Rapport technique - Robot Veronica

Aymeric COURILLEAU, Martin GARDINETTI, Luan JUPPET et Elwyn VINKLER

Contexte et problématique

Face à la question “Y a-t-il de la vie sur Mars ?” et à l’importance scientifique des découvertes effectuées sur la planète rouge, la société SOLEC s’est lancée vers de nouveaux horizons à la conquête de l’espace, et a commandité un robot guidé à distance permettant d’explorer le sol particulier de Mars, pour y détecter la présence d’eau. Il a été nommé Veronica, en référence à la série télévisée américaine Veronica Mars.

Son parcours, composé de tronçons de lignes droites et de virages, sera transmis au fur et à mesure depuis une base terrienne. Les données collectées seront enregistrées et transmises à intervalle régulier. Ce robot sera basé sur un système à deux roues indépendantes motorisées et à une roue libre pour se déplacer. Des données sur son environnement, prises tous les 10 cm par différents capteurs, seront transmises toutes les 10 minutes à la base.

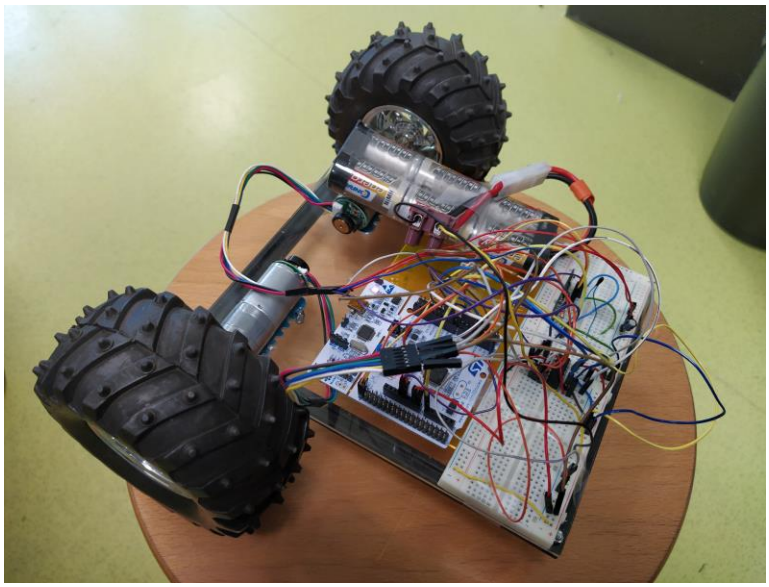

Figure 1 - Prototype final du rover.

Les performances attendues en termes de rapidité, de fiabilité, d’autonomie et d’ergonomie sont listées ci-dessous :

- Vitesse comprise entre 10 et 30 cm/s.
- Erreurs maximales de 2 cm sur la position et de 3° sur l’angle.
- Parcourir 1 km sans rechargement des batteries.
- Interface homme-machine intuitive.

Notre équipe, composée de Aymeric Courilleau, Martin Gardinetti, Luan Juppet et Elwyn Vinkler, a accepté le défi et tenté de concevoir un tel robot en une vingtaine d'heures, accompagnée des équipes de SOLEC.

I - Solution technique proposée

Nous verrons dans cette partie les solutions techniques retenues lors de la conception du robot Véronica, à savoir les solutions sur la motorisation du robot, sur l'interface homme-machine d'entrée des commandes, et sur le système et le langage de télécommunication entre la base et le robot. Toutes n'ont pas pu être mises en pratique, mais les éléments relatifs à leur mise en place sont donnés.

1 - Motorisation :

La surface de Mars est un terrain accidenté, avec des zones sableuses propices à l'embourbement des roues, ou encore des pierres très abrasives. Dans ce contexte, et pour assurer une mobilité totale à notre rover, les roues doivent être bidirectionnelles, et commandées indépendamment.

Cette architecture est en effet très pratique, car elle permet une très grande liberté de déplacement. Il lui est possible, en plus d'avancer tout droit ou de reculer, de tourner sur lui-même ou d'effectuer un virage avec un quelconque rayon de braquage. Cependant, cette architecture présente quelques contraintes : il faut piloter les deux moteurs en même temps, c'est-à-dire les synchroniser au démarrage, et les asservir en vitesse, car ils ont une réponse différente à une même tension d'entrée.

Nous verrons comment nous avons pensé et surmonté ces difficultés avec les solutions que nous avons eues le temps de mettre en place.

Architecture de la motorisation

Afin de piloter les moteurs à courant continu dans les deux sens, nous avons mis en place un montage constitué de deux ponts en H commandés par une carte Nucléo. Pour cela, nous avons utilisé le composant L293D de chez Texas Instruments, qui inclut plusieurs ponts en H. Le schéma du circuit est donné en **Figure 2**. Une capacité a été branchée en parallèle de l'alimentation afin de sécuriser le L293D en cas de brusques coupures de courant, telle qu'une déconnection de fils.

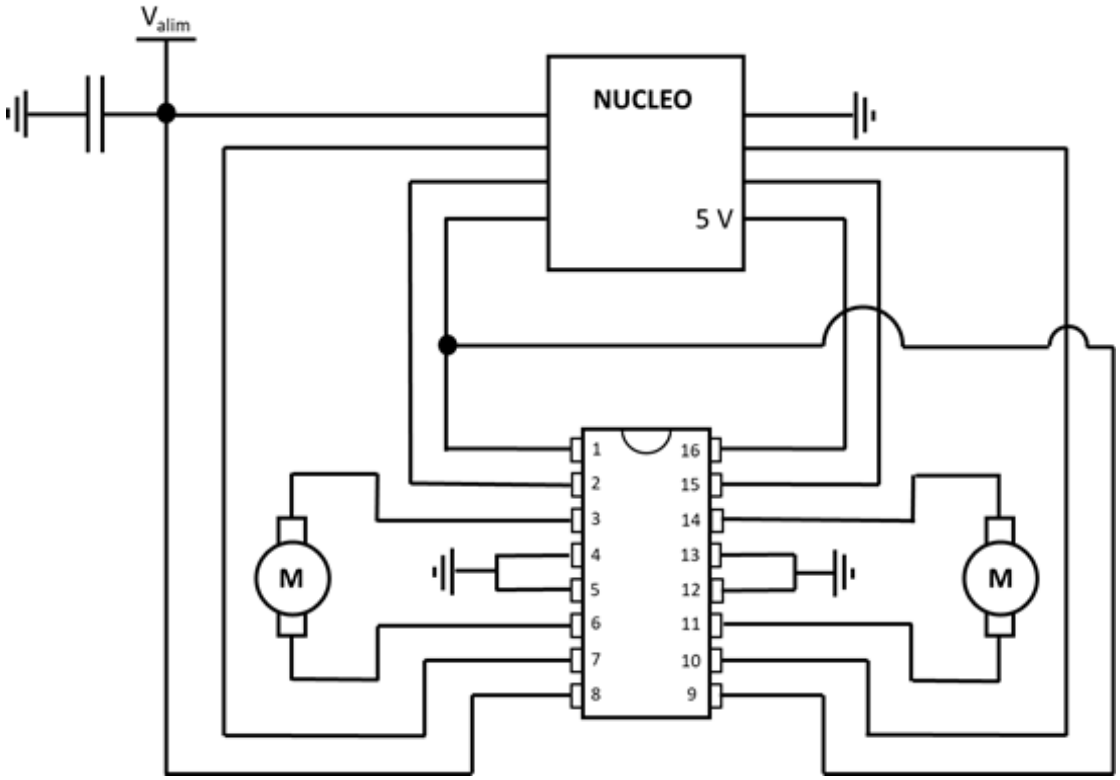

Figure 2 - Circuit autour du L293D pour la motorisation du robot.

Les entrées 1, 2, 7, 9, 10 et 15 sont reliées à la carte Nucléo et sont des entrées logiques qui permettent de piloter la vitesse et le sens de rotation des moteurs. L'alimentation des moteurs est reliée à l'entrée 8. Cette même alimentation permet d'alimenter la carte Nucléo ; elle est constituée d'une batterie délivrant une tension de 8V. Les sorties 3, 6, 11 et 14 sont reliées aux moteurs. Les pattes 4, 5, 12 et 13 sont branchées à la masse. Enfin, la patte 16 est reliée à la sortie 5 V de la nucléo afin d'alimenter le composant. Les moteurs sont pilotés avec des commandes PWM qui délivrent un signal créneau de rapport cyclique ajustable. Il faut donc utiliser ce type de sortie pour les pattes 2, 7, 10 et 15. Les sorties 1 et 9 doivent être maintenues « hautes » pour faire tourner les moteurs. Nous utilisons donc une sortie analogique commune qui délivre une tension continue. Il est possible de ne plus délivrer de tension sur les sorties 1 et 9 pour permettre au robot de se déplacer en roues libres.

Cette architecture fonctionne très bien, cependant les moteurs ne sont pas synchronisés et ne tournent pas exactement à la même vitesse pour une même commande. Nous avons constaté une différence constante en fréquence de 50 Hz entre les deux moteurs, soit de 0,2 tours.s^{-1}. Cela est gênant pour aller correctement dans la direction souhaitée. Il faut donc mettre en place un asservissement en vitesse.

Pour cela, on utilise les codeurs incrémentaux dont disposent les moteurs. Ces codeurs renvoient un signal créneau dont la fréquence est proportionnelle à la vitesse des moteurs. La documentation technique ne donnant pas le nombre d'incréments sur les codeurs, nous avons comparé le nombre de tours effectués à la fréquence renvoyée. Nous avons mesuré un total de 240 incréments. Les données renvoyées par ces codeurs vont nous permettre d'asservir les moteurs en vitesse et le rover en position.

Asservissement en position

Un premier asservissement en position a été mené afin d'assurer le respect de la consigne en cas de dépassement malheureux du point d'arrivée. La contrainte principale étant la nécessité de poser un point d'arrêt à la commande en cours pour pouvoir procéder à la suivante, il a fallu mettre en place le concept de temps d'arrêt. Avec une durée suffisamment longue pour garantir l'atteinte de l'objectif par le robot, et suffisamment courte pour qu'il puisse passer au suivant assez rapidement, ce temps d'arrêt commence à partir du moment où le robot est passé à proximité de l'objectif avec une incertitude d'un incrément sur la position des roues. Pendant ce temps, la commande d'arrêt des moteurs est lancée.

Au bout du temps d'arrêt, si le robot a dépassé la limite d'un incrément au-delà de l'objectif, il fait marche arrière avec pour objectif le précédent point qu'il a dépassé. Sinon, il passe à la commande suivante. L'asservissement en position n'a été mené que sur la roue gauche, une seule roue étant suffisante lorsqu'un asservissement en vitesse est en place. Aucun test rigoureux n'a pu être mené à cause du manque de l'asservissement en vitesse, sans lequel le robot avance mal.

Asservissement en vitesse

Un second asservissement en vitesse a été pensé, et attend d'être mis en place pour être testé. Il est en effet nécessaire puisque pour une même tension d'alimentation, la réponse en vitesse peut être différente, comme représenté en **Figure 3**. Cela peut s'expliquer par les inductances qui composent les moteurs. Il y a également le couple à vide qu'il faut vaincre pour démarrer le moteur qui peut être différent. L'usure de l'axe peut, par exemple, expliquer cette différence. Pour résoudre ce problème de moteur qui ne respecte pas la consigne en vitesse, il faut mettre en place un asservissement en boucle fermée sur la vitesse de rotation.

Nous savons que la réponse d'un moteur à courant continu s'exprime de la façon suivante : $V_{mot} = k\omega - RI$. Dans cette expression, nous voyons que la vitesse est proportionnelle à la tension appliquée au moteur et que l'ordonnée à l'origine dépend du courant qui est proportionnel au couple résistant.

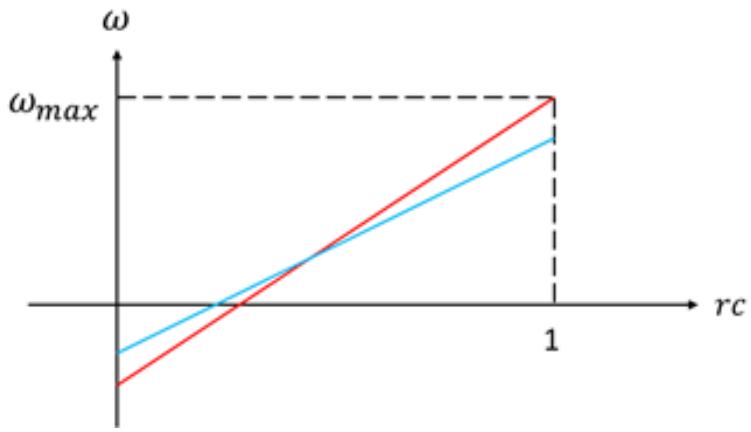

Figure 3 - Représentation de la réponse des moteurs.

Nous voyons que nous pouvons déterminer le coefficient directeur de la commande à vide. Si le couple varie, il ne fera que translater toute la réponse comme on peut le voir sur la **Figure 4**. Nous devons chercher les réponses des deux moteurs afin de pouvoir déterminer correctement la correction à appliquer. Pour commencer, nous devons déterminer la vitesse maximale à vide du moteur, puis nous devons chercher toujours à vide le rapport cyclique pour lequel le moteur démarre. Ces deux mesures nous permettent de déterminer le coefficient directeur de la réponse du moteur :

$$A = \frac{\omega max}{1 - rc_0} = \frac{f_{max}}{k_{capt}(1 - rc_0)}$$

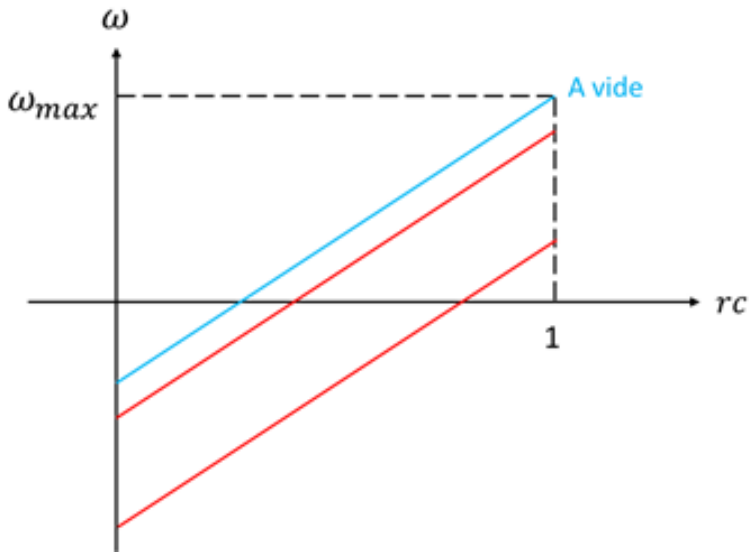

Figure 4 - Représentation de la réponse d'un moteur pour différents couples.

Nous savons qu'à vide la vitesse de rotation du moteur suit l'équation $\omega = (rc - rc_0)\ A$. Le rapport cyclique pour lequel le moteur démarre dépend du couple résistant. Grâce au capteur sur le moteur, nous pouvons déterminer la vitesse réelle et la comparer à la consigne : $\Delta\omega = \omega_c - \omega_{mes} = \omega_c - f_{mes}/k_{capt}$

Nous avons ainsi accès à la correction en rapport cyclique : $\Delta rc = \Delta\omega/A$. A chaque instant où nous mesurons la vitesse du moteur, il faut la réajuster :

$$rc = rc + \Delta rc = \frac{2\omega_c}{A} + rc_0 - \frac{f_{mes}}{A.k_{capt}}$$

Dans cette formule, rc_0 est le rapport cycle nécessaire pour faire démarrer le moteur à vide. Il doit être très proche de zéro. Il faut remarquer que si le couple devient trop fort alors il ne sera pas possible d'atteindre la vitesse demandée car le rapport cyclique ne peut pas être supérieur à 1. De même, si la vitesse de consigne demandée est supérieure à la vitesse maximale à vide alors le rapport cyclique de consigne sera de 1 et la vitesse souhaitée ne sera pas atteinte.

Nous avons pensé mettre en place un asservissement numérique. Nous n'avons pas eu le temps de le mettre en place mais nous avons néanmoins réfléchi à comment faire cet asservissement. Nous pensons faire des mesures à intervalle de temps régulier pour déterminer en continu la fréquence du signal fourni par le capteur. Le principe est de mesurer le temps séparant deux montées (ou descentes) successives du signal. Puisque le signal est périodique, nous pouvons calculer la fréquence du signal de la façon suivante : $f_{mes} = \frac{1}{nT_e}$

La période du signal est un multiple de la période d'échantillonnage. A chaque fois que nous sommes en présence d'un front montant, nous pouvons calculer la fréquence du signal et ajuster la commande en rapport cyclique.

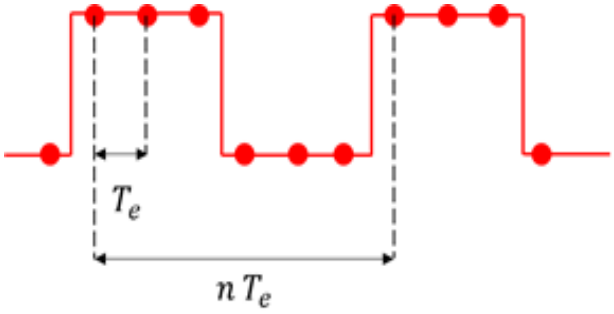

Figure 5 - Échantillonnage du signal issu des capteurs des moteurs.

Pilotage des moteurs

Les commandes qui ont été mises en place pour piloter le robot permettent de faire de la marche avant ou de la marche arrière sur une certaine distance, de tourner sur lui-même dans le sens horaire ou antihoraire d'un certain angle, et de tourner à droite ou à gauche avec un rayon libre de braquage en marche avant comme en marche arrière.

Pour communiquer ces commandes à la carte Nucléo, nous avons pensé à les transmettre sous forme d'un triplet ayant pour coordonnées : la distance à parcourir ou le rayon de braquage (cela dépend de la commande), l'angle de rotation et la vitesse. L'information sur le sens de la marche est portée par le signe de la distance. Le sens de la rotation est, quant à lui, porté par le signe de l'angle tandis que la vitesse est toujours positive. La **Figure 6** décrit les mouvements du robot en fonction du triplet de commande entré.

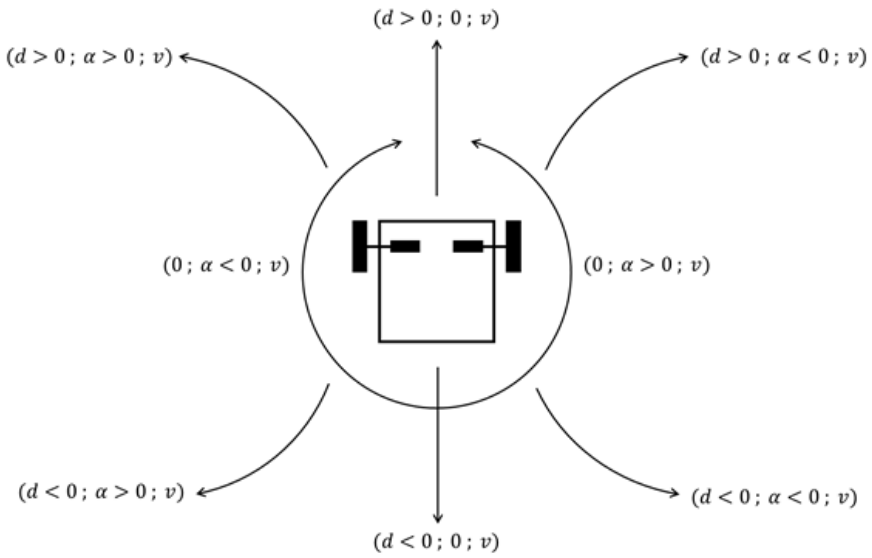

Figure 6 - Mouvements des moteurs associées aux triplets de commandes envoyés.
d : distance, α : angle, v : vitesse

Pour se déplacer en ligne droite, l'angle de consigne doit être nulle et pour tourner sur soi-même, la distance de consigne doit être nulle. La commande permettant de stopper le robot est la consigne de rayon et d'angle nul. Pour tourner à droite, l'angle de consigne doit être négatif (il s'agit du sens trigonométrique si on regarde le robot du dessus comme sur la **Figure 6**), il faut donc un angle de consigne positif pour tourner à gauche. A partir des triplets de commande, nous pouvons déterminer les consignes que nous devons appliquer aux moteurs. Les consignes fournies aux moteurs sont le rapport cyclique des sorties PWM ainsi que la condition portant sur les angles dont les moteurs doivent tourner.

On utilise les notations suivantes :

- ω_x est la vitesse de rotation de la roue x (négatif si marche arrière)
- φ_x est l'angle en valeur absolue que doit faire la roue x
- L est la largeur du châssis
- R est le rayon des roues

Le calcul des consignes se fait de la manière suivante :

- Rotation sur lui-même $d = 0$:
    - Rotation vers la gauche $\alpha > 0$:

- $\omega_G = v\frac{-L}{2R}$ et $\omega_D = v\frac{L}{2R}$: vitesses angulaires des roues
- $\varphi_G = \alpha\frac{L}{2R}$ et $\varphi_D = \alpha\frac{L}{2R}$: angles à parcourir

- Rotation vers la droite $\alpha < 0$:
  - $\omega_G = v\frac{L}{2R}$ et $\omega_D = v\frac{-L}{2R}$
  - $\varphi_G = |\alpha|\frac{L}{2R}$ et $\varphi_D = |\alpha|\frac{L}{2R}$

- Marche avant $d > 0$:
  - Déplacement rectiligne $\alpha = 0$:
    - $\omega_G = \frac{v}{R}$ et $\omega_D = \frac{v}{R}$
    - $\varphi_G = \frac{d}{R}$ et $\varphi_D = \frac{d}{R}$
  - Virage à gauche $\alpha > 0$:
    - $\omega_G = v\frac{d}{R(d+L/2)}$ et $\omega_D = v\frac{d+L}{R(d+L/2)}$
    - $\varphi_G = \alpha\frac{d+L}{R}$ et $\varphi_D = \alpha\frac{d}{R}$
  - Virage à droite $\alpha < 0$:
    - $\omega_G = v\frac{d+L}{R(d+L/2)}$ et $\omega_D = v\frac{d}{R(d+L/2)}$
    - $\varphi_G = |\alpha|\frac{d}{R}$ et $\varphi_D = |\alpha|\frac{d+L}{R}$

- Marche arrière $d < 0$:
  - Déplacement rectiligne $\alpha = 0$:
    - $\omega_G = \frac{-v}{R}$ et $\omega_D = \frac{-v}{R}$
    - $\varphi_G = \frac{|d|}{R}$ et $\varphi_D = \frac{|d|}{R}$
  - Virage à gauche $\alpha > 0$:
    - $\omega_G = v\frac{|d|}{R(|d|+L/2)}$ et $\omega_D = v\frac{|d|+L}{R(|d|+L/2)}$
    - $\varphi_G = \alpha\frac{|d|+L}{R}$ et $\varphi_D = \alpha\frac{|d|}{R}$
  - Virage à droite $\alpha < 0$:
    - $\omega_G = v\frac{|d|+L}{R(|d|+L/2)}$ et $\omega_D = v\frac{|d|}{R(|d|+L/2)}$
    - $\varphi_G = |\alpha|\frac{|d|}{R}$ et $\varphi_D = |\alpha|\frac{|d|+L}{R}$

Pour les virages, la consigne de vitesse communiquée est appliquée à la roue la plus proche du centre du cercle de braquage. Dans le script du programme permettant de piloter les moteurs, il y a trois sorties pour chaque moteur dont une commune. Lorsque les vitesses sont négatives, il faut juste changer le rapport cycle de sortie pour aller en marche arrière. L'angle calculé que doit faire une roue permet de réaliser un asservissement en boucle ouverte afin d'atteindre la consigne. En effet, chaque moteur est équipé d'un capteur qui délivre un signal en créneau. Il est ainsi possible de faire un codeur incrémental en comptant le nombre de fronts montants (ou descendants) afin de connaître l'angle que les roues ont réalisé et d'arrêter la séquence de consigne en cours pour passer à la suivante.

2 - Interface homme-machine :

L’interface homme-machine a été codée sous Python 3, avec la bibliothèque d’interfaçage graphique PyQt5, et s’ouvre depuis un exécutable. La fenêtre de l’interface est affichée en **Figure 7**, et se décompose en 3 parties :

- La connexion et l’envoi de commandes au robot en rouge.
- L’affichage des données récupérées par les sondes du robot en bleu.
- L’affichage des images prises par les caméras du robot en vert.

La première partie est constituée d’un bouton “Établir la connexion", sur lequel il faut appuyer pour connecter le robot à l’interface. Une fois la carte Nucléo du circuit de la base terrienne branchée à l’ordinateur, il suffit d’appuyer sur le bouton pour établir la connexion avec la carte Nucléo du circuit du rover sur Mars. Une fois la connexion établie, un message indique que l’on est “Connecté“ au robot. Si elle n’est cependant pas trouvée, un message d’erreur indique un “Échec de la connexion".

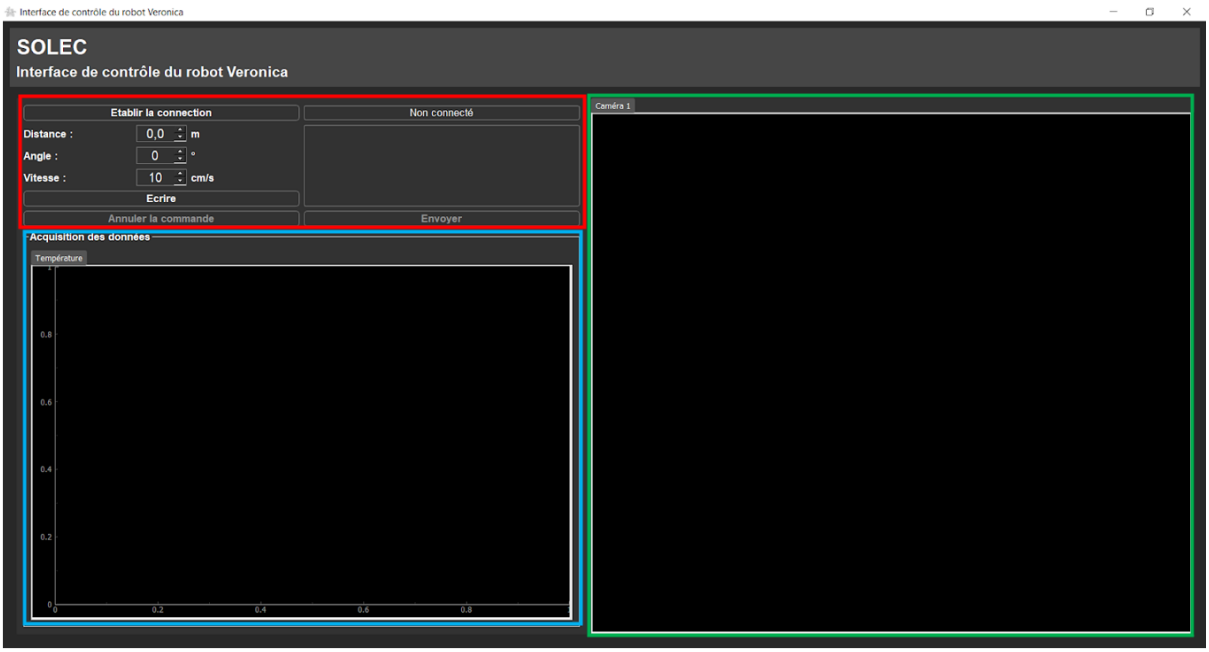

Figure 7 - Fenêtre de l’interface de contrôle du robot Veronica.

La distance comprise entre -999,9 et 999,9 mètres, l’angle de rotation compris entre -180 et 180 degrés et la vitesse comprise entre 10 et 30 cm/s peuvent être choisis depuis les boîtes tournantes associées, et compactés en une commande de 3 variables à envoyer au robot à l’aide du bouton “Ecrire”. Une fois la suite de commandes écrite, il suffit d’appuyer sur le bouton “Envoyer” pour que le robot reçoive les commandes et les exécute. Pour annuler la dernière commande écrite, il suffit d’appuyer sur le bouton “Annuler la commande”.

La deuxième partie est constituée d'autant de languettes qu'il y a de sondes sur le robot. Chaque languette a son propre graphique, et affiche la donnée associée à sa sonde en fonction du temps. A titre de rappel, les données sont reçues toutes les 10 minutes, et constituent la moyenne glissante des mesures prises tous les 10 cm. [Fonction dont l'implantation est à poursuivre]

La troisième partie est constituée d'une zone d'affichage des différentes caméras du robot. Chacune des images envoyées par le robot permettent d'obtenir une image de ce que voient les caméras. Pour passer d'une caméra à l'autre, il suffit d'appuyer sur la languette associée au-dessus de l'affichage. [Fonction dont l'implantation est à poursuivre]

3 - Télécommunication :

Les systèmes de télécommunications, pour la commande comme sur le véhicule, se composent d'une carte KAPPA M868 connectée à une carte Nucléo. Le système est construit de la manière suivante :

La carte Radio Fréquence (RF) est alimentée par la carte Nucléo en 5V. Une résistance de 100 kΩ est connectée au Pin 4 (cf. **Figure 8**). On connecte le Pin 7 RX à l'entrée d'information de la carte Nucléo et le Pin 8 TX à la sortie d'information de la carte Nucléo. De cette manière, les deux cartes sont connectées de sorte à être correctement alimentées et à pouvoir échanger convenablement des signaux. Le schéma du circuit est donné en **Figure 9**.

Pin No	Name	Direction	Description
1,11	GND	-	Ground connections
2	ANT	Both	Antenna connection matched to 50ohm
3	LED	Out	TX/RX notification LED
4	RTS	In	Low level RS232 RTS
5	CTS	Out	Low level RS232 CTS
6	NC	-	Do not connect
7	TX	Out	Low level RS232 data out
8	RX	In	Low level RS232 data in
9	NC	-	Do not connect
10	VCC	IN	Vcc +2.2 - 3.6V dc
12	NC	-	Do not connect

Figure 8 - Description des PIN

L'envoi des triplets de commandes s'effectue sous une forme compressée de caractères, afin de repousser la limite d'envoi des données dans le cas où il deviendrait important. La suite de caractères type utilisée se décompose de la sorte :

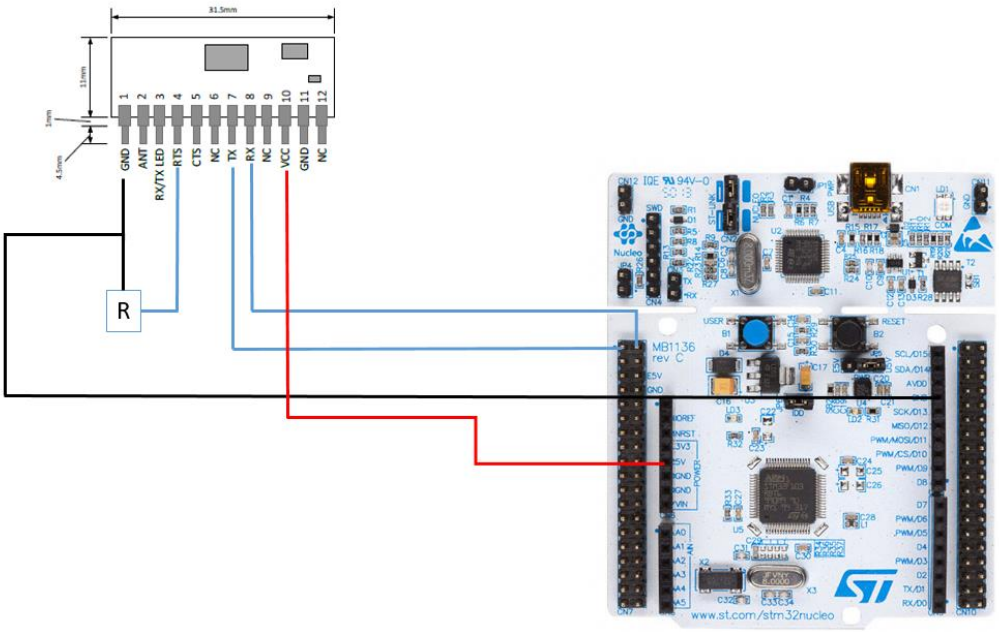

Figure 9 - Schéma connectiques des systèmes de télécommunication.

‘s_XXXX.X_YYYY_ZZ__XXXX.X_YYYY_ZZ_[...]_XXXX.X_YYYY_ZZ_e’, où XXXX.X représente la distance ou le rayon de braquage, YYYY représente l’angle et ZZ représente la vitesse. La distance s’écrit sur la plage [‘-999.9’, ‘0999,9’] en mètre, l’angle s’écrit sur la plage [‘-180’, ‘0180’] en degré et la vitesse s’écrit sur la plage [‘10’, ‘30’] en cm/s, d’où le nombre de caractères associé à chaque consigne. La suite de commande commence par un ‘s’ pour start et finit par un ‘e’ pour end, permettant de repérer le début et la fin des commandes parmi les données parasites que peut recevoir le robot. Chaque consigne est séparée par un ‘_’, et chaque triplet de commande est séparé par ‘__’. Les valeurs d’angle et de vitesse reçues par le robot sont ensuite converties en radian et en m/s.

II - Retour d’expérience

1 - Répartition du projet :

Pour réaliser ce prototype, nous avons réparti le travail en trois, selon les parties présentées précédemment. Elwyn était responsable de l’interface homme-machine, et Aymeric de la télécommunication. Ils ont travaillé ensemble à l’harmonisation de leurs codes respectivement développés. Quant à Martin et Luan, ils se sont occupés de la partie motorisation, dont l’électronique et l’alimentation. De nombreuses interactions entre chaque acteur de ce projet ont été nécessaires, afin d’uniformiser le code de commande du robot. Nous pouvons résumer le travail accompli, et sa répartition, en s’appuyant sur le schéma de la **Figure 10** et le tableau de la **Figure 11**.

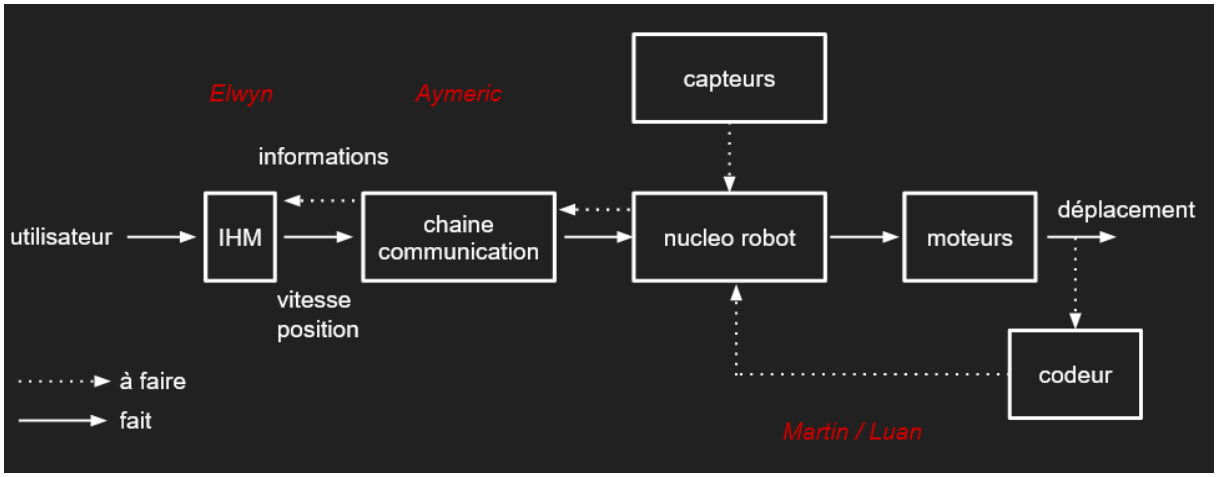

Figure 10 - Schéma bloc du projet avec ce qui est fait et ce qu'il reste à faire.

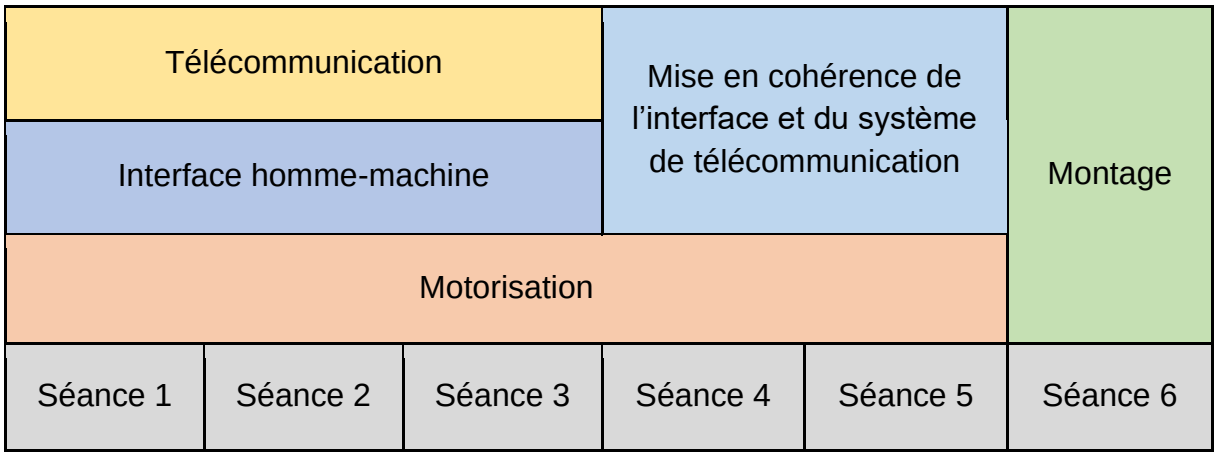

Figure 11 - Répartitions des tâches tout au long du projet.

Télécommunication

- Séance 1 : Recherche, planification et documentation sur le système de communication le plus adapté au pilotage du robot. Développement du code de télécommunication.

- Séance 2 : Finalisation du code et pratique de tests de communication entre deux unités. Problèmes de réception donnant suite à la recherche de son origine.

- Séance 3 : Problèmes résolus et finalisation de la mise en place d'un système de télécommunication à deux sens.

Interface homme-machine

- Séance 1 : Présentation d'un design d'interface programmé à l'avance. Réflexion sur les entrées des commandes depuis l'interface. Tentative de premiers contacts avec la carte Nucléo.

- Séance 2 : Modification du design d'entrée des commandes en prenant en compte les paramètres pensés lors de la séance précédente, donnant la version finale de l'interface. Mise en place du code de transformation des commandes en une forme facilement décodable propre à être envoyée au robot. Développement du code C de réception sous MBed pour vérifier les données envoyées.

- Séance 3 : Tests différents de l'interface, avec résolution de divers bugs. Mise en place de la réception en continu des données envoyées par le robot (thread). Tests de l'affichage vidéo avec la caméra de l'ordinateur, prêt à être employée pour afficher la vidéo des futures caméras du robot.

Mise en cohérence

- Séance 4 : Fusion du programme C de lecture des données envoyées par l'interface homme-machine et du programme C de télécommunication. Problème de réception de données erronées. Résolution du problème qui était lié à la vitesse d'envoi et de réception des données limitée par la carte Radio Fréquence.

- Séance 5 : Développement du programme de lecture des données réceptionnées par le robot. Problèmes de réception à partir de la lecture du second paramètre de la commande. Résolution du problème qui était lié à la caractéristique asynchrone de la communication, alors que le programme effectuait des calculs entre la réception de deux de données.

Motorisation

- Séance 1 : Prise en main des moteurs et réflexion sur le pilotage du robot.

- Séance 2 : Réflexion sur la structure de la motorisation et discussion avec la partie IHM quant à la forme des consignes communiquées aux moteurs.

- Séance 3 : Mise en place d'une première architecture de la motorisation. Difficultés rencontrées sur les ponts en H à cause du manque d'expérience sur de telles structures intégrés (*).

- Séance 4 : Tentative d'amélioration de la motorisation pour ne plus avoir à faire face, entre autres, à des déconnexions de fils électriques au niveau des moteurs, entraînant une détérioration des ponts en H.

- Séance 5 : Achèvement de la motorisation et réflexion sur l'asservissement en vitesse des moteurs.

* Nous avons perdu beaucoup de temps pour mettre en place la motorisation car nous n'avions auparavant jamais utilisé de pont en H et piloté de moteur. Ce précieux temps nous a manqué pour ensuite asservir convenablement les moteurs.

Montage

- Séance 6 : Fusion de la partie télécommunication + interface homme - machine avec la partie motorisation, complétant ainsi le montage du robot. Correction de bugs dans le programme de pilotage du robot suite à la réception des commandes. Pilotage du robot à distance réussi, mais problèmes de synchronisation des moteurs lié à l'asservissement en vitesse non encore développé.

2 - Compétences développées :

Pour mener à bien ce projet, nous avons dû mettre en œuvre et développer nos aptitudes de programmation, mais aussi de montage électronique. Les compétences à maîtriser pour avancer sur ce travail ont été :

- une bonne communication pour avoir un code homogène et fonctionnel,
- une connaissance du fonctionnement des moteurs à courant continu et des ponts en H,
- une connaissance des problématiques liées à l'asservissement (difficulté pour passer de la théorie à la pratique lorsqu'on en n'a jamais fait),
- une connaissance avancée en programmation, pour avoir une interface fonctionnelle et esthétique

Conclusion

Nous avons réussi à piloter notre robot avec des instructions données à distance sur une interface de commande. La vitesse du rover est bien comprise entre 10 et 30 cm/s. Des instruments de mesure ont été installés, il ne reste plus qu'à les étalonner. Nous sommes donc sur la bonne voie pour remplir tous les objectifs du cahier des charges.

Nous avons malheureusement eu quelques retards, dus à des problèmes techniques, ce qui nous a empêché de mettre en place l'asservissement en vitesse des roues, ainsi que le retour des données. Nous sommes en train de plancher sur ces problématiques, ainsi que sur l'ajout de caméras sur le rover et sur la conception de la structure externe du robot. Mais notre retard est uniquement le fruit de notre engagement sans faille pour cette mission. Chaque détail doit être pris en compte pour que ce projet soit un succès. Avec 2 ans de retard, Curiosity est une réelle prouesse technique. Laissez-nous encore quelques années et nous emmènerons des hommes sur Mars. Nous vous remercions pour votre confiance sans limite, de la part de toute l'équipe de conception du robot Veronica Mars.