

# Projet Robot Wi-Fi

## Rapport

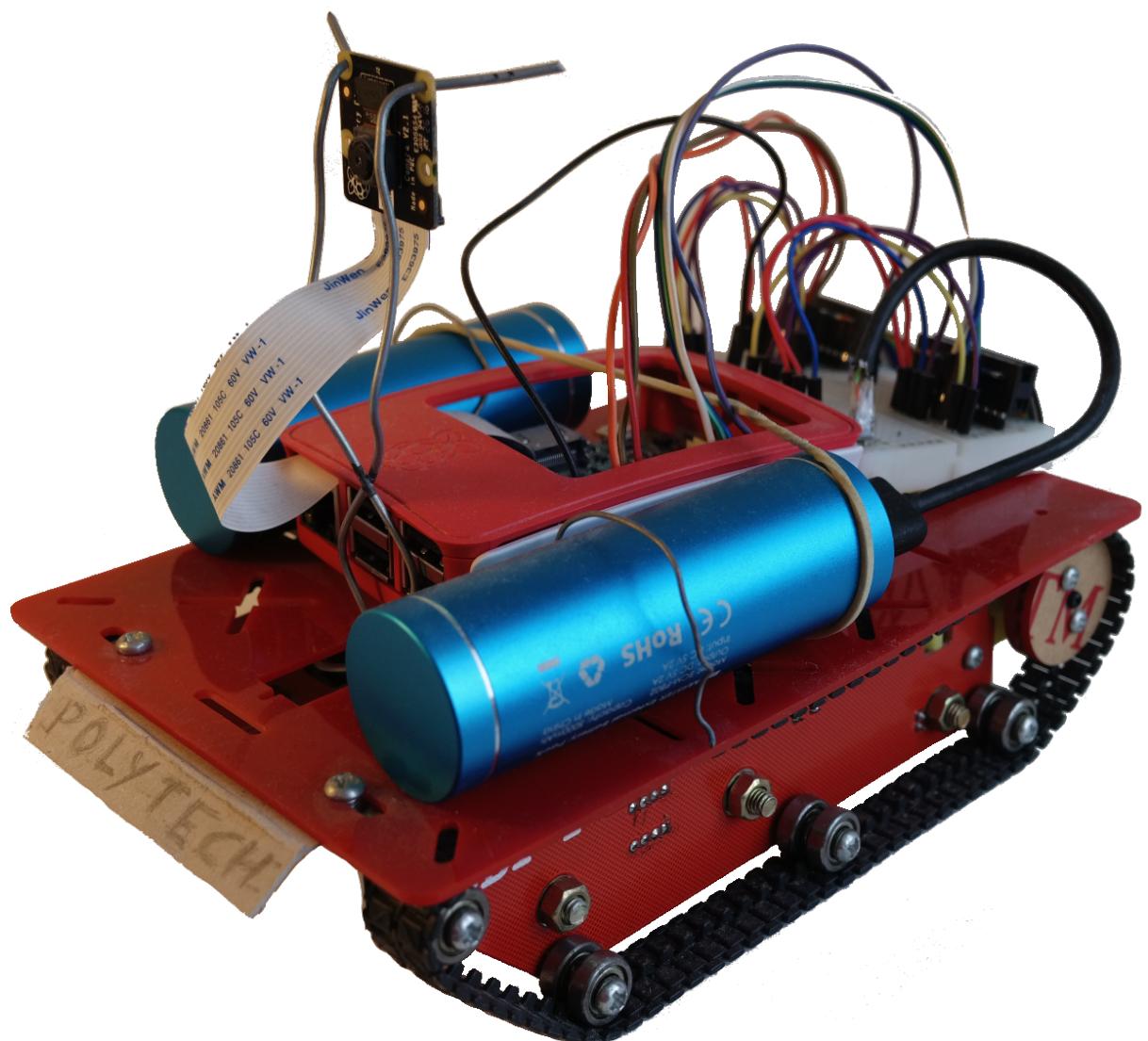
Jean-Marc Abbas

Olivier Léobal

Charles Leroux

Coordinateur : François Aguillon

Avril 2017



# Sommaire

<b>1 Projet initial et évolution</b>	<b>4</b>
<b>2 Gestion</b>	<b>5</b>
2.1 Découpage en deux semestres . . . . .	5
2.2 Organisation & Communication . . . . .	7
<b>3 Réalisation</b>	<b>8</b>
3.1 Construction . . . . .	9
3.2 Electronique . . . . .	9
3.2.1 Moteurs . . . . .	9
3.2.2 Batteries . . . . .	9
3.3 Informatique . . . . .	10
3.3.1 Materiel . . . . .	10
3.3.2 Logiciel . . . . .	10
<b>4 Produit fini</b>	<b>12</b>
4.1 Comparaison à l'existant . . . . .	12
4.2 Utilisations envisageables . . . . .	14
<b>5 Suite du projet</b>	<b>15</b>
5.1 Possibilités de continuation . . . . .	15
5.2 Améliorations possibles . . . . .	15
5.2.1 Prélude . . . . .	15
5.2.2 Projets d'amélioration . . . . .	16
<b>Annexe technique</b>	<b>18</b>
A.1 Informatique . . . . .	18
A.1.1 Solution logicielle . . . . .	18
A.1.2 Puissance de calcul nécessaire . . . . .	18
A.1.3 Latence . . . . .	18
A.2 Électronique . . . . .	19
A.2.1 Châssis . . . . .	19
A.2.2 Batteries . . . . .	19

# Introduction

Ce rapport a pour objet la création d'un robot chenillé télécommandé par interface web. Il s'agit d'une étude de faisabilité, où on présentera les principes directeurs et les principaux problèmes, afin de profiter à ceux qui pourraient reprendre notre travail. La question est : est-il possible de télécommander un robot terrestre via une interface web avec flux vidéo en direct ? Cette étude a été conduite à travers la réalisation d'un prototype.

Le but de ce projet est avant tout pédagogique ; il ne comporte pas par exemple d'étude pour une possible commercialisation. Il reste particulier par sa grande autonomie, mais il s'agit d'une force : nous avons pu agir comme le ferait une équipe dans une petite ou moyenne entreprise, nous intéressant à plusieurs domaines de compétences plus ou moins proches de nos spécialités.

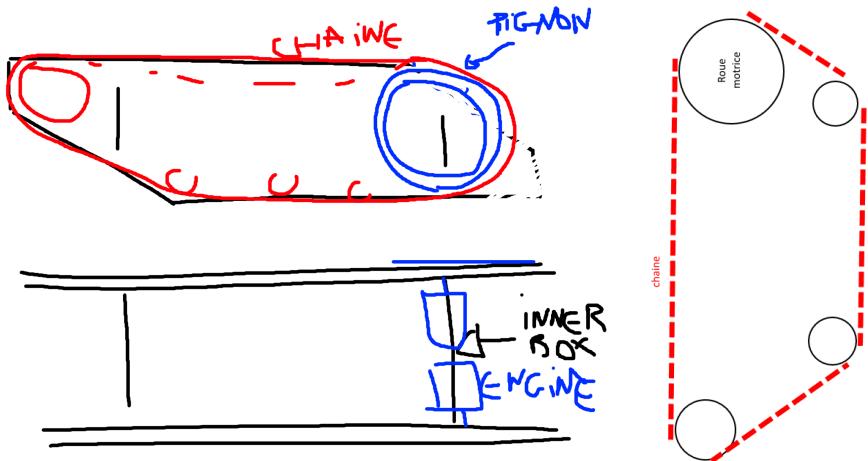
## Remerciements

Nous voudrions remercier MM. Aguillon et Bernez pour leur aide au long du projet, et Mme Guimard pour avoir encouragé les élèves à proposer leurs propres projets. J'en (Olivier) suis particulièrement reconnaissant, parce que ce projet a été crucial dans l'obtention de mon stage de fin d'année.

Nous voudrions aussi remercier M. Dominic Szablewski (nous utilisons son lecteur `JSmpeg`), les développeurs du serveur `Web Tornado` (et aussi `Facebook`, qui l'a rendu public), les contributeurs du projet `FFmpeg`, les contributeurs du projet `Debian`, ainsi que tous ceux dont le travail a permis aux personnes déjà mentionnées de nous aider à leur tour.

# 1 Projet initial et évolution

Le projet initial (qui a été approuvé par l'école) prévoyait de construire un robot chenillé de  $40 \times 20 \times 20$  cm, ayant un châssis et une motorisation construites pour. Il prévoyait également un pilotage par interface web (via un lien sans fil wi-fi), un flux vidéo en direct, et une caméra sur tourelle mobile. Il devait être construit en quatre mois, sans aide financière.



**Figure 1 :** Des dessins de fonctionnement tels que prévus à l'origine. On peut noter que la roue motrice ici est la plus grande, ce qui n'est pas idéal pour un véhicule chenillé (très inefficace pour augmenter le couple, mais augmente la vitesse).

Après discussion avec notre tuteur (M. Aguillon), les objectifs ont été revus à la baisse. Le prototype utilise maintenant un ensemble châssis + moteurs + chenilles, bien plus petit, et sans tourelle. Les objectifs informatiques (flux vidéo et commande web) ont été conservés, malgré la perte d'un informaticien sur les trois du départ.

Il faut signaler que là où l'électronique utilisée est éprouvée, la partie informatique est particulièrement intéressante parce qu'elle fait usage de technologies qui font à peine leur entrée dans cet environnement. En particulier, le flux vidéo en direct a été un défi du développement web pour une décennie, surtout sans programmes annexes (plug-ins) et avec une latence (décalage) minimale.



**Figure 2 :** Un Mark IV Britannique de 1917 (gauche) et une mine télécommandée Goliath Allemande de 1942 (droite). Cette forme permet de passer au mieux les obstacles.

## 2 Gestión

### 2.1 Découpage en deux semestres

Le découpage en deux semestres (un de planification, un de réalisation) a un but pédagogique évident. Cela dit, nous nous intéressons avant tout à son utilisation dans le projet.

On nous a suggéré plusieurs outils, que nous avons tenté de nous approprier et avons tenu à jour autant que possible.

**Plan Qualité Projet** Il nous semble que le PQP serait particulièrement utile pour présenter le projet à de nouveaux collaborateurs et s'accorder sur la mise en œuvre. Certains composants ont reçu davantage d'attention que d'autres, mais nous n'en avons globalement pas fait grand usage. Comme il faut le tenir à jour, on peut supposer qu'une petite équipe technique comme nous pourrait se passer d'une grande partie de la documentation (non technique) pour ne la rédiger que quand elle deviendrait utile.

**Analyse des risques** L'outil nous a paru essentiel même sur de petits projets (qui ont une chance d'échec non négligeable), pour évaluer et allouer les ressources. Nous n'avions que peu de ressources à risquer, et dans le cadre de notre étude les risques sont ce qui nous empêche de mener à bien l'étude plutôt que ce qui mène à un échec du prototype (l'étude n'en étant pas moins enrichie). En pratique, le seul risque de cette nature pour nous a été l'achat d'un châssis chinois et ses délais de livraison flous ; nous n'en recommandons pas moins à tous les groupes de travailler sur une analyse des risques.

**Diagramme de Gantt** Nous pensons tous que le diagramme permet une vue d'ensemble des tâches et des ressources, mais c'est au prix d'un travail important et sur la durée (il faut le tenir à jour). Nos tâches étaient relativement linéaires et monolithiques (*fig. 3*), sans compter que nous n'avions pas beaucoup de ressources à répartir (la division du travail n'était pas un problème). Une partie du travail était d'explorer différentes technologies : le diagramme était utile pour visualiser le temps restant, mais l'évaluation du temps nécessaire proprement dite était impossible.

**Compte-rendu de séance** Nous avons établi un compte-rendu à chaque réunion. Nous aurions pu nous en passer, mais ce n'est que parce que nous sommes en contact via des méthodes ayant une mémoire des anciens messages, et même pour nous il était très utile de faire le point des avancées de chaque semaine de manière claire et facile à accéder. Les compte-rendus ne requièrent que peu d'efforts (et par leur nature n'ont pas besoin d'être tenus à jour), nous les tenons donc comme très bénéfiques.

En considérant tous ces outils, il faut se rappeler que nous étions en contact permanent et que nous ne communiquions presque pas avec l'extérieur. Beaucoup d'outils de coordination et de standardisation peuvent alors devenir superflus. De plus (et c'est plus spécifique à notre groupe), l'objet du projet était très mal défini au commencement ; dès lors, tenir à jour les outils de gestion devenait une tâche plus lourde qu'elle ne l'aurait normalement été, parce que nous changions des aspects importants du projet presque chaque semaine à cette période.

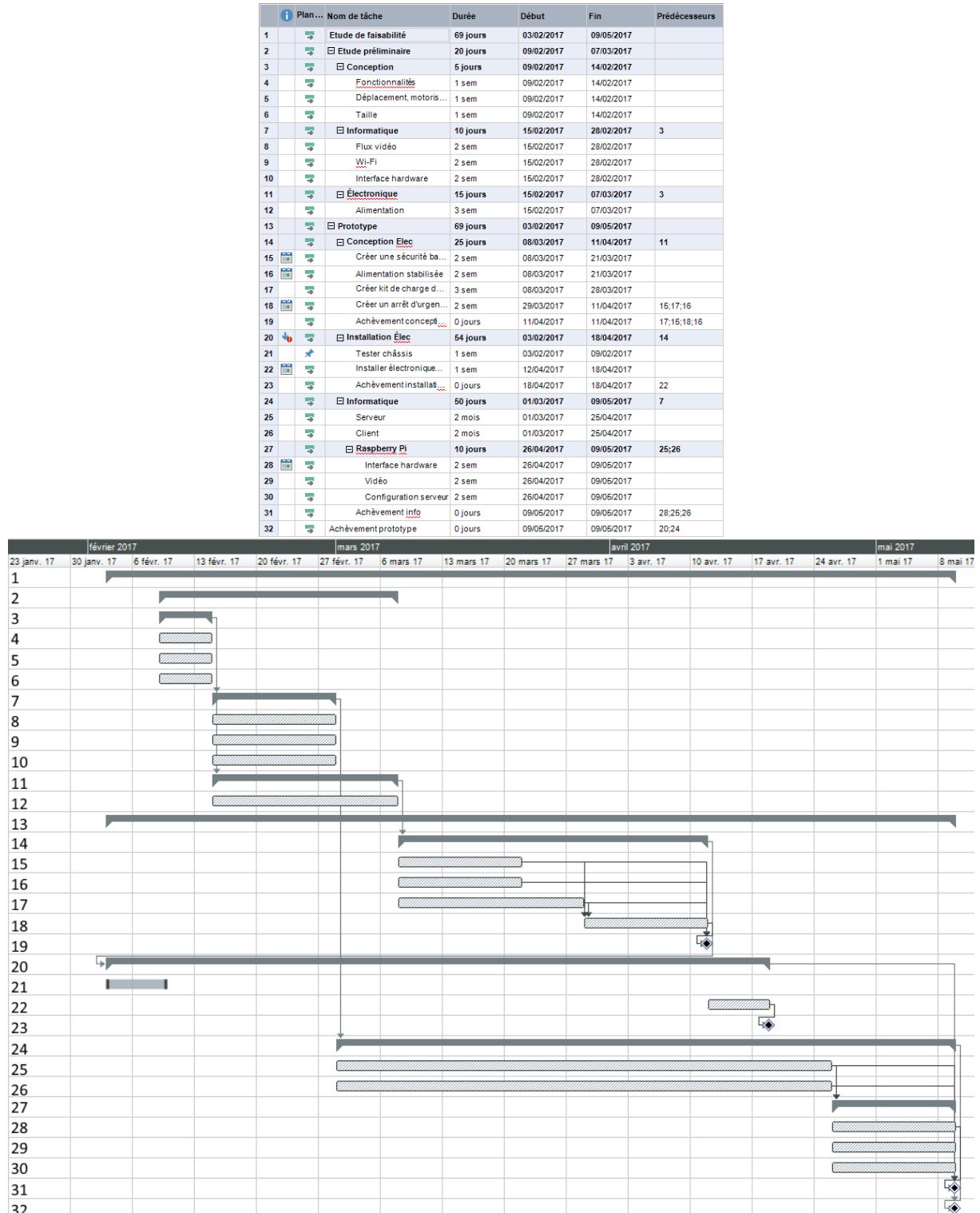


Figure 3 : Notre diagramme de Gantt prévisionnel à la fin du premier semestre

## 2.2 Organisation & Communication

Certains outils mis en place durant la phase de réalisation nous ont été particulièrement utiles et nous les recommandons aux groupes qui pourraient venir après nous.

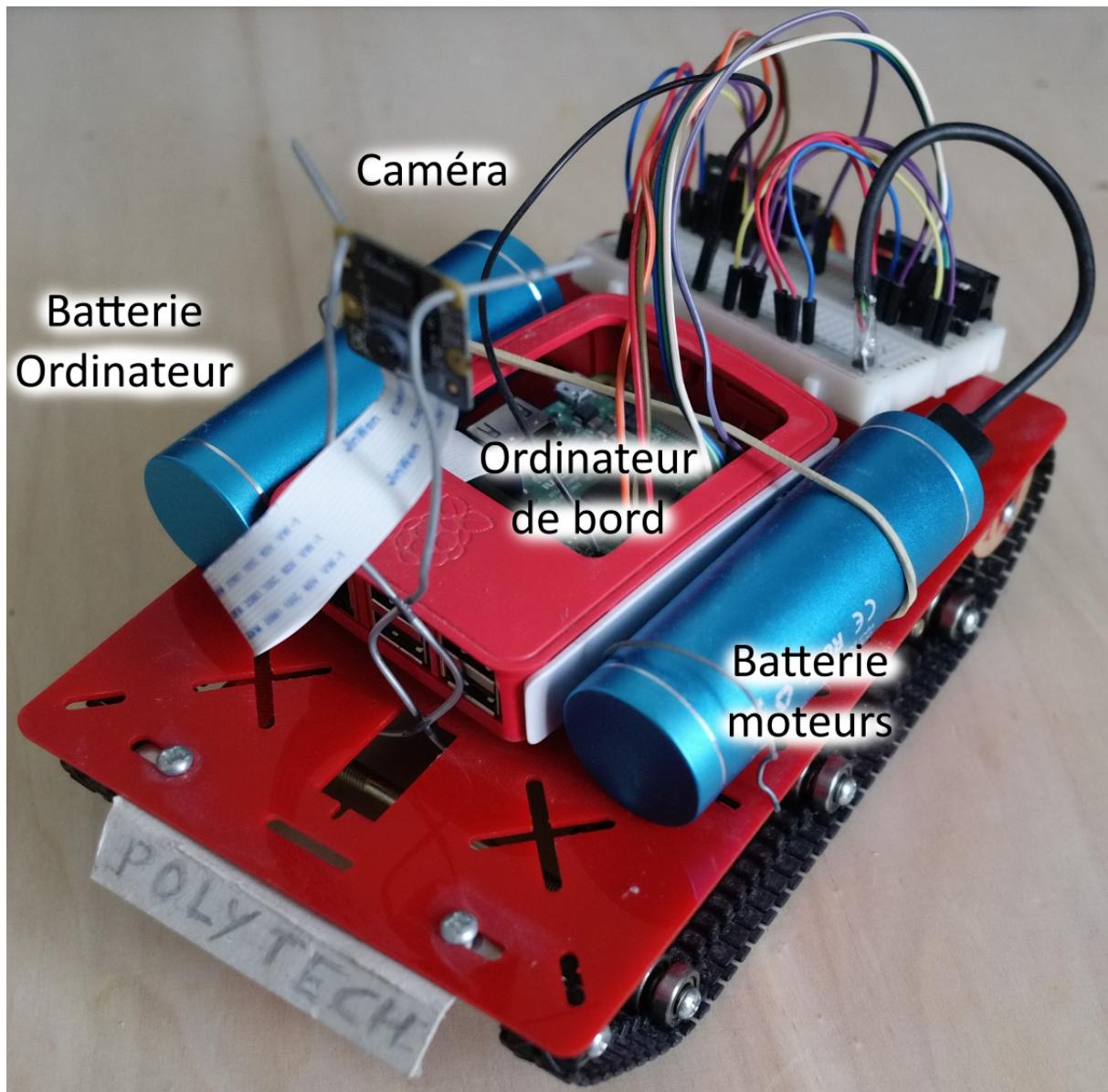
**Réunions régulières** Comme dit plus haut, malgré notre contact quasi-permanent, nous avons trouvé très utile de faire le point toutes les semaines et de consigner le résultat dans une archive facilement accessible. Bien sur, une salle de TP de l'école est également bien plus conductrice au travail de l'électronicien, alors que l'informaticien est plus flexible.

**Partage des compétences** Nous n'avions qu'un seul électronicien, mais il a profité des réunions en salle de TP pour expliquer les décisions prises et a même pu recevoir un peu d'aide, les informaticiens ayant tous les deux ayant reçu une formation de base en électronique au lycée. En dehors des grandes structures qui peuvent placer des spécialistes sur un système très précis, beaucoup de projets tirent parti de plusieurs compétences, et il est utile que les membres de l'équipe aient des notions pour pouvoir se comprendre.

**Tuteur & professeurs** Ce n'est pas un outil à proprement parler, mais M. Aguillon nous a dit s'être mis à notre disposition ; il nous a beaucoup facilité la tâche, nous a aiguillé vers la bonne direction et nous a fourni un point de vue extérieur, plus large et plus expérimenté ; il a par exemple suggéré des bornes au projet qui se sont avérés être d'une dimension presque parfaite au cadre donné par l'école. M. Bernez nous a aidé à obtenir du matériel prêté par l'école (nous n'étions pas au courant que c'était possible). Nous recommandons de ne pas oublier de communiquer avec les professeurs.

### 3 Réalisation

Nous allons ici évoquer les aspects les plus importants de la réalisation du prototype, qui était l'axe principal de notre étude.



*Figure 4 : Notre prototype*

## 3.1 Construction

Le robot est construit sur un châssis «Tchang DD1-1» (34€ sur *eBay*). Il s'agit d'une plate-forme avec deux moteurs liés à deux chenilles, et deux petits circuits de contrôle permettant de s'abstraire d'une partie de l'électronique.

On utilise deux batteries de téléphone identiques du commerce pour alimenter notre robot : elles ont la particularité de fournir un courant d'intensité plus élevée que la plupart des batteries (2,5 ampères, contre 1 en général) ; sinon, notre ordinateur de bord et notre électronique ne pourraient pas fonctionner à pleine capacité. Une batterie est dédiée à l'ordinateur, l'autre aux moteurs.

On a fait le choix d'avoir un ordinateur embarqué qui se chargera de tout ce qui est à faire, pour ne pas avoir besoin d'autre appareil que le robot. La taille du châssis pose des contraintes importantes sur la taille, mais il est heureusement facile de trouver des ordinateurs miniatures ("SoC", *System on a Chip*) de la bonne taille.

## 3.2 Electronique

### 3.2.1 Moteurs

Les moteurs que nous avons utilisés étaient directement intégrés au châssis. Ils étaient déjà équipés de réducteurs de vitesse et ils avaient une tension de fonctionnement allant de 3V à 8V. Un défaut d'assemblage empêchait la rotation d'un des moteurs : le moteur gauche était désaxé et se bloquait contre le capteur de vitesse du châssis. Nous avons en premier lieu cru à un dysfonctionnement du moteur, puis nous avons remonté le moteur une fois le problème trouvé. Nous avons fait des test de consommation et de vitesse à vide afin de connaître les caractéristiques des deux moteurs ; nous n'avons pas eu besoin du capteur de vitesse du châssis pour le pilotage, les moteurs ayant une caractéristique linéaire.

### 3.2.2 Batteries

Nous avons choisi d'utiliser des batteries de 5V afin de simplifier plusieurs points du projet. Le 5V est la tension standard du câble USB. Nous avions ainsi un système de recharge simplifié, soit depuis le secteur avec un adaptateur, soit depuis un ordinateur. L'autre avantage est que l'ordinateur de bord que nous avons utilisé s'alimentait en USB aussi : nous pouvions ainsi directement connecter la batterie dessus. L'autonomie des batteries était plus que suffisante par rapport à nos objectifs (nous voulions un minimum d'une dizaine de minutes d'utilisation). Les batteries, avec une capacité de 5 000 mAh, pouvaient être utilisées une demi-journée en fonctionnement avant d'être rechargés avec la consommation finale du prototype d'environ 1 ampère par heure par batterie.

Afin de faire varier la vitesse des moteurs, il faut faire varier leur tension d'alimentation. Nous avons donc réalisé un hacheur afin de moduler les 5V fournis par les batteries. Le hacheur est piloté par les broches de l'ordinateur de bord, à l'aide d'une PWM logicielle. (Voir l'annexe technique en A.2.1)

### 3.3 Informatique

#### 3.3.1 Materiel

Le Raspberry Pi 3, dont la taille nous convient, est choisi pour trois raisons :

- Il est peu coûteux (35€ seul) ;
- Il est très répandu, ce qui garantit la présence d'accessoires, de programmes fonctionnant avec, et d'assistance en ligne si besoin est ;
- Par rapport au Raspberry Pi 2 (son prédecesseur), il est plus puissant et dispose d'un modem Wi-Fi ; justement, on a besoin de puissance pour traiter notre flux vidéo, et de Wi-Fi pour le transmettre.

En outre, il existe une caméra officielle pour, ce qui nous garantit un fonctionnement correct. C'est de celle-ci qu'est équipé le prototype.

#### 3.3.2 Logiciel

Nous cherchons à envoyer de la vidéo et recevoir des commandes ; la contrainte principale ici est la latence (c'est à dire le décalage entre émission et réception).

On fait le choix de tout gérer depuis l'ordinateur embarqué du robot lui-même : il créera lui-même son réseau Wi-Fi, filmera et communiquera tout lui-même, et ainsi de suite. Le robot sera donc tout à fait autonome.

**Envoyer des commandes** : le côté client (c'est à dire la machine de l'utilisateur) est à la fois l'endroit le plus intéressant et celui où on a le moins de choix. Comme on ne veut rien installer sur la machine du client (pas de plug-ins/extensions), on est forcés d'utiliser ce qui est présent sur le navigateur de l'utilisateur, c'est à dire le langage *Javascript* et toutes les extensions qu'il a reçu au fil du temps. Pour envoyer des commandes, on utilise la nouvelle technologie *WebSockets* : en informatique réseau, on se repose toujours sur des "couches" logicielles inférieures, et cette technologie nous permet d'en court-circuiter plusieurs, espérant obtenir une meilleure latence.

**Jouer de la vidéo** sur le navigateur n'est pas forcément simple ; c'est un marché en pleine extension, et de nombreux concurrents existent. On a examiné plusieurs technologies (*HLS* d'Apple, *MPEG-Dash..*), mais aucune ne satisfait nos objectifs de latence tout en utilisant des technologies libres. Enfin, une solution se présente : le lecteur *jsmpeg*, qui, comme son nom l'indique, utilise le langage Javascript déjà présent sur le navigateur pour décoder une vidéo au format *MPEG-1*. Ce format ("codec") est si obsolète que les brevets dessus ont expiré, mais nous conviendra.

**Filmer** est possible grâce à plusieurs programmes fournis par le fabricant de la caméra Raspberry Pi. On utilise *picamera*, mais le format de vidéo est différent : on utilise donc (le projet libre) *FFmpeg* pour le transformer en MPEG-1 que notre lecteur saura lire. Cela prend beaucoup de puissance de calcul, et nous perdons en qualité, mais ce n'est qu'une démonstration de principe (*Proof of concept*)

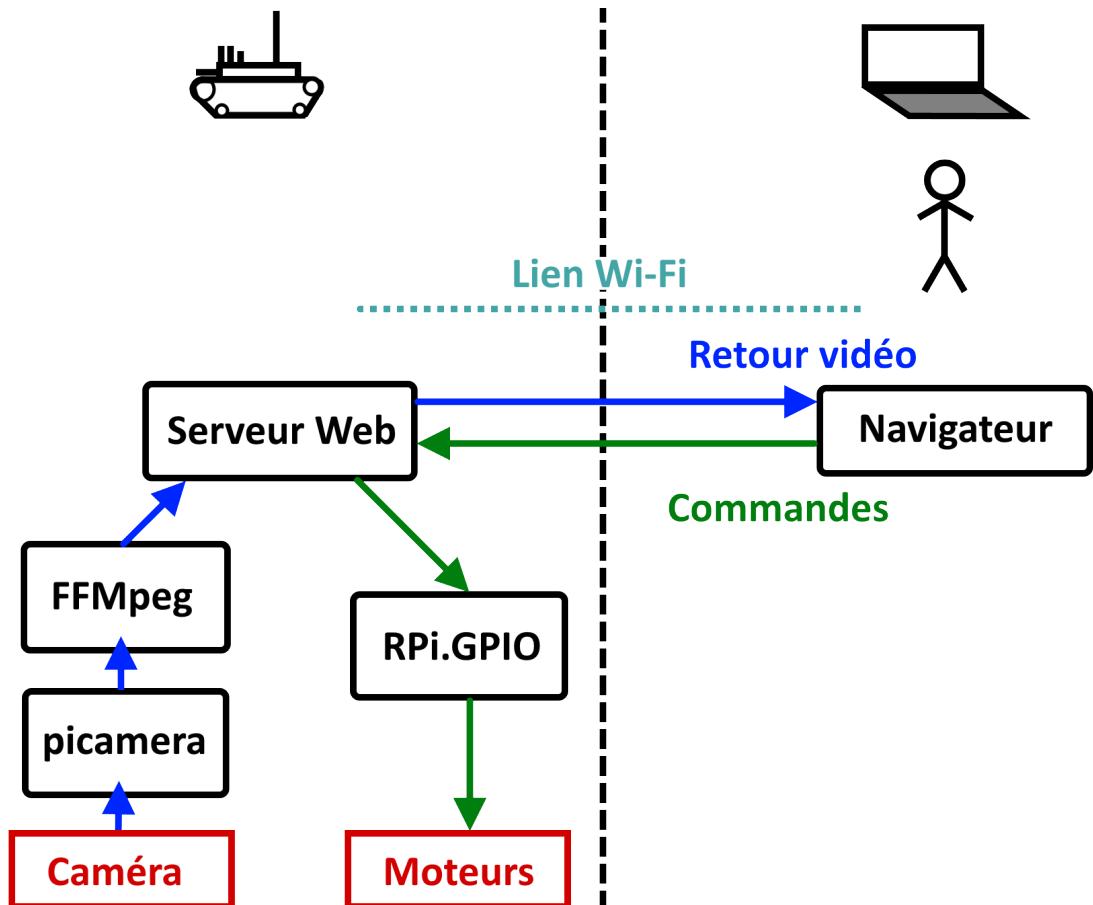


Figure 5 : La "pile" technologique (tech stack) de notre projet.

**Bouger** requiert simplement d'utiliser les broches électroniques (*General Purpose Input/Output*, GPIO pins) du Raspberry Pi, pour lesquelles le fabricant fournit bien sur un programme d'utilisation (appelé *RPi.GPIO*). Certains détails d'électronique (comme la variation des moteurs par PWM, évoquée en 3.2.2) sont gérés ici, étant donné qu'il est plus rapide d'écrire quelques lignes de code que de re-câbler les moteurs.

**Lier ensemble ces composants** est fait par le serveur Web. Comme on veut la meilleure latence possible, on utilise un serveur *non bloquant*, c'est à dire qu'il ne va pas jamais se bloquer en attendant qu'une tâche soit finie : *Tornado*, édité par Facebook. Le serveur est en langage *Python*, ce qui nous permet d'unifier nos outils sous ce langage (il est possible de gérer la caméra en Python également, mais on veut changer le format). On note également que ce serveur est parfait pour communiquer avec plusieurs utilisateurs à la fois.

## 4 Produit fini

### 4.1 Comparaison à l'existant

Il existe aujourd’hui des robots dans le commerce au fonctionnement relativement similaires au notre. Notre robot s’inscrit donc dans une catégorie, mais on peut aussi le comparer à des véhicules télécommandés en général.

Le robot se connecte en Wi-Fi (un signal standard très répandu en informatique grand public), ce qui ne lui donne une portée que d’une quinzaine de mètres directement. Seulement, on peut passer d’un relais et d’un autre (donc éviter de perdre la connexion tant qu’on reste dans la zone couverte), et le réseau Wi-Fi peut être interconnecté avec d’autres réseaux, parmi lesquels Internet. Cela veut dire qu’on peut en principe télécommander notre robot depuis l’autre côté de la planète ; en pratique, nous avons testé de Paris à Orsay avec succès et une latence très acceptable.



**Figure 6 :** Le «WowWee Rovio» Hong-Kongais, vendu 300\$ à son lancement, et son interface de contrôle sur Windows.

Le réseau Wi-Fi est nécessaire parce que le robot a depuis le départ été conçu pour diffuser un flux vidéo en direct, ce qui nécessite une méthode capable de transférer beaucoup de données en peu de temps, ce que n'est pas capable de faire une voiture télécommandée traditionnelle par exemple. Les robots du commerce ont fait le même raisonnement pour les mêmes raisons (mais nous ne le savions pas à l'époque, nous n'avons fait de recherches plus avancées qu'après).

Beaucoup de robots similaires au nôtre (*fig. 6, 7*) nécessitent l'installation d'un logiciel sur ordinateur ou téléphone. Le nôtre dispose d'une couverture bien plus grande étant donné que n'importe quel navigateur ou téléphone récent peut utiliser notre système, sans aucune installation nécessaire.



**Figure 7 :** Le «Varram Appbot Riley» Sud-Coréen, vendu 200€ sur Internet (mais en promotion à 182 000 Wons sur le site du fabricant, soit 135€), nécessite une application pour smartphone.

Grâce à l'utilisation du web et au niveau d'accessibilité dont nous disposons, nous pouvons en principe gérer un très grand nombre de connexions. Notre prototype n'est pas conçu pour cela (et n'a pas la puissance nécessaire), mais rien n'interdit de gérer des dizaines, voire des centaines ou des milliers d'utilisateurs. Cela est beaucoup plus difficile pour d'autres technologies et n'est (pour autant que nous puissions dire) pas une idée explorée commercialement.

## 4.2 Utilisations envisageables

Techniquement, l'interface web et la gestion de la communication permet:

- Une modification aisée grâce à la modularité, flexibilité et ubiquité des technologies web ;
- La possibilité à plusieurs personnes de contrôler le véhicule en même temps ;
- Un contrôle sur grandes distances à travers un réseau local ou même Internet.

De ce fait, et après quelques ajustements, on peut imaginer utiliser le robot à des fins de divertissement ou comme jouet. Plusieurs personnes pourraient par exemple tenter de le contrôler en même temps, de manière chaotique ou coopérative afin d'explorer des endroits, et ce, à distance ou dans la même pièce. Pour pousser le concept plus loin, on pourrait imaginer une intégration à des services sociaux et vidéo en direct, afin de pouvoir les contrôler à distance dans le cadre d'une communauté.

Une autre utilisation serait en tant qu'appareil de surveillance. On pourrait par exemple parcourir des locaux la nuit à distance ou explorer des endroits difficiles d'accès (c'est ainsi que sont vendus beaucoup de robots du commerce). Cependant, les spécifications du véhicule devront être revues pour en faire un engin efficace à de telles tâches.

## 5 Suite du projet

### 5.1 Possibilités de continuation

Il a été suggéré que le projet pourrait devenir un "projet d'école", une sorte de mascotte améliorée d'année en année. Charles, Jean-Marc et moi sommes tout aussi enthousiastes que Mme Henriot à l'idée que le projet soit repris par d'autres. Nous sommes prêts à passer devant une promotion et avons tourné quelques vidéos de démonstration supplémentaires pour préparer une vidéo, au début de l'été.

Nous avons réfléchi à des projets potentiels. Nos critères principaux sont la charge de travail et l'intérêt du projet (nous ne voulons pas distribuer des projets "moins intéressants" que le nôtre). Mais avant tout projet d'amélioration, il faut travailler sur le robot pour le rendre plus fiable ; sinon, impossible de fournir des fondations fiables sur lesquelles construire. Le plus gros problème est qu'il serait difficile d'abandonner ces problèmes aux autres, tant pour nous que pour eux. Moralement et logiquement, c'est à nous de le faire.

### 5.2 Améliorations possibles

#### 5.2.1 Prélude

Pour nous, avant de pouvoir envisager une continuation du projet, il faudrait :

**Rendre plus solide la partie informatique** qui ne suffit que pour une présentation, avec un technicien à coté. Pour que quelqu'un d'autre s'en serve (a fortiori un non-informaticien), il y a un nombre relativement important de petites fonctionnalités à ajouter et de problèmes à résoudre. Il faudrait qu'il n'y ait qu'à allumer l'appareil pour pouvoir se servir de la partie informatique, et qu'elle se comporte bien en cours d'exécution.

**Régler les problèmes électroniques** comme la batterie des moteurs qui semble s'éteindre sans qu'on puisse y faire grand-chose (un problème gênant à l'utilisation, mais pas bloquant). Si aucune solution n'est trouvée, il faut remplacer les batterie par des modèles sans mode de mise en veille.

## 5.2.2 Projets d'amélioration

Nous avons dressé une liste de projets que nous pensons intéressants :

- **Bras robot**

Spécialité : EES, Info

*Développement d'un petit bras robot commandé de la même manière que les moteurs.*

Dimension : 3 personnes

Avantage : relativement indépendant

- **Ajout du son**

Spécialité : EES, Info ?

*Ajout d'un micro et d'un traitement de son pour supprimer le bruit des moteurs.*

Dimension : 1-2 personnes

- **Commande autonome**

Spécialité : EES, Info

*Application des cours de traitement d'image (des EES) pour permettre par exemple au robot d'avancer en évitant les obstacles.*

Dimension : 3-4 personnes

Avantage : relativement indépendant du reste du projet

- **Modem 3G/4G**

Spécialité: Info

*Permet une connexion internet à bien plus grande distance que le Wi-Fi.*

Dimension : 2 personnes

- **Nouveau châssis**

Spécialité : EES, Materiaux

*Créer un châssis sur mesure (et faire un meilleur cablage) pour obtenir un ensemble plus solide.*

Avantage : dimensionnement flexible

# Conclusion

Notre prototype dispose de toutes les fonctionnalités qu'on attendait de lui, même si il reste quelques problèmes non critiques. Nous avons réussi à diminuer la latence, en dessous d'une demi-seconde, et n'avons utilisé que des technologies déjà incluses dans tous les navigateurs web modernes ; nous atteignons aussi l'objectif d'accessibilité, étant donné que même les téléphones peuvent utiliser notre système sans installation d'une application spécifique. Qui plus est, nous avons découvert des pistes intéressantes, comme la commande via Internet qui n'était à l'origine pas prévue. L'étude se conclut donc sur un succès. Il reste, bien sûr, encore quelques aspects à améliorer, comme l'interface, plus de fonctionnalités, ou encore une coque pour protéger et embellir le robot. Ce que nous avons réaliser peut aussi servir de base à un autre projet utilisant ses caractéristiques (mobilité, flux vidéo).

# Annexe technique

## A.1 Informatique

### A.1.1 Solution logicielle

Notre Raspberry Pi 3 fonctionne sous Raspbian. Nous utilisons le pipeline suivant :

1. `raspivid` capture la vidéo (en H.264) et l'envoie sur la sortie standard
2. Il est pipé à `ffmpeg` qui le réencode en MPEG-1
3. Il est pipé au serveur asynchrone Tornado qui le récupère avec un `PipeIOStream`. Le signal est alors envoyé via un `WebSocket` au client.
4. Le client (à qui on a fait télécharger le lecteur `jsmpeg` en Javascript) peut lire la vidéo grâce aux extensions `HTML5-MSE`

On tire parti des technologies récentes WebSocket et surtout HTML5-MSE. Le lecteur JSmpeg utilise le codec MPEG-1 (qui lui est antique) car les brevets ont expiré. Une solution bien plus optimale serait d'utiliser directement le format H.264, qui est bien plus performant, en plus d'être nativement disponible au Raspi3 ; on pourrait donc économiser toute la tâche de réencodage tout en profitant d'une meilleure qualité. Malheureusement, H.264 est une technologie propriétaire et payante qui n'est donc pas disponible partout.

On a tenté d'utiliser `MPEG-DASH`, mais le principe opératoire (découper la vidéo en petits segments) ne permet pas une latence suffisamment basse pour ce que nous voulons faire. On a disqualifié des technologies comme HLS d'office, n'étant pas assez universelles.

La commande est elle aussi faite à partir d'un `WebSocket`. Le système utilise un pouls régulier pour s'assurer que la connexion est toujours bonne.

### A.1.2 Puissance de calcul nécessaire

Nous avons voulu le dernier modèle d'ordinateur embarqué Raspberry Pi 3, pensant qu'un flux vidéo en direct serait gourmand. A l'heure actuelle, le processeur est à 75% dédié à l'encodage vidéo et de 5 à 10% dédié au serveur web : la puissance était donc justifiée. Notons que le Raspi 3 dispose d'une puce graphique dédiée, et que l'encodage tire parti des 4 coeurs du processeur.

### A.1.3 Latence

Le travail sur la latence est essentiel pour que le flux vidéo soit utile pour le pilotage. Nous parvenons à une latence inférieure à une demi-seconde, ce qui est acceptable. Cela dit, quelqu'un ayant monté une pile comparable (`MPEG-1` via `WebSocket` à `JSmpeg`) dit n'obtenir que 50 millisecondes de latence, ce que nous n'atteignons pas.

## A.2 Électronique

### A.2.1 Châssis

Le châssis tout-en-un utilisé est un "Tchang DD1-1" que l'on peut trouver sur plusieurs sites de 30 à 40€. On peut accéder directement aux bornes des moteurs, mais il y a aussi une couche d'électronique avec un pont en H et un capteur de vitesse. Le pont en H nous permet de réaliser un hacheur afin de diminuer la tension perçue par les moteurs. La fréquence du signal de commande a été choisie à 1kHz afin d'être grandement supérieure à la constante de temps des moteurs, et ainsi fournir une alimentation semblable à du continu. Les transistors du pont sont pilotés par pairs à l'aide des pins IA et IB. Alimenter IA connecte le moteur à l'endroit, alimenter IB le connecte à l'envers. Il est possible de réduire le nombre de signal de pilotage à un par moteur au lieu de deux si on place une porte non avant l'un des pins. Nous avons fait de la PWM logicielle (avec le Raspberry Pi), et le capteur de vitesse pouvait être utilisé dans un système ajustant la vitesse des moteurs (asservissement numérique), mais nous n'en avons pas eu besoin. En effet, la qualité des moteurs n'est pas très bonne et ils ont des caractéristiques très différentes, ce qui fait que notre robot aura tendance à aller sur le côté parce qu'un moteur va plus vite que l'autre. En pratique, le robot avance tout droit grâce aux chenilles. Cela nous évite d'appliquer des coefficients de correction ou de réaliser un asservissement à l'aide du capteur de vitesse.

Afin de simplifier le câblage du robot, nous avons installé une plaquette de connections sur le dessus. Les pins du châssis sont ramenés sur le dessus grâce à des nappes. Cela nous a permis une plus grande accessibilité et une plus grande souplesse sur le câblage. Il aurait été très difficile de réaliser un prototype en faisant des soudures à chaque fois qu'un changement était nécessaire.

### A.2.2 Batteries

On a utilisé comme alimentation des batteries de téléphone à 10€ environ l'une, de 5 Ah, capables de délivrer 5V/2A, ce qui satisfait le Raspberry Pi 3 (qui refusera de démarrer ou affichera une icône d'alerte sur un câble USB typique 5V/1A). L'alimentation par USB et les mécanismes de sécurité intégrés les rendent très commodes (et on peut facilement les connecter à un circuit électronique en découpant un câble USB 4-broches), mais elles souffrent d'un sérieux défaut : conçues pour des téléphones, elles se mettent en veille de manière intempestive sans qu'on puisse les en sortir quand on les connecte à nos moteurs.

On a mentionné qu'il serait préférable d'alimenter les moteurs en 8V et pas 5V. Il y a deux solutions : utiliser des batteries à cette tension, ou utiliser un hacheur élévateur pour générer 8V à partir de nos batteries 5V; cela aurait pour effet de doubler la vitesse. En pratique, les moteurs consomment 300mA à vide, montant jusqu'à 600mA en charge. Nous avions peur d'avoir un appel de courant trop fort au démarrage, mais il s'est avéré que nous n'avons pas eu besoin de mettre un système de démarrage spécifique.