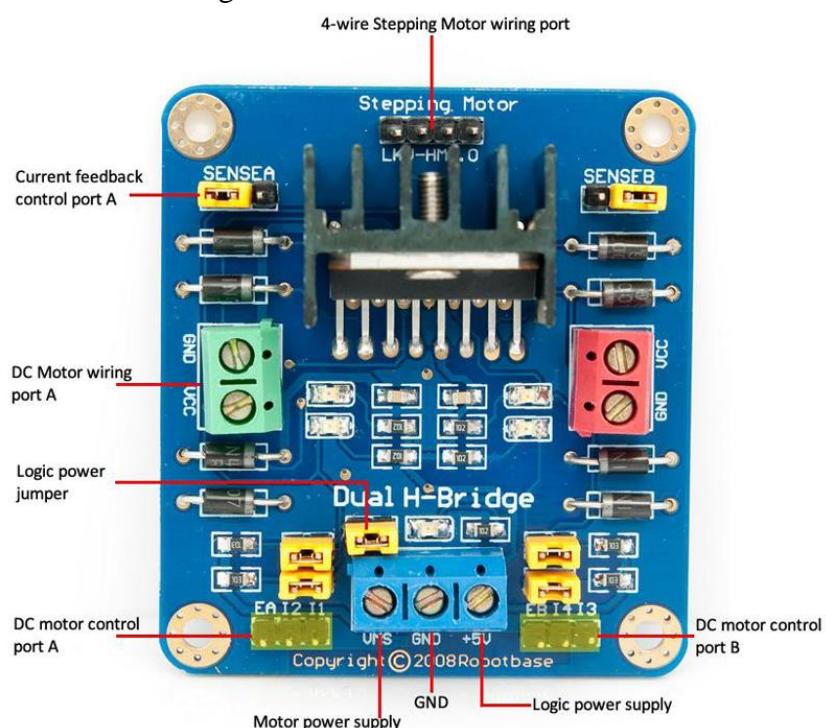


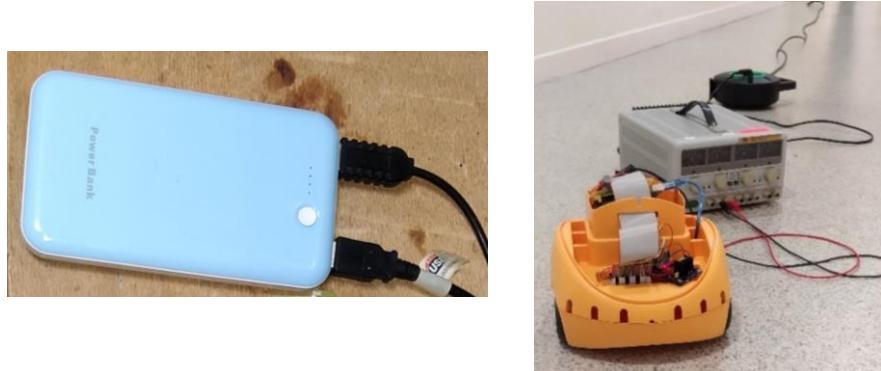
Figure 14 : Caractéristiques techniques du pont en H

Pour plus d'informations, nous vous renvoyons vers la documentation technique simplifiée de ce module : <https://www.robotshop.com/media/files/pdf/datasheet-mot103b1m.pdf>

➤ Les batteries 5V et 12V

Le robot n'ayant pas été utilisé depuis une quinzaine d'années, nous n'avons pas pris le risque de recharger les batteries fournies avec le robot. Celles-ci sont certainement inutilisables. Il nous a donc fallu prévoir une nouvelle alimentation. Pour cela, nous avons choisi d'avoir une batterie qui alimenterait uniquement les moteurs (tension entre 12V et 24V) et une batterie de 5V double dont un circuit alimenterait uniquement le Raspberry et l'autre sortie 5V alimentera le reste du circuit électronique.

Remarque : Suite à un timing trop serré, nous n'avons pas pu commander la batterie de 12V. Nous alimentons donc notre robot depuis une alimentation externe branchée sur secteur.



*Figure 15 : Batteries de Pekee
(à gauche : batterie externe de 5V, à droite : batterie 12-24V sur secteur)*

➤ Le bouton ON/OFF et une carte de relais

Afin de pouvoir contrôler l'alimentation générale du robot, nous avons décidé d'intégrer un bouton ON/OFF à notre système. Comme nous pourrons le détailler dans la partie Électronique, nous avons eu besoin d'une carte de relais pour nous permettre d'éteindre correctement le Raspberry. La carte de relais et le bouton ON/OFF utilisé sont les suivants :

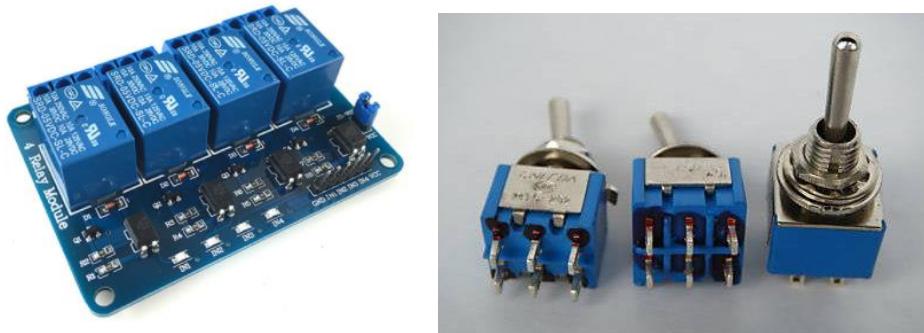


Figure 16 : Carte de relais et bouton d'alimentation du robot

Remarque : Le bouton On/Off présent sur le robot est de couleur rouge et non de couleur bleue

La référence de la carte relais est : *Tongling JQC 3FF S Z* et la documentation technique de celle-ci se trouve à cette adresse : <https://www.generationrobots.com/media/JQC-3FF-v1.pdf>

Un bon tutoriel a été fait sur ces relais (la référence change mais le principe reste le même) : <http://supertos.free.fr/supertos.php?page=1741>

Avec cette illustration, vous pourrez comprendre son fonctionnement :

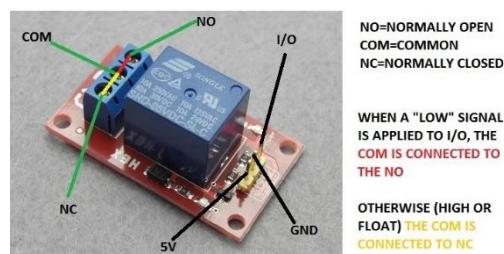


Figure 17 : Illustration du fonctionnement d'un relais 5V

➤ La roue folle

Suite à la conclusion et les remarques des étudiants du projet de l'année 2016/2017 sur la roue folle, nous avons choisis de la remplacer par une roue à bille afin d'avoir une meilleure trajectoire au démarrage. Cette roue a été personnellement confectionnée pour pouvoir s'adapter au robot.



Figure 18 : Changement de la roue folle

➤ Le joystick

Pour piloter le robot, n'ayant pas eu le temps de réaliser un asservissement à l'aide du capteur 9 axes, nous avons opté pour un asservissement manuel, c'est-à-dire le cerveau humain. Pour cela, nous offrons la possibilité à l'utilisateur de piloter le robot à l'aide d'un Joystick « *Extreme 3D Pro* ».



Figure 19 : Joystick de pilotage de Pekee

Ce joystick est relié au Raspberry depuis un port USB, mais à terme, il sera déporté sur un ordinateur de contrôle pour avoir un robot complètement autonome.

➤ Les capteurs infrarouges et MinIMU 9 axes

Les derniers ajouts sur le robot sont les capteurs infrarouges et le capteur 9 axes. Nous les détaillerons dans la prochaine partie. Nous n'avons pas pu intégrer les capteurs infrarouges sur le robot et cette problématique est laissée pour la poursuite du projet l'année prochaine. Le capteur 9 axes, quant à lui, a été intégré sur le robot, mais, faute de temps, il n'est pas exploité pour l'asservissement.



Figure 20 : Capteur MinIMU (à gauche) et capteur infrarouge (à droite)

Pour vous permettre de mieux comprendre ces capteurs, nous vous mettons un bon tutoriel sur le capteur MinIMU : <https://www.pololu.com/product/2468> et la documentation technique du capteur infrarouge GP2Y0A41SK0F : <https://www.pololu.com/file/0J713/GP2Y0A41SK0F.pdf>.

Pour résumer l'ajout des différents éléments sur le robot, voici une illustration de l'organisation des composants sur Pekee.

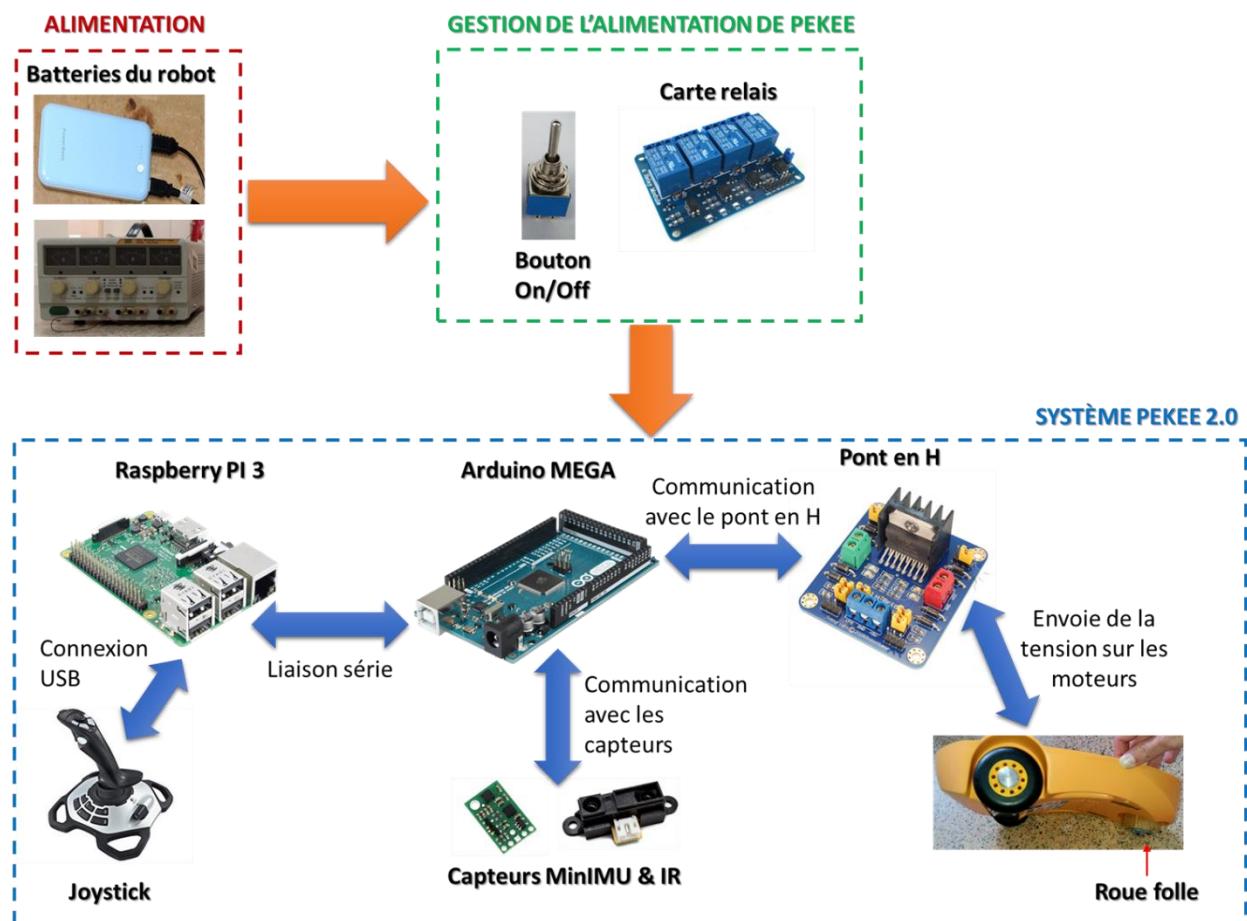


Figure 21 : Organisation des nouveaux composants sur Pekee 2.0

3) Gestion de projet

Avant de vous détailler notre projet, nous voulions consacrer une courte partie à notre gestion de projet. Celle-ci est importante lorsqu'un projet s'étale dans le temps. Pour notre cas, nous avons décidé de mettre en place un Trello dans lequel nous avons défini au fil du temps les différentes tâches à effectuer. Par ailleurs, cela nous permettait de garder une trace des tâches effectuées d'une séance à l'autre.

Nous avons également mis en place un Github pour ce projet. Celui-ci se situe à l'adresse suivante :

<https://github.com/chapellu/pekee2.0.git>

Vous retrouverez les différents programmes que nous avons pu faire, les documentations techniques de la plupart des composants cités précédemment, les librairies nécessaires pour les capteurs, le PFE de l'année 2016/2017 et d'autres ressources. Nous avons fait des mises à jour à la fin de chaque séance de projet, permettant une mise à jour régulière du git et garantissant la pérennité de notre projet.



Figure 22 : Outils de gestion de projet utilisés (Trello & Github)

IV. Conception électronique & Étude des capteurs

Lors de ce semestre, notre travail s'est essentiellement axé sur l'architecture matérielle du robot, l'étude des capteurs et la partie électronique du robot. Nous avons pu auparavant vous présenter les différents éléments composant le robot Pekee 2.0 toutefois, il nous a fallu concevoir différentes cartes électroniques pour nous permettre de faire communiquer tout le système et ainsi avoir un robot fonctionnel et autonome. Cette partie vous présente donc ce travail.

1) Structure de Pekee

Pekee a une architecture quelque peu contraignante dû à des contraintes de place et d'agencement. Nous avons donc exploité au maximum l'espace disponible au sein du robot tout en étant limités par sa conception originelle. Une fois la coque complètement vidée nous possédons un espace de travail de taille correct. Nous avons alors décidé de répartir les fonctionnalités dans l'espace, au plus proche de leurs lieux d'application. Afin de faciliter le câblage entre les différents emplacements nous avons utilisé une nappe.

Pekee est alors réparti en 3 grandes parties : la base, la tête et le cerveau.

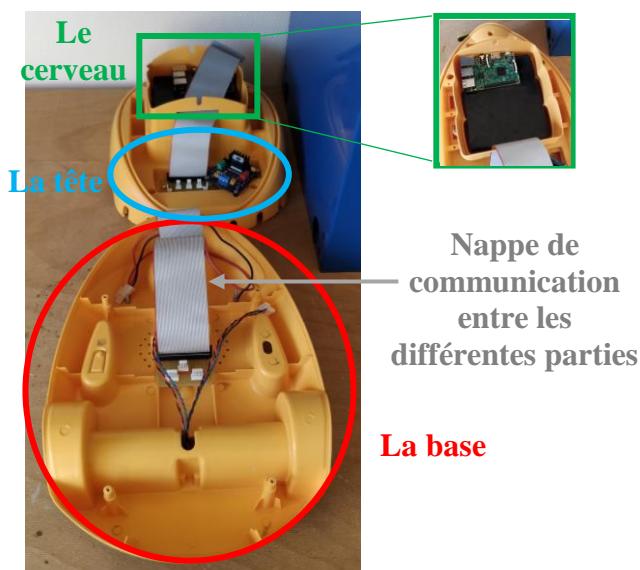


Figure 23 : Les 3 parties de Pekee

➤ La base de Pekee

La base de Pekee contient les moteurs. Nous avons donc choisi de concevoir une carte électronique qui permettra d'avoir accès au moteur. Dans cette base, nous trouverons également les différents capteurs de Pekee (Infrarouges, 9 axes...) et nous avons laissé un emplacement vide pour le stockage des batteries des moteurs. On trouve également à l'arrière de la base la queue de Pekee. Cette queue ayant un espace creux offrira un accès aux câbles permettant le recharge des différentes batteries.

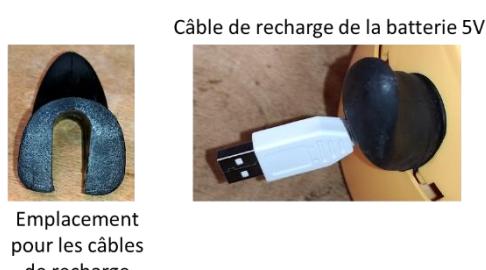


Figure 24 : Queue de Pekee pour la recharge des batteries

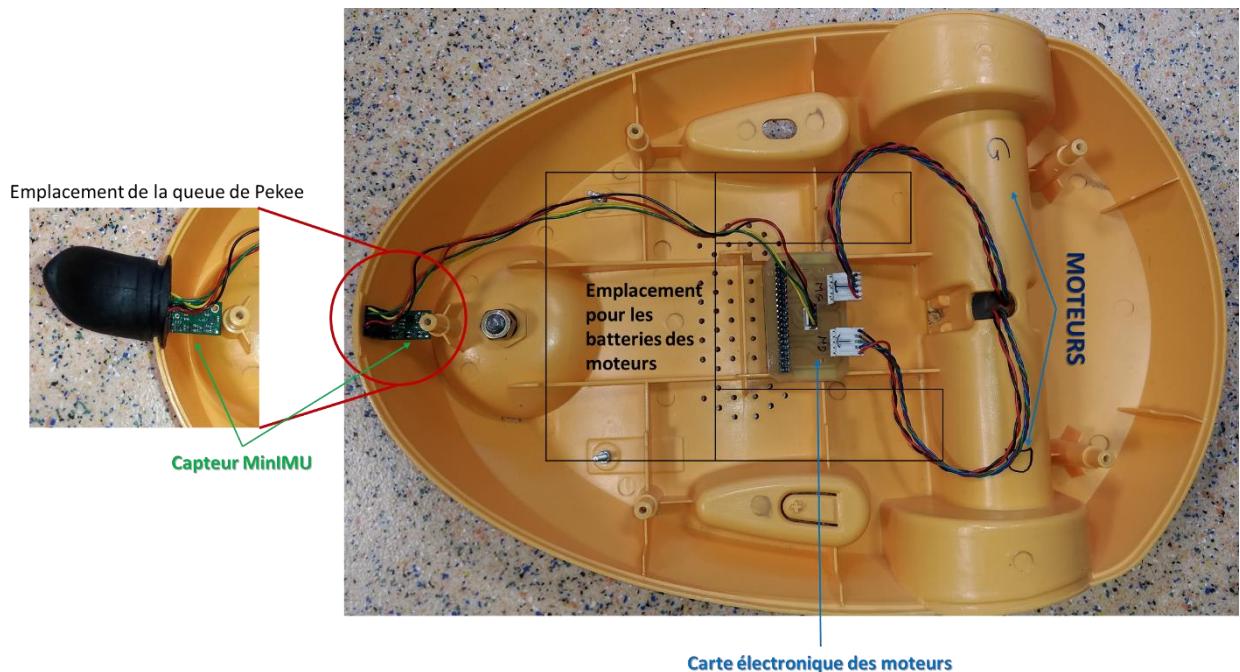


Figure 25 : Décomposition de la base de Pekee

La carte des moteurs a été conçue pour permettre d'alimenter à la fois les moteurs et les encodeurs (présent au niveau des moteurs) ainsi que le capteur MinIMU. Elle permet également de transmettre la tension à appliquer sur chacun des moteurs, reçue depuis le pont en H que nous verrons à l'étage supérieur. Enfin, elle permet de remonter les informations transmises par le capteur MinIMU et les encodeurs à l'Arduino au travers de la nappe.

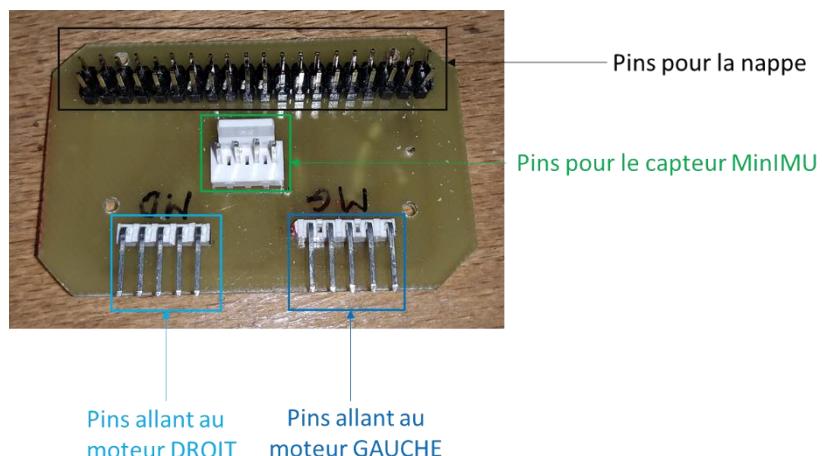


Figure 26 : Carte des moteurs

Concernant le branchement de la nappe, il est important de respecter le code couleur que nous avons pu mettre en place sur chacune de nos cartes, à savoir : le côté rouge de la nappe se branche sur le côté rouge de la carte comme l'illustre la figure suivante.

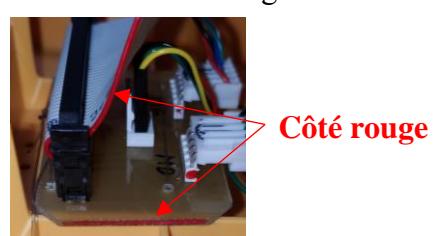


Figure 27 : Branchement de la nappe

Pour le branchement des moteurs nous avons mis une flèche sur le dessus des pins indiquant comment les connecter. Et pour le capteur MinIMU, il est important de respecter le câblage comme illustrer sur la figure 29.

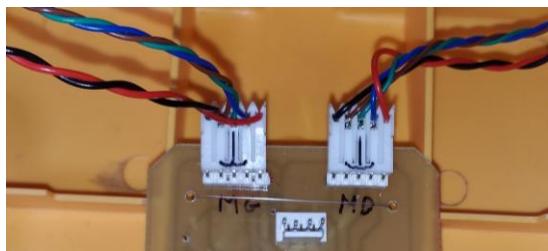


Figure 28 : Branchement des moteurs

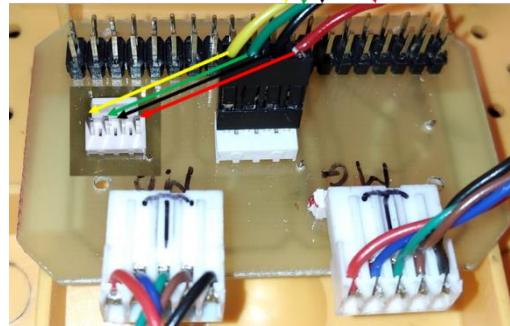
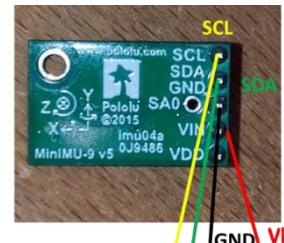


Figure 29 : Branchement du capteur MinIMU

Remarque : Comme vous pourrez le constater nous avons oublié de faire le report des pins de la nappe. Suite à une erreur de conception, nous ne pouvons donc pas ajouter de nouveaux capteurs sur cette carte. Dans la partie piste d'évolution, nous suggérons que la prochaine équipe refasse cette carte électronique. Pour cela nous avons joint, l'ensemble des schématiques et boards conçus sur le logiciel Altium sur le Github dans le dossier « Cartes électroniques ». Cette carte est nommée PekeeCarte1.

Par ailleurs, une autre erreur a été faite lors de la conception de cette carte. En effet, nous avons joint la borne négative des moteurs à la masse commune, ce qui n'est pas le cas car les moteurs sont alimentés avec une tension variable non-alignée sur la masse. Nous avons donc corrigé cette erreur sur le schématique et le board fourni et nous avons corrigé manuellement notre erreur sur la carte en rayant certaines pistes et en faisant des ponts vers les bonnes pistes.

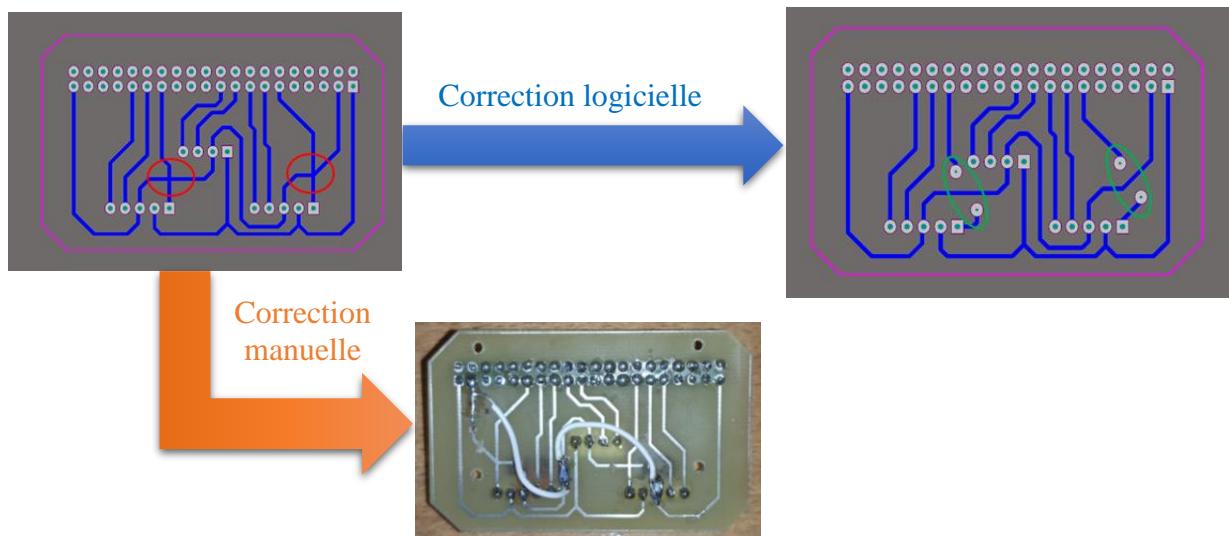


Figure 30 : Correction de l'erreur de conception (erreur en rouge sur le board de gauche et correction en vert sur le board de droite)

➤ La tête de Pekee

La tête de Pekee constitue la partie haute. Celle-ci contient 2 sous-parties, l'avant et l'arrière de la tête, séparé par le cerveau.

Sur la partie avant, nous retrouvons le pont en H ainsi que la carte électronique associée à celui-ci que nous avons conçu.

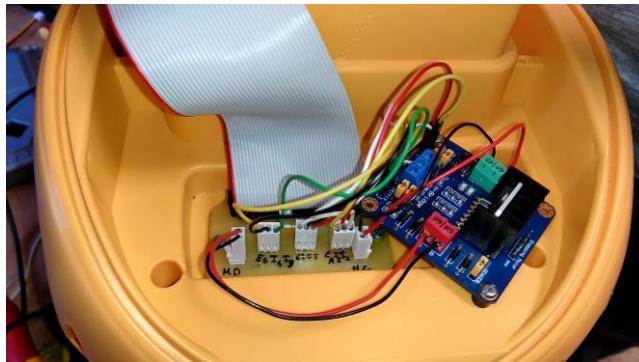
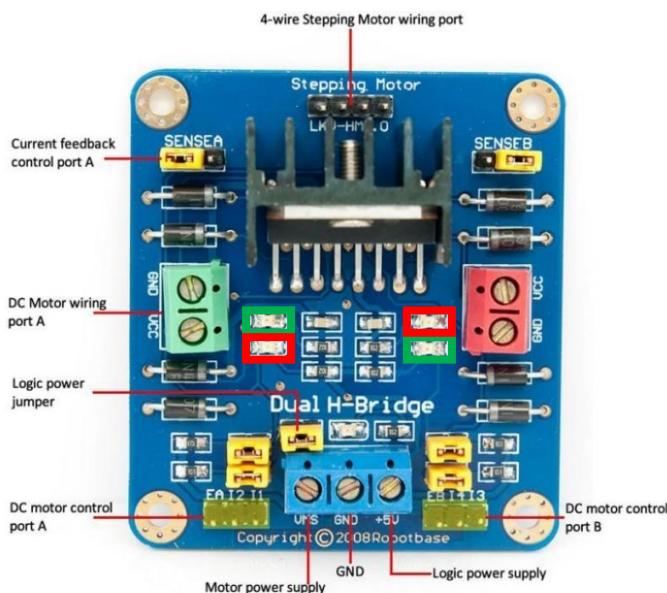


Figure 31 : L'avant de la tête de Pekee contenant le pont en H et la carte électronique associée

Comme nous avons pu le dire dans la présentation du nouveau matériel, nous allons détailler le pont en H à l'aide de la Figure 14 que nous remettons ici.



Rappel Figure 14 : Caractéristiques du Pont en H

Un point important sur lequel nous attirons votre attention : le **jumper Logic Power** doit être mis dès que la tension appliquée sur les moteurs, via la broche **Motor power supply** dépasse les 20V autrement vous risquer de griller le régulateur présent sur la carte.

Le pont est alimenté en 5V via la broche **Logic Power Supply**. Les moteurs sont connectés sur les borniers vert et rouge. Dans notre configuration, nous avons choisi que le bornier rouge correspond au moteur droit et le bornier vert au moteur gauche. Ces borniers ont un GND et une tension VCC (qui correspond aux lettres A et B sur notre pont). Toutefois, la masse n'est pas la masse commune comme nous avons pu l'expliquer dans la remarque précédente. Nous avons donc branché la borne A sur la borne positive du moteur et la borne B sur la borne négative.

Enfin, c'est à partir des pins EA, I2 et I1 que l'on commande le moteur gauche (bornier vert) et les pins EB, I4 et I3 servent à commander le moteur droit (bornier rouge).

Le pont dispose également de 2 LEDs indicatrices sur chacun des borniers. Selon le code couleur, nous pouvons savoir la commande envoyée et connaître le sens de rotation des moteurs. Pour résumer ce code couleur et le sens des rotations des moteurs, nous regroupons les différentes possibilités dans le tableau suivant. La notation H correspond à l'instruction HIGH et L à l'instruction LOW.

EB (EA)	I4	I3	I2	I1	Bornier Vert	Bornier Rouge	Sens de rotation des moteurs		
					Led Verte	Led Rouge	Led Verte	Led Rouge	
Consigne Vitesse (allant de 0 à 255)	L	L	L	L	1	1	1	1	Pas d'entraînement moteur (roue libre)
	L	H	H	L	1	0	1	0	Sens horaire
	H	L	L	H	0	1	0	1	Sens anti-horaire
	H	H	H	H	0	0	0	0	Blocage moteur

Après cette étude, nous avons donc conçu la carte électronique qui regroupe l'ensemble des entrées/sorties du pont en H, à savoir les entrées I1, I2 et EA avec les sorties du bornier vert pour le moteur gauche et les entrées I3, I4 et EB avec les sorties du bornier rouge pour le moteur droit. Nous avons également les pins d'alimentation VMS pour les moteurs et le +5V pour le pont en H ainsi que la masse GND.

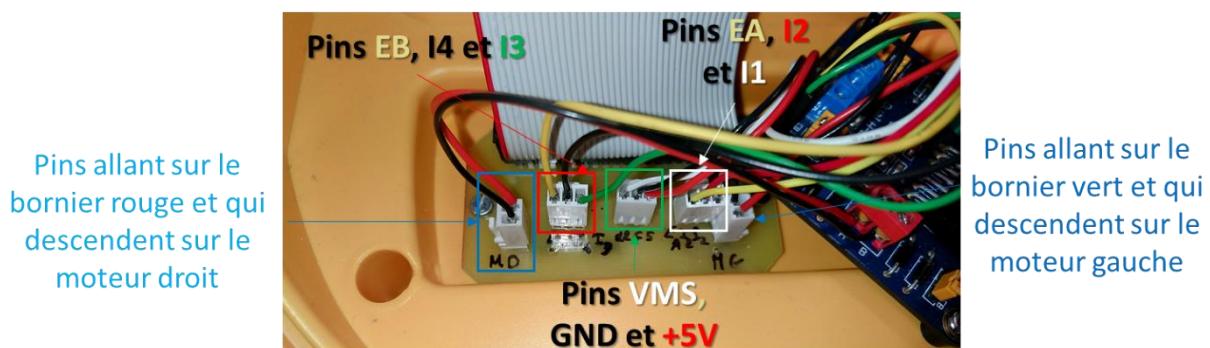


Figure 32 : Branchements des différents pins sur la carte associée au pont

Remarque : Vous trouverez le schématique et le board associé à cette carte dans le dossier « Cartes électroniques » sur Github. Cette carte correspond à la carte n°2 d'où le nom PekeeCarte2.

Concernant la partie arrière de la tête, nous trouvons un emplacement permettant de stocker l'ensemble des câbles d'alimentations de notre système.

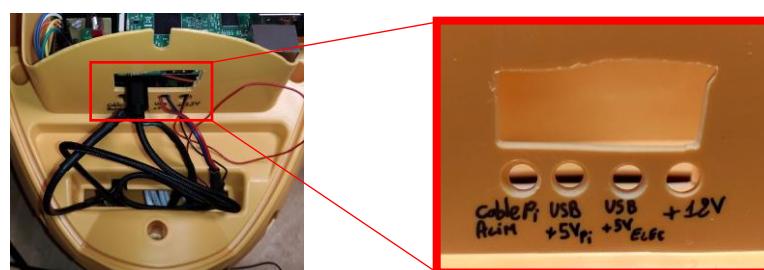


Figure 33 : Zone arrière de la tête de Pekee (emplacement réservé aux câbles d'alimentation)

➤ Le cerveau de Pekee

Le cerveau de Pekee est composé de 3 couches : la batterie 5V, le relais et la carte associée pour l'alimentation du système, et la dernière couche accueille les cartes Arduino et Raspberry ainsi que le bouton On/Off du système.

❖ La 1^{ère} couche : Batterie 5V

Chaque couche est séparée par une plaque de plastique réalisée en impression 3D par des camarades de la filière Mécanique Mécatronique. En bas du cerveau, on trouve la batterie 5V. Elle possède deux ports USB ce qui nous permet d'alimenter séparément le Raspberry Pi du reste du circuit électronique. Nous avons fait ce choix afin d'assurer une alimentation plus stable à la Pi. Le câble de recharge de la batterie est accessible depuis la queue de Pekee. La seule chose non-accessible sur la batterie est son niveau de charge, toutefois une astuce pour voir son niveau de charge est de regarder la petite lueur bleue en soulevant la tête de Pekee. Les leds clignotent durant la recharge et elles s'éteignent lorsque la batterie est complètement chargée. Il faut compter approximativement 6h pour une recharge complète. La batterie se recharge soit depuis un port USB d'un ordinateur, soit depuis un chargeur de portable universel.



Figure 34 : Première couche du cerveau →
Emplacement de la batterie 5V

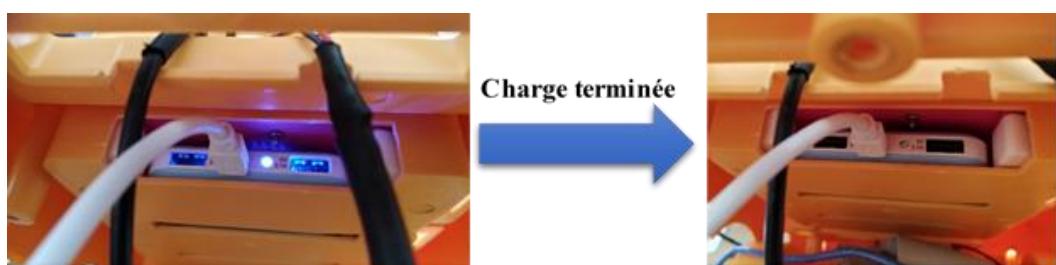


Figure 35 : Astuce pour suivre le chargement de la batterie (vue de dessous de la tête) :
batterie en charge (à gauche) et batterie chargée (à droite)

Juste au-dessus de cette batterie nous trouvons la 2^{ème} couche du cerveau. La délimitation de ces deux couches est marquée par la PekeeTable. Cette carte 3D a été conçue pour accueillir la carte électronique de la gestion de l'alimentation du système ainsi que le module des relais.

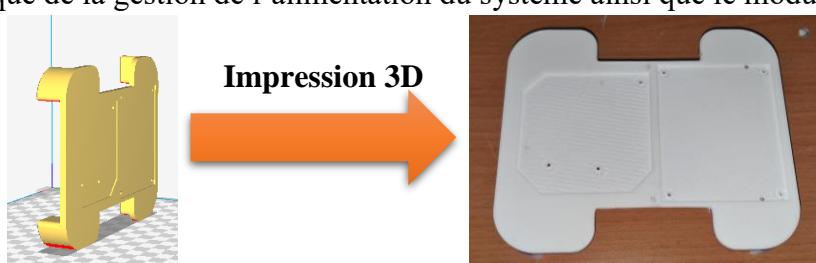


Figure 36 : La PekeeTable (modélisation 3D à gauche et support réel à droite)

Remarque : Vous pourrez trouver le fichier 3D associé à ce support sur le Github dans le dossier « PekeeBody ».

❖ La 2^{ème} couche : La gestion de l'alimentation

Sur cette couche nous pouvons retrouver la carte de module des relais ainsi que la carte électronique associée à ce module. La gestion de l'alimentation du système a été une partie délicate du fait que nous ayons un Raspberry. En effet nous ne pouvons pas couper directement l'alimentation de celui-ci au risque d'endommager le système d'exploitation. Pour pallier à ce problème, c'est le Raspberry Pi qui va être responsable de la fermeture et l'ouverture du système d'alimentation. C'est lui qui commande les différents relais du système.

La particularité de la carte des modules de relais est que la fermeture d'un module est faite lorsque nous appliquons une masse sur le pin de commande (comme décrit sur la *Figure 17*). Nous allons donc utiliser cette particularité pour gérer l'alimentation du système. Le rôle du relais est de permettre le maintien de l'alimentation du circuit pendant l'extinction du Raspberry.

L'astuce réside dans le fait que lorsque le Raspberry Pi est allumé, il applique la masse de son alimentation à ses pins (à condition d'avoir correctement configuré le GPIO au démarrage de la PI comme nous pourrons le voir dans la partie suivante). Or lorsqu'il est éteint la masse devient flottante. Cela nous permet donc de commander les différents relais grâce à cette masse flottante ou non-flottante.

Nous avons décidé d'utiliser 3 relais sur 4 :

- Un relais 5V pour le Raspberry Pi,
- Un relais 5V pour le reste du système électronique, Relais pour la PI
- Un relais 12V pour les moteurs.
- Le dernier relais n'est pas utilisé pour le moment.

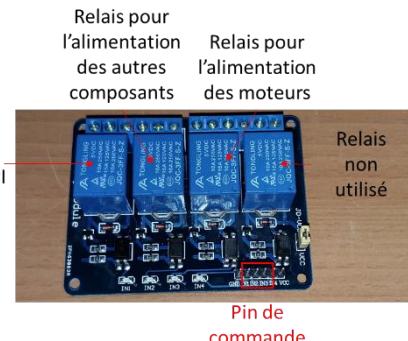


Figure 37 : Carte des modules de relais

À ces relais, nous avons ajouté une carte électronique complémentaire qui va gérer tout le système d'alimentation. Cette carte est légèrement plus complexe que les précédentes car elle permet de faire les liaisons entre la carte des relais, les différentes batteries et le bouton ON/OFF que l'on vous présentera par la suite. Elle ressemble à cela :

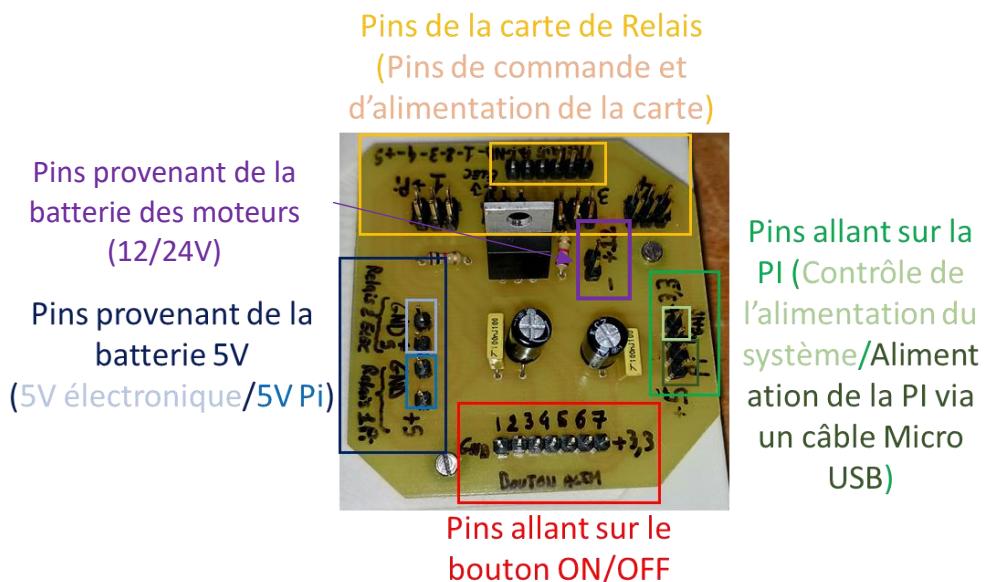


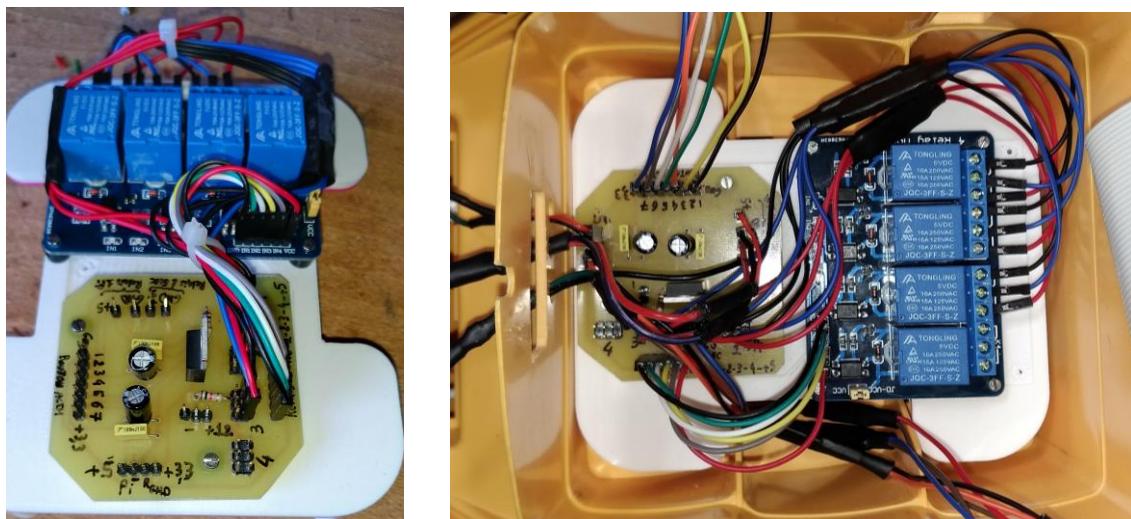
Figure 38 : Carte électronique de la gestion de l'alimentation de Pekee

Nous distinguons sur cette carte 4 grandes familles de pins :

- Ceux des batteries 5V et 12/24V (en bleu et violet sur la Figure 38). Ils permettent d'apporter les différentes alimentations sur la carte.
- Les pins allant au bouton ON/OFF (en rouge). Ces pins servent simplement à remonter les bonnes pistes à la couche supérieure où se trouve le bouton d'alimentation.
- Les pins allant au Raspberry PI (en vert). Une partie de ces pins servent à fournir l'alimentation sur le port MicroUSB du Raspberry (en vert foncé) et les autres pins permettent de gérer l'alimentation du système grâce à un pin de masse et un autre pin utilisé pour détecter la demande d'extinction du circuit comme nous pourrons le voir par la suite (en vert clair).
- Les pins allant au relais (en orange). Ils permettent de regrouper tous les pins de la carte de relais sur cette carte.

Enfin, nous avons des condensateurs, de 100nF et 100µF, pour filtrer les tensions d'alimentation de la batterie 5V et des résistances de 10k Ohms faisant office de résistance pull-down.

Nous avons également un régulateur de tension 5V vers 3.3V pour nous permettre de détecter l'ouverture du bouton à l'étage supérieur. Cette astuce va nous permettre sur le Raspberry de détecter un front montant car lorsque le bouton ON/OFF va s'ouvrir (position OFF) il fera arriver du 5V sur le régulateur, le circuit étant auto-alimenté grâce au relais, qu'il transformera en 3.3V. La raison de cette adaptation est la tension limite sur le port GPIO du Raspberry. Cette tension sera lue par un pin du Raspberry. Ce pin verra donc un front montant et ordonnera l'extinction de la Pi pour permettre de mettre hors tension l'ensemble du système une fois la PI éteinte.



*Figure 39 : La PekeeTable avec la carte de relais et la carte électronique (à gauche) ;
branchement complet de la 2^{ème} couche du cerveau de Pekee (à droite)*

Remarque : La carte électronique conçue se trouve dans le dossier « Cartes électroniques » sur Github et elle se nomme carteRelais.

❖ La 3^{ème} couche : La partie intelligente

Cette dernière partie regroupe la carte Arduino MEGA, le Raspberry et le bouton d'alimentation du système. Cette dernière couche est séparée par une seconde table modélisée en 3D (le fichier est fourni dans le dossier PekeeBody).

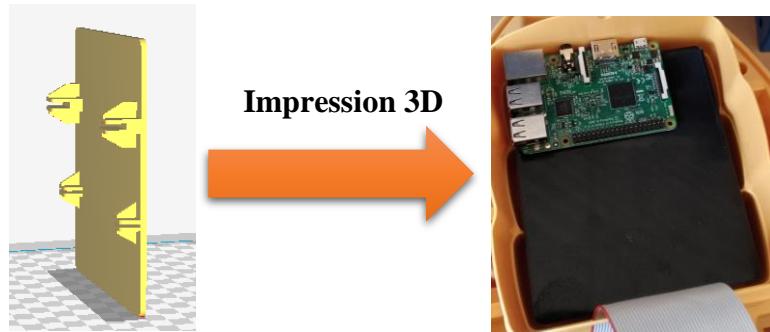


Figure 40 : Le support du cerveau de Pekee (modélisation 3D à gauche et le support intégré dans le cerveau de Pekee à droite)

Afin de pouvoir brancher la nappe sur l'Arduino MEGA, nous avons conçu un shield pour l'Arduino. Une autre carte a été réalisée, il s'agit de la carte pour le bouton d'alimentation qui comporte le bouton ON/OFF et une led indiquant si le système est sous tension ou non.

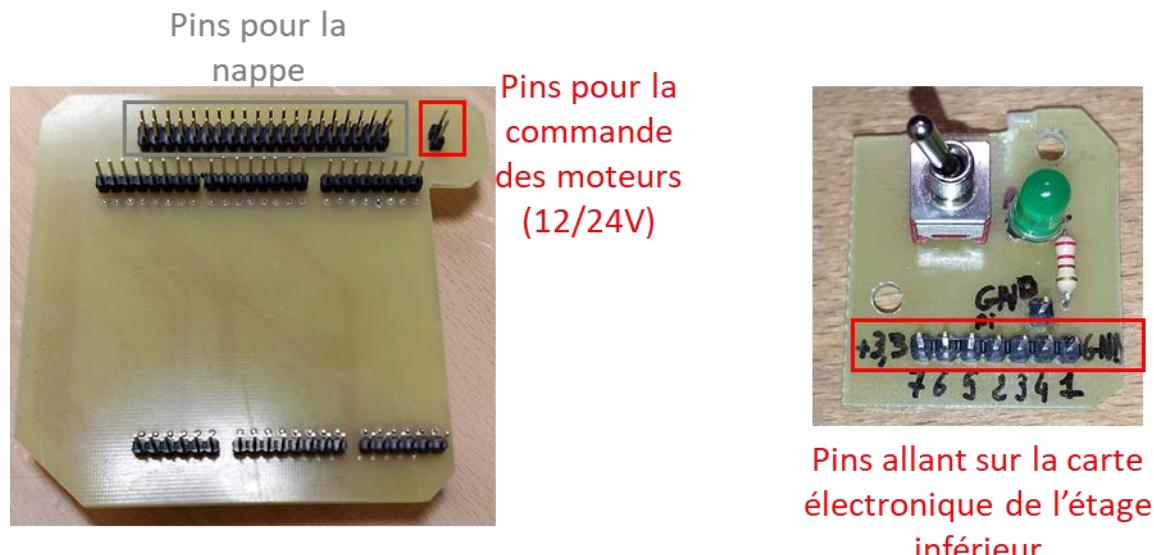


Figure 41 : Shield Arduino (à gauche) et carte du bouton d'alimentation (à droite)

Remarques : Nous avons également fourni le board et le schématique de ces 2 cartes dans le dossier « Cartes électroniques » sous le nom de CarteShieldArduino et CarteBouton.

Par ailleurs, comme vous avez pu le constater sur les différentes figures nous avons essayé de mettre pour chaque pin sa signification sur la carte afin de faciliter les branchements et de savoir où se branche tel pin en cas de débranchement.

Une fois le tout monté, le robot Pekee 2.0 ressemble à ceci :



Figure 42 : Pekee 2.0 assemblé

➤ Résumé des connexions

Afin de résumer les différents câblages réalisés, nous avons décidé de vous mettre ici les pins utilisés sur l'Arduino MEGA et par la nappe ainsi que le schéma électronique simplifié de la gestion de l'alimentation du système.

❖ Pins utilisés sur l'Arduino MEGA & Pins de la nappe

Pour synthétiser, voici un tableau récapitulatif des pins utilisés. Pour la convention des pins utilisés par la nappe voici notre numérotation :



Figure 43 : Numérotation des pins de la nappe

Numéro Pin NAPPE	Pin ARDUINO MEGA	Signification
40	5V	Alimentation Pont/Capteurs
39	GND	Masse
38	Digital 13	Commande Pont en H : I1
37	Digital 12	Commande Pont en H : I2
36	Digital 11	Commande Pont en H : EA
34	Digital 3	Encodeur Moteur Gauche (pin interruption)
32	Digital 20	SDA (Liaison I ² C pour le capteur MinIMU)
31	Digital 21	SDL (Liaison I ² C pour le capteur MinIMU)
30	Digital 10	Commande Pont en H : I1
29	Digital 9	Commande Pont en H : I1
28	Digital 8	Commande Pont en H : EB
26	Digital 2	Encodeur Moteur Droit (pin interruption)
22	Non connecté	Borne positive batterie moteur
21	Non connecté	Borne négative batterie moteur (Masse)
Pins de 1 à 20	NON CONNECTÉ	Pins non utilisé
Pins 35,33,27,25,24,23	NON CONNECTÉ	Pins non utilisé

Nous avons également utilisé le port USB de l'Arduino pour faire la liaison série avec le Raspberry.

❖ Schéma électrique du système d'alimentation

Faute de temps, nous vous donnons la version papier de notre schéma électrique (que nous avons mis au propre).

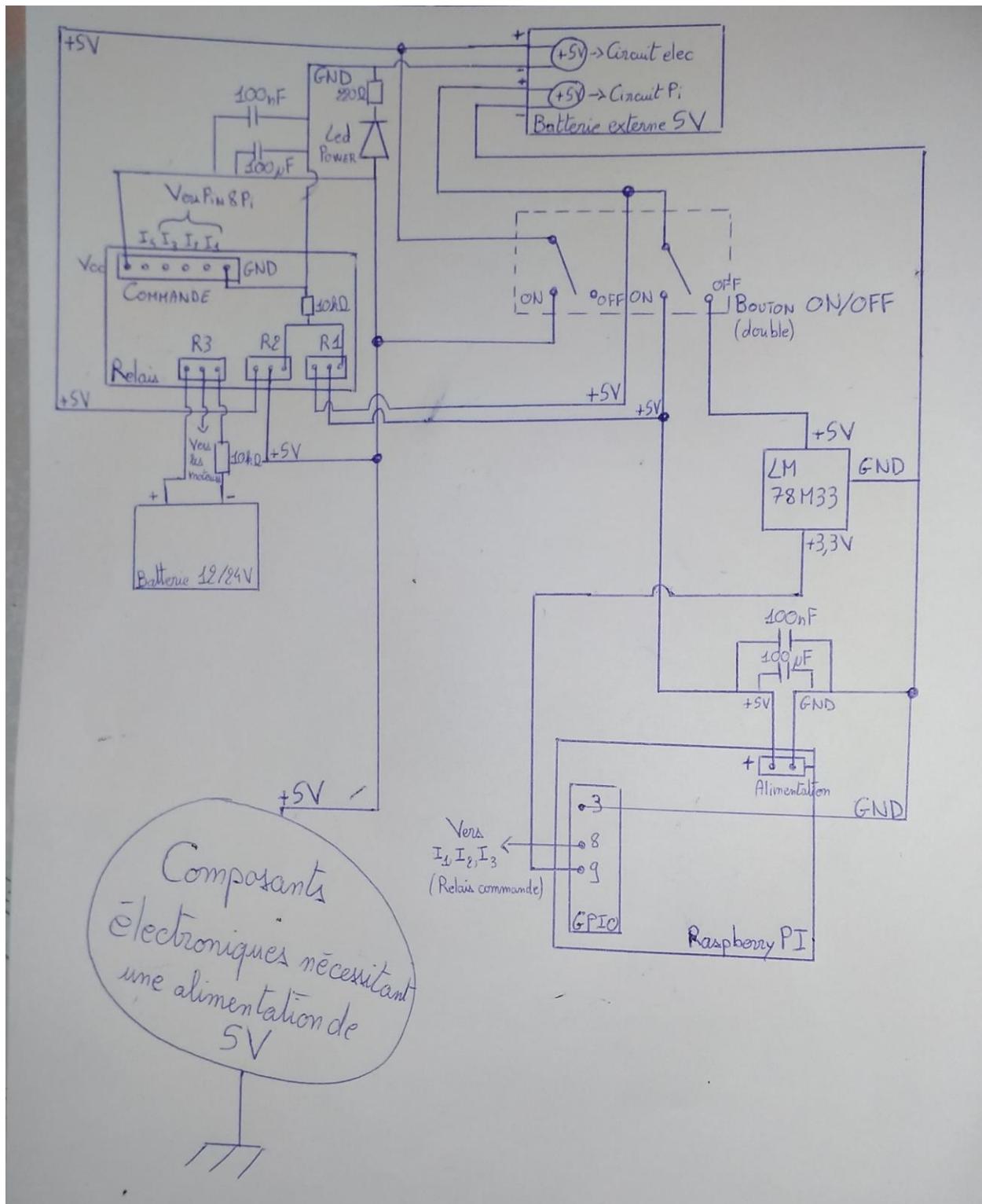


Figure 44 : Schéma électrique du système d'alimentation de Pekee

2) Étude des capteurs

Comme nous avons pu le présenter dans les ajouts de matériel, Pekee 2.0 embarquera des capteurs infrarouges et un capteur 9 axes. Le capteur 9 axes est déjà embarqué sur Pekee et il se situe à l'arrière de Pekee afin d'être au plus près de la roue folle et permettre une correction rapide de la trajectoire. Toutefois, nous nous sommes penchés au début du projet sur son étude mais n'avons pas réussi à l'exploiter correctement. Par ailleurs, il ne s'agit pas d'un capteur analogique mais numérique. Par conséquence les données sont nécessairement de type numérique. Nous verrons donc dans la prochaine partie le programme utilisé pour récupérer les valeurs des différents capteurs qu'il intègre.

Concernant les capteurs infrarouges, nous les avons reçus trop tard pour pouvoir les embarquer sur le robot. Ce sera une problématique lors de la reprise du projet. Cependant nous avons pu faire l'étude analogique de ces capteurs.

Dans un premier temps, nous avons vérifié les caractéristiques techniques du capteur en faisant son étalonnage. Le protocole d'étalonnage a été le suivant :

« À l'aide d'un appareil de mesure de tension, mesurer tous les 0.5cm la valeur analogique renvoyée par le capteur sur une distance de 0 à 40cm ».

En effet, le capteur est alimenté par une tension de 5V et il retourne une tension comprise entre 0 et 3V qui est proportionnelle à la distance de détection. L'objectif était donc de trouver cette relation entre la tension et la distance de détection. La courbe obtenue est alors la suivante :

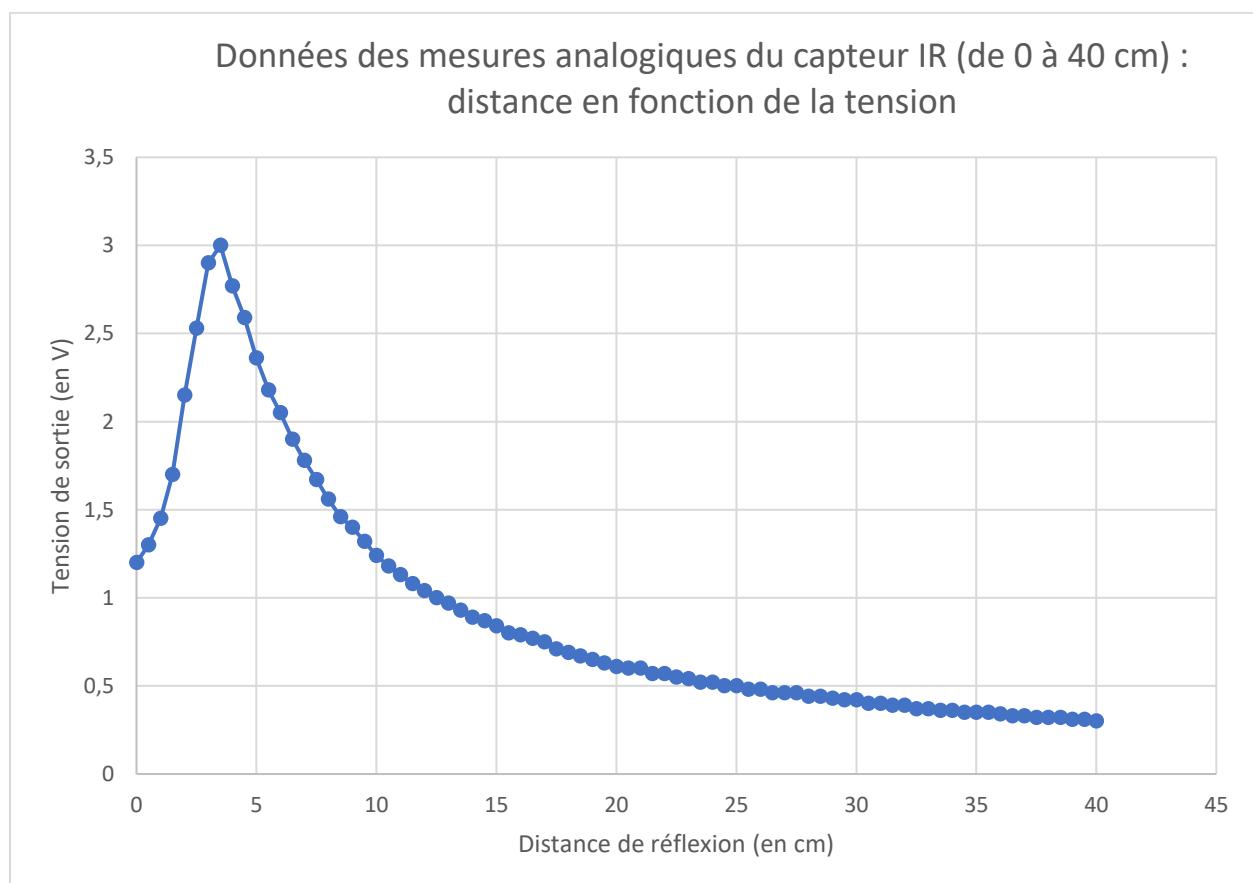


Figure 45 : Courbe obtenue après étalonnage du capteur

Remarque : Les mesures réalisées ont été jointes dans un fichier Excel présent sur Github dans le dossier « Documentation ».

On distingue sur cette courbe 2 zones de fonctionnement : une quasiment linéaire (pour une distance comprise entre 0 et 3.5cm) et une exponentiellement décroissante (sur une distance de réflexion comprise entre 3.5cm et 40cm).

Le capteur sera donc utilisé sur la 2^{ème} zone de fonctionnement et nous considérons qu'à une distance de 3.5cm, le robot a détecté un obstacle et il doit le contourner ou faire demi-tour. La fonction obtenue permettant de remonter à la distance de réflexion à partir de la tension mesurée sur cette zone de fonctionnement est la suivante :

$$Distance = 11.855 * U_{mes}^{-0.987}$$

Cette équation peut être retrouvée dans le classeur Excel fourni.

Les capteurs infrarouges sont donc opérationnels pour être intégrés sur Pekee. Il manque simplement la déportation des pins de la nappe sur la carte 1 pour permettre le branchement de ces capteurs sur l'Arduino, mais nous reviendrons sur cette piste lors de la partie des évolutions. Néanmoins nous conseillons de réaliser une ceinture avec ces capteurs autour de Pekee : 3 devants et 2 de chaque côté, soit un total de 7 capteurs.

En conclusion de cette partie, nous avons pu proposer une nouvelle architecture matérielle pour le robot. Aujourd'hui le robot est quasiment opérationnel d'un point de vue électronique. Il ne lui manque que la batterie des moteurs (12V ou 24V se sera à décider) et l'intégration des capteurs infrarouges. Une fois que l'électronique du robot faite, il nous a fallu programmer les cartes Arduino et Raspberry pour nous permettre de mettre en route le système et piloter le robot. C'est ce que nous allons aborder dans la partie suivante.

V. Conception logiciel Arduino & Raspberry

À présent que la partie électronique est fonctionnelle, il nous faut exploiter le robot. Pour ce faire nous allons utiliser un Arduino Mega pour la partie bas niveau et un Raspberry Pi pour la partie haut niveau. En effet l'idée à terme est d'avoir un robot à deux couches d'intelligence. Une première intelligence peu évoluée mais rapide afin de permettre le contrôle en temps réel du robot, et ce, manuellement ou logiciellement. Une deuxième beaucoup plus poussée embarquant son propre système d'exploitation et offrant un véritable ordinateur mobile.

1) Arduino

L'Arduino est responsable de la communication entre le Raspberry Pi et l'Arduino : de la commande en temps réel des moteurs jusqu'à la gestion des capteurs.

La communication se fait par liaison série entre le Raspberry Pi et l'Arduino. En effet, nous avons décidé de changer de liaison série par rapport aux étudiants précédents car la liaison I2C est déjà utilisée par le capteur MinIMU sur l'Arduino. Par ailleurs, la liaison série nous offre de plus grande possibilité car nous pouvons programmer notre Arduino à distance grâce au Raspberry sur lequel nous avons installé l'IDE d'Arduino.

La plus grande problématique de cette partie concerne la gestion des buffers. Dans un premier temps, nous avions choisi de formaliser les transferts via un string. Nous voulions pouvoir envoyer plusieurs commandes et paramètres en même temps. Les différentes commandes étant séparées par un ‘&’ et les paramètres par des ‘:’. Un caractère de fin de message avait également été mis en place, le ‘!’. Cela nous donnait donc un message de la forme « m&50:50 ». Le buffer était lu jusqu’au caractère de fin de message puis le string était séparé en différents substring avant d’être traité. Ce raisonnement est malheureusement difficile et coûteux à réaliser en C.

Nous avons rapidement remarqué un décalage au niveau de l'Arduino dû au temps d'exécution de ce traitement. Nous avons donc choisi de changer de paradigme et de fixer la taille du message. Le premier caractère est l'ordre, les 4 suivants contiennent le premier paramètre et les 4 d'après le second. Ce paradigme beaucoup moins souple à l'évolution est cependant beaucoup plus rapide car il n'y a aucune analyse nécessaire pour sa description.

Autre détail la communication se fait par interruption, un event est déclenché lorsqu'une donnée arrive dans le buffer de l'Arduino.

La commande des moteurs se fait à l'aide du pont en H. Celui-ci nous permet de commander simplement les moteurs grâce à 4 pins d'état (I1, I2, I3 et I4) et 2 pins analogiques (EA et EB). Les pins d'état permettent de définir le sens de rotation du moteur (comme nous avons pu le voir dans la partie IV sur la description du pont en H en page 18) et les pins analogiques définissent la vitesse de celui-ci. Étant donné qu'il s'agit d'une valeur analogique, cette vitesse est comprise entre 0 et 255 où 0 correspond à la vitesse minimale et 255 à la valeur maximale.

Parallèlement à tout ceci l'Arduino s'occupe des différents capteurs : les encodeurs, les capteurs IR, le capteur MinIMU...

Les encodeurs ne sont pas équipés de buffer, s'est donc l'Arduino qui est responsable du décompte de ceux-ci. Ils sont branchés sur deux pins d'interruptions qui font appel à une simple routine d'incrémentation. À partir de ce moment-là, on peut, grâce à la communication, demander la valeur actuelle des encodeurs ou bien la réinitialiser.

La valeur des encodeurs est codée sur un entier non signé, allant jusqu'à 4 294 967 295. Cette valeur est relativement grande et de ce fait aucune gestion de débordement n'est prévu pour le moment.

Pour plus d'informations sur la commande des moteurs et l'étude des encodeurs, nous vous suggérons de vous reporter au travail réalisé par les étudiants de l'année 2016/2017 qui ont fait une bonne étude sur ces moteurs/encodeurs.

Remarque : Le programme de pilotage se trouve dans le dossier « Arduino → Pekee ».

➤ Programmes Arduino permettant l'étude des capteurs

Pour nous permettre d'exploiter logiciellement les capteurs, nous avons pu récupérer les librairies associées à ceux-ci, à savoir la librairie lsm6 et lis3mdl pour le capteur MinIMU et la librairie Sharp-IR pour le capteur infrarouge.

Si vous suivez le lien fourni lors de la présentation du matériel pour le capteur MinIMU, vous pourrez trouver un programme Arduino permettant l'acquisition des différentes données fournies par le capteur. Après de longues heures passées dans la documentation technique pour comprendre les données reçues dans le Serial Monitor de l'Arduino, nous avons abandonné son étude pour nous concentrer sur la partie électronique. Vous pourrez retrouver le programme Arduino permettant d'exploiter le capteur, dans le dossier « Arduino → minimu-9-ahrs-arduino-master ».

Toutefois, grâce à l'aide de M VERNIER, nous avons pu nous apercevoir que nous détections des variations de changement de valeur lorsqu'une perturbation apparaissait sur le capteur comme nous pouvons le voir sur la figure ci-contre.

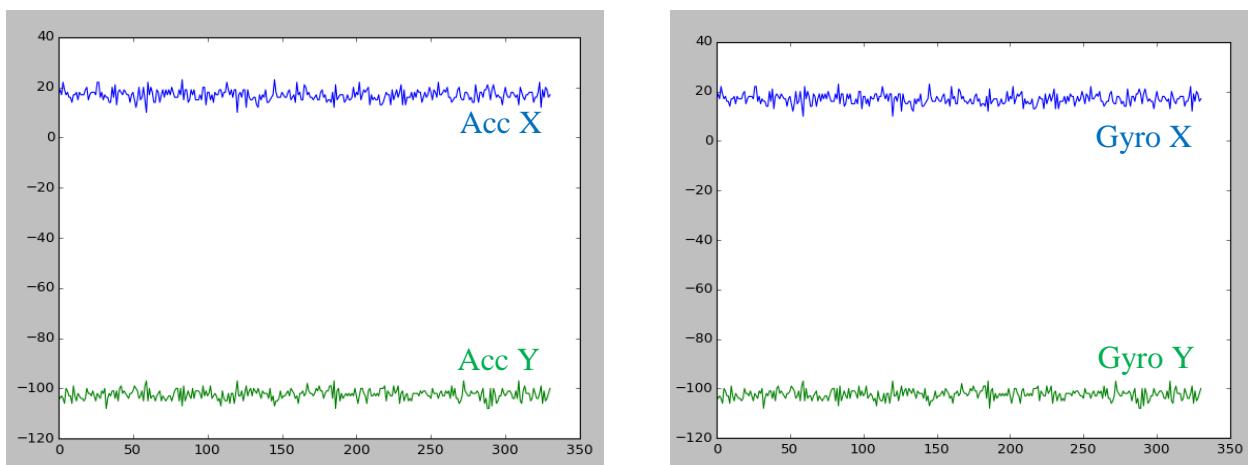


Figure 46 : Courbes obtenues de l'accéléromètre (à gauche) et du gyromètre (à droite) sans perturbation

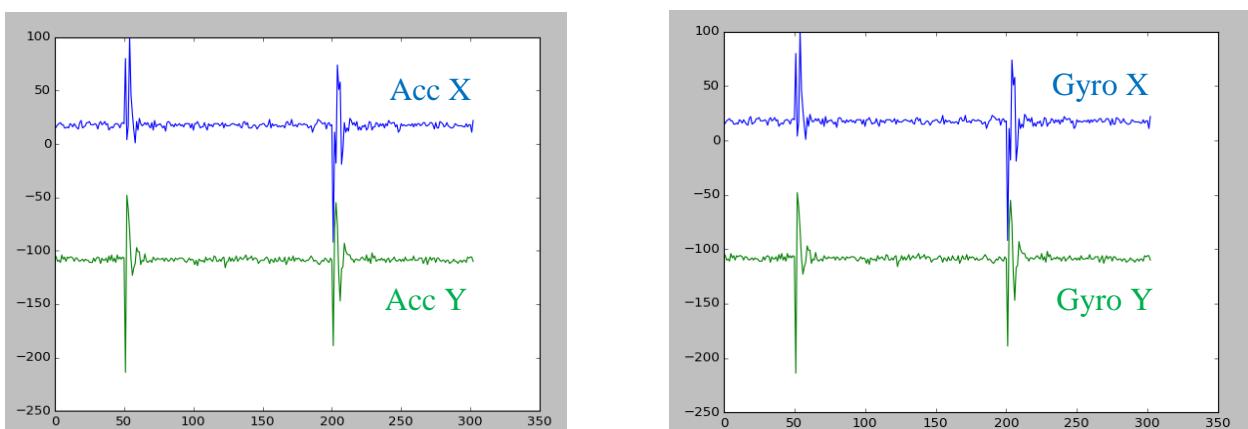


Figure 47 : Courbes obtenues de l'accéléromètre (à gauche) et du gyromètre (à droite) avec perturbation

Remarque : Ces données sont disponibles dans le dossier « Données capteur 9 axes » sur le Git.

On peut donc remarquer que nous détectons les perturbations et que le capteur fonctionne parfaitement. Ici la perturbation était aléatoire, on ne peut pas évaluer la trajectoire qui correspond à ces courbes. Néanmoins le travail a réalisé pour la suite du projet, maintenant que le robot est entièrement commandable, est d'évaluer les données à partir d'une trajectoire connue. Nous suggérons de faire une fenêtre de moyennage et de définir un seuil à partir duquel une perturbation doit être détectée.

Par ailleurs, un programme Python permet de visualiser en 3D les données du magnétomètre, reçues via la liaison série entre l'Arduino et le PC. Cette visualisation 3D peut également permettre d'évaluer le capteur.

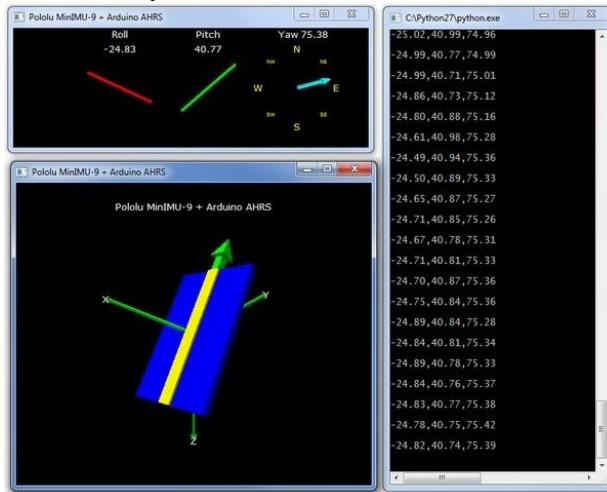


Figure 48 : Visualisation 3D du capteur

Ce capteur est particulièrement complexe car il nécessite une programmation des différents registres qui n'est possible après une étude minutieuse de la documentation technique des capteurs qu'il intègre. Dans un premier temps, nous vous conseillons de laisser les paramètres par défaut et de ne pas toucher aux registres, simplement étudier les données reçues lorsque vous pilotez le robot.

Dans un second temps, vous pourrez essayer de configurer les registres pour obtenir une meilleure précision.

Nous vous conseillons de bien lire l'ensemble des documents fournis pour ce capteur ainsi que le Readme du github du capteur : <https://github.com/pololu/minimu-9-ahrs-arduino>

Concernant le capteur infrarouge, nous avons réutilisé la librairie fournie (au lien donné lors de la présentation du matériel pour ce capteur) avec le programme Arduino donné. Ce programme peut être trouvé dans le dossier « Arduino → Sharp-IR-master ». Il permet simplement de retourner la distance de détection de l'obstacle. Nous ne l'avons pas intégré à notre programme Pekee car nous vous laisserons juger à partir de quelle distance vous considérez avoir détecté un obstacle et quelle action vous envisagez.

Ce capteur nécessite d'être connecté à une broche analogique de l'Arduino et d'être alimenté par une tension de 5V.



Figure 49 : Branchement du capteur infrarouge

2) Raspberry

Le Raspberry PI est le cerveau du système.

Sa première mission est la gestion de son alimentation. Cette gestion se fait en arrière-plan grâce à un petit programme qui se lance automatiquement au démarrage (fichier nommé Start.py). Ce programme met le pin 8 à la masse, permettant la fermeture des relais et surveille le pin 9 afin de détecter le signal d'extinction. Lorsque ce signal est détecté la PI lance sa phase d'extinction proprement et lorsque celle-ci est finie, la masse devenant flottante ouvre les relais ce qui met la totalité du robot hors tension.

Remarque : Le script utilisé est celui d'un internaute : <http://raspi.tv/2013/how-to-use-interrupts-with-python-on-the-raspberry-pi-and-rpi-gpio>

Sa deuxième mission est l'interaction et la commande du robot à plus haut niveau. Elle a pour objectif à terme de permettre différents niveaux d'abstraction sur le robot. Elle permet aussi d'interagir à distance avec le robot, que ce soit pour le programmer ou pour le contrôler. Cela est possible pour le moment grâce au SSH mais par la suite cela pourrait très bien se faire via une interface web ou tout autre technologie.

La PI permet également de diriger Pekee grâce à un joystick. Celui-ci ainsi que ses codes d'exploitations nous ont été fournis par M VERNIER et nous avons simplement réadapté le code à notre système. Cette adaptation a principalement consisté à l'exploitation de la communication série entre la Pi et l'Arduino. Cela nous a permis de tester les limites de celle-ci et la bonne réactivité de notre système.

Nous avons également réduit la sensibilité du joystick en rajoutant différents seuils afin de ne pas saturer la liaison série.

Pour piloter le robot, cela est possible uniquement avec le joystick. Voici donc une illustration qui vous présente comment piloter le robot avec ce joystick.



Figure 50 : Fonctionnement du Joystick pour piloter Pekee

Pour pouvoir piloter Pekee, nous avons défini que le bouton 7 permettait de démarrer le programme de gestion du joystick et le bouton 8 de l'arrêter. Il est bien évidemment possible de réexécuté le programme en appuyant sur le bouton 7. En effet, nous avons décidé de lancer le programme en arrière-plan et de détecter l'appui sur le bouton 7. Il faut savoir que l'on peut utiliser ce joystick grâce à la librairie Pygame de python. Une fonction de cette librairie permet de détecter tous les event sur le joystick que nous pouvons récupérer grâce à un dictionnaire et ainsi agir en conséquence.

Nous avons prévu que le robot pouvait se déplacer dans toutes les directions possibles. Pour cela, selon l'angle d'inclinaison du joystick, nous appliquons un différentiel de tension sur les moteurs afin de se rapprocher au maximum de la commande. Toutefois, il n'est pas possible de déplacer le joystick selon l'axe y (car physiquement le robot n'en est pas capable). Nous bloquons donc les moteurs lorsque la valeur du joystick en y vaut 0.

Le point d'arrêt se trouve au centre, c'est-à-dire en $x=0$ et $y=0$.

Pour tourner sur place, il suffit de se trouver au point d'arrêt et vriller le joystick vers le côté où vous désirez faire tourner le robot.

Nous avons également prévu un frein d'urgence grâce à la gâchette présente sous le joystick. Lorsque vous appuyez sur ce bouton, les moteurs sont bloqués jusqu'à ce que vous remettiez le joystick au point d'arrêt.

D'autres boutons ont été configurés pour exécuter certaines actions, à savoir le bouton 9 qui permet le reset des capteurs et le bouton 10 qui permet de remonter les données des capteurs. Enfin, nous avons décidé d'utiliser le potentiomètre du joystick pour paramétrier la vitesse du robot. À l'initialisation, celle-ci est de 100%. Il suffit ensuite de bouger le curseur pour faire varier la vitesse du robot.

Remarque : Le programme Python réalisant le pilotage de Pekee se trouve dans le dossier « Python → Pilotage Pekee » du github.

Nous avons choisi de ne pas mettre d'extrait de code dans ce rapport car ils sont joints à ce rapport (via le Github) et nous avons essayé de commenter au maximum ces programmes permettant une reprise de projet plus simple.

Remarque sur le Raspberry : Concernant les configurations du Raspberry, le TP de M. MONNET mis dans les documentations → Raspberry, explique pas à pas comment il faut procéder. Par ailleurs, nous vous conseillons de sauvegarder régulièrement l'image de votre carte SD. En effet, nous avons cassé notre carte SD et nous n'avions pas fait de sauvegarde de celle-ci. Nous avons donc tout perdu. Afin d'éviter cela, nous vous suggérons d'utiliser régulièrement git depuis le Raspberry.

En conclusion de cette partie, nous avons pu établir une communication série entre l'Arduino et le Raspberry qui permet aux deux cartes de s'échanger diverses données : commande de pilotage, récupération des données des capteurs... Actuellement le robot est pilotable à l'aide d'un joystick branché directement sur le Raspberry mais à terme celui-ci sera déporté pour avoir un robot complètement autonome.

Nous allons voir dans la prochaine partie les pistes d'évolutions pour permettre la poursuite du projet.

VI. Pistes d'amélioration

Si nous reprenons notre objectif initial, nous pouvons voir que nous l'avons presque atteint. Néanmoins il reste quelques améliorations possibles et qui sont nécessaires pour obtenir un robot complètement autonome. Cette partie vous présente donc les pistes d'évolution envisageables mais celles-ci ne sont pas exhaustives.

Les pistes d'amélioration peuvent se décomposer en 2 parties :

1) Partie électronique

La première piste d'amélioration est l'intégration d'une batterie pour les moteurs. Nous conseillons de prendre une batterie dont la tension est comprise entre 12 et 24V mais avec une grande capacité d'ampérage. Nous avons pu voir que les moteurs consomment jusqu'à 1A au démarrage puis ensuite fluctuent autour de 500mA en fonctionnement normal. Nous conseillons donc une capacité comprise entre 8mAh à 12mAh garantissant une plus longue durée de fonctionnement.

Concernant l'alimentation du système, il pourrait être intéressant d'avoir un suivi de la décharge des batteries afin de savoir quand les recharger. Des petits modules simples à brancher peuvent être trouvé pour faire un suivi de tension.



Figure 51 : LCD Testeur de capacité de batterie

Nous suggérons également d'alimenter l'Arduino via son port d'alimentation. Pour cela, lorsque vous aurez la batterie des moteurs, il suffira de convertir la tension vers du 9V (tension optimale pour faire fonctionner l'Arduino depuis le port d'alimentation) avec un régulateur de tension. Cela permettra de préserver le Raspberry car actuellement nous alimentons l'Arduino depuis la liaison série avec le Raspberry.



Figure 52 : Les ports de l'Arduino

Concernant les cartes électroniques, une erreur de conception sur la carte des moteurs a été faite (comme expliqué dans la partie concernée). Nous suggérons donc de la refaire et surtout de faire le rappel des pins de la nappe pour vous permettre de remonter les informations de nouveaux capteurs.

En abordant le thème des capteurs, plusieurs pistes d'améliorations sont envisageables. La première étant l'intégration des capteurs infrarouge sur le robot Pekee d'un point de vue matériel et logiciel. Nous suggérons de réaliser une ceinture de capteurs autour du robot (en les fixant avec de la colle chaude et en faisant passer les câbles au travers des trous déjà prévu à cet effet).

D'un point de vue logiciel, nous avons programmé un petit programme vous permettant de récupérer la distance à laquelle l'obstacle a été détecté mais nous ne l'avons pas intégré à notre programme de pilotage car nous souhaitons vous laissez décider de l'action à réaliser en cas de détection.

La seconde piste est l'exploitation du capteur MinIMU. Celui-ci est déjà intégré au robot et les données peuvent déjà être récupérées par l'Arduino. Néanmoins nous avons récupéré un programme (fourni dans le Git) pour pouvoir l'exploiter mais comme nous avons pu le dire dans la partie test des capteurs, nous n'avons pas réussi à correctement l'exploiter mais nous garantissons son fonctionnement. Il apparaît nécessaire de l'étalonner mais cette étape n'a pas été réalisée faute de temps.

D'un point de vue électronique nous ne voyons pas d'autres pistes d'amélioration.

2) Partie logicielle

Dans cette partie, de nombreuses améliorations sont possibles et vous n'avez quasiment aucune limite, à part celle de votre imagination et du matériel que vous avez à disposition. Nous suggérons donc quelques évolutions envisageables qui sont les plus importantes.

La priorité est de réussir à asservir le robot pour qu'il puisse suivre une trajectoire en ligne droite, tourner du bon angle... Pour cela, il est nécessaire de prendre en compte les données mesurées par le capteur MinIMU ainsi que les encodeurs des moteurs. Il faut également intégrer le programme des capteurs infrarouges.

En parallèle, une autre évolution serait d'avoir le joystick déporté sur un ordinateur depuis lequel vous pourriez piloter le robot et acquérir les données des capteurs en temps réel. Pour ce faire, il faudra développer une petite application Web permettant de piloter le robot et qui afficheraient les données reçues sur un graphique par exemple.

Une fois ces 2 améliorations réalisées, vous pourriez envisager de développer une application Android/IOS ainsi qu'une application sur Windows (comme celle développée par Wany) qui permettrait de rentrer les consignes des moteurs, visualiser l'environnement de Pekee grâce au capteur infrarouge, donner des trajectoires comme avancer sur une certaine distance, tourner d'un certain angle...

À la fin, une fois que le robot sera complètement autonome, vous pourriez envisager une interaction avec les autres robots de Robotique de service.

En conclusion, de nombreuses améliorations sont envisageables et pour résumer les plus importantes, voici une liste des tâches classée du plus important au moins important :

- Intégration d'une batterie pour les moteurs
- Refaire la carte électronique des capteurs pour faire un rappel des pins de la nappe
- Évaluation du capteur MinIMU et intégration des capteurs infrarouges
- Prévoir un monitoring pour le niveau de charge des batteries
- Déporter le joystick pour avoir un robot complètement autonome
- Développer une application HTML pour accéder aux données des capteurs
- Développer une application Windows (par exemple comme celle de Wany Environnement)
- Développer une application Android/IOS
- Intégrer le robot avec les robots de l'APP Robotique de service.

VII. Conclusion

L'objectif initial de notre projet était de livrer un robot autonome que l'on peut piloter à l'aide de fonctions intuitives et simples.

Nous avons dans un premier temps repris le travail de nos anciens camarades afin de prendre en main le robot et définir nos tâches de travail. Suite à leur conclusion, nous avons décidé de nous focaliser sur l'exploitation du capteur MinIMU et sur l'asservissement du robot pour permettre un contrôle précis de ces trajectoires.

Malheureusement nous avons perdu beaucoup de temps sur cette partie pour au final ne pas réussir à exploiter ce capteur. Nous avons alors changé de direction et de point de vue – grâce au conseil de M VERNIER – en prenant le problème à l'envers. Cela consistait à intégrer directement le capteur dans le robot et depuis des trajectoires connues, évaluer le capteur.

Toutefois, pour ce faire, il nous a fallu concevoir une nouvelle architecture du robot.

Ce travail a constitué la majeure partie de notre temps de projet car il nous a fallu réfléchir à l'agencement des composants, concevoir les cartes électroniques, les tester et assembler l'ensemble du robot. Il nous a également fallu réfléchir comment gérer l'alimentation du système. Nous avons développé un circuit électronique complet qui permet d'avoir une bonne gestion de l'alimentation de Pekee.

La vision que nous avons eue pour cette conception était de déporter les cartes électroniques au plus près des éléments concernés. Le robot Pekee ressemble à un « mille-feuille » mais le gros avantage à cela est que si vous avez besoin de changer un élément défectueux, vous avez simplement besoin de l'enlever et de le remplacer sans affecter le reste du système. Cette vision est peut-être futuriste mais cela garantit une maintenance plus simple, plus efficace et plus rapide sur le robot.

Une fois que la conception électronique du robot a été faite, nous avons pu programmer le robot – et plus particulièrement l'Arduino et le Raspberry – pour permettre un pilotage simple, intuitif et fluide. Celui-ci se fait actuellement à partir d'un joystick mais à terme cela pourra être fait depuis une interface Web par exemple.

Le point fort de notre projet est la portabilité du robot. En effet, vous pouvez le programmer à distance. Grâce à la liaison série entre l'Arduino et le Raspberry vous pouvez avoir accès à l'ensemble des données des capteurs remontées au niveau de l'Arduino.

Le projet n'est donc pas encore finalisé, c'est pourquoi nous avons pu suggérer quelques améliorations possibles pour la reprise du projet comme l'intégration d'une batterie pour les moteurs (son intégration est déjà prévue sur le système, il n'y a plus qu'à brancher la borne positive et la borne négative sur les pins prévus à cet effet sur la carte électronique de la gestion d'alimentation). L'étude du capteur MinIMU est un point sur lequel nous sommes un peu frustrés car nous n'avons pas réussi à l'exploiter. Nous espérons néanmoins que les futurs étudiants réussiront à l'exploiter pour réaliser l'asservissement du robot.

En conclusion, ce projet s'est révélé être un des plus intéressants de notre formation (en plus des APP). Il nous a permis de mettre à l'épreuve nos compétences acquises lors de celle-ci, et ce, sur tous les plans : instrumentation avec l'intégration et l'exploitation de capteurs, automatisation avec l'asservissement du robot et enfin de l'informatique à la fois embarquée avec l'Arduino et plus standard avec le Raspberry Pi. Nous avons pu échanger avec beaucoup de personnes sur les différents aspects de notre projet afin d'obtenir un ensemble cohérent et fonctionnel. Nous remercions encore une fois toutes les personnes qui nous ont accordé de leur temps et sans qui ce projet n'aurait pu être réalisé.

Nous souhaitons également bon courage à nos successeurs pour la reprise du projet et espérons qu'un jour Pekee rejoindra les autres robots d'enseignement de Polytech.

Annexe 1 : Webographie et sources du rapport

Cette annexe conclue le rapport et vous présente les différentes sources utilisées pour le rédiger.

➤ Sources des images

Figures 2 & 6 : Carte enfichable & Carte mère de PekeeI

→ <http://www.wanyrobotics.com/pekeeI.html>

Figure 4 : Pekee en train de réaliser un chemin de ronde

→ <http://www.cnrs.fr/cw/dossiers/dosrob/legendes/wany01.html>

Figure 3 & 5 : Vue éclatée de PekeeI & Logiciel de contrôle

→ http://coursups.free.fr/L1/TP_Navigation_Robot_Mobile.pdf

Figure 8 : Schéma synoptique : Ordinateur / Portable / Tablette / Joystick / Page Web / Logo Android / Logo AppleStore / Robots Nao & Pepper / Robotino

→ <http://www.icone-png.com/informatique/ordinateur/2.php>

→ <https://goo.gl/images/MN1jE5>

→ <https://www.nextinpack.com/archive/69480-google-tablette-199-fabriquee-asus.html>

→ <https://www.freepngs.com/joystick-pngs>

→ <https://openclassrooms.com/courses/apprenez-a-creer-votre-site-web-avec-html5-et-css3/votre-premiere-page-web-en-html>

→ <http://michel-eynard.fr/Android.html>

→ http://iaddict.com/apple/251_lapp-store-sameliore-en-orthographe-vraiment

→ <https://goo.gl/images/RZpB6b>

→ <https://trac.fawkesrobotics.org/wiki/Robotino>

Figure 9 : Raspberry

→ <https://m.reichelt.com/fr/fr/RASPBERRY-PI-3/3/index.html?ACTION=3&LA=0&ARTICLE=164977&artnr=RASPBERRY+PI+3>

Figure 10 : Caractéristiques du Raspberry

→ <http://branchez-vous.com/2016/02/29/le-raspberry-pi-3-passage-64-bits-pour-le-pc-miniature-35-us/>

Figure 11 : Arduino MEGA

→ <http://shyrobotics.com/2013/02/10/tutoriel-2-comment-programmer-une-carte-arduino/>

Figure 12 : Caractéristiques de l'Arduino MEGA

→ <http://www.redohm.fr/2014/12/arduino/>

Figure 13 : Pont en H

→ <http://www.trossenrobotics.com/store/p/6508-L298-Dual-H-Bridge-Motor-Driver.aspx>

Figure 14 : Caractéristiques du pont en H

→ <https://www.robotshop.com/en/seedstudio-l298-dual-h-bridge-motor-driver.html>

Figure 15 : Module de Relais & Boutons

→ <https://fr.aliexpress.com/item/Free-shipping-4-channel-relay-module-4-channel-relay-control-board-with-optocoupler-Relay-Output-4/32702042620.html?spm=a2g0s.9042311.0.0.BEZ5KO>

→ <http://www.overgratt.com/achat/contacteur-selecteur-micros-guitare-micro-switch.html>

Figure 18 : Roue folle à bille

→ https://www.castorama.fr/roulette-fixe-a-platine-fixe-o15-8-mm/3700001719044_CAFR.prd

Figure 19 : Joystick

→ <https://www.amazon.fr/Logitech-Extreme-Pro-Joystick-Noir/dp/B00CJ5FPTA>

Figure 20 : Capteurs IMU & Infrarouge

→ <https://www.pololu.com/product/2468>

→ <https://www.robotshop.com/ca/fr/capteur-distance-ir-sharp-gp2yoa21-10cm-80cm.html>

Figure 22 : Logos Trello & Github

→ https://fr.wikipedia.org/wiki/Fichier:Logo_Trello.png

→ <https://github.com/logos>

Figure 51 : Écran LCD testeur de batterie

→ https://www.amazon.com/DROK-DC8-63V-Capacity-Indicator-Protective/dp/B01MUWG96H/ref=sr_1_4?s=hi&ie=UTF8&qid=1498637896&sr=1-4&keywords=LCD-Battery-Capacity-Tester-Indicator-12V

→ <https://www.banggood.com/fr/Battery-Capacity-Tester-with-LCD-Indicator-for-12V-24V-36V-48V-Lead-acid-Lithium-LiPo-p-991337.html?rmmds=search>

Les autres *Figures* présentes dans le rapport sont des photos personnelles que nous avons réalisées.

Enfin les logos se trouvant sur la page d'accueil et en pied de page sont issues de l'université suivante :

- Université Savoie Mont Blanc : <https://www.univ-smb.fr/>
- Polytech Annecy-Chambéry : <http://blog.polytech.univ-savoie.fr/>
<http://www.polytech.univ-smb.fr/>

➤ Sources du rapport

Concernant les éventuelles sources du rapport nous les avons directement indiquées dans le rapport au moment convenu mais voici les principales sources du rapport :

- Document sur le robot Pekee ancienne génération

→ <http://www.wanyrobotics.com/pekeeI.html>

→ http://coursups.free.fr/L1/TP_Navigation_Robot_Mobile.pdf

- Documentation sur le Raspberry

→ <https://www.framboise314.fr/raspberry-pi-3-quoi-de-neuf-docteur/>

→ <http://raspi.tv/2013/how-to-use-interrupts-with-python-on-the-raspberry-pi-and-rpi-gpio>

→ <https://raspbian-france.fr/>

- Documentation sur l'Arduino

→ <http://www.redohm.fr/2014/12/arduino/>

→ <https://www.arduino.cc/>

→ <https://openclassrooms.com/courses/programmez-vos-premiers-montages-avec-arduino/le-moteur-a-courant-continu-partie-2-le-pont-en-h-et-les-circuits-integres>

→ <https://zestedesavoir.com/tutoriels/686/arduino-premiers-pas-en-informatique-embarquee/>

- Documentation sur le Joystick

→ <http://home.wlu.edu/~levys/software/pyquadstick/>

→ <https://www.pygame.org/news>

- Documentation sur le capteur MinIMU et IR

→ <https://www.pololu.com/product/2738>

→ <https://www.pololu.com/product/1136>

Le GITHUB de Pekee 2.0 : <https://github.com/chapellu/pekee2.0.git>