RAPPORT DE PROJET

Commande d'un robot de grande taille

HAVARD Nicolas

IMA SA 4

POLYTECH LILLE

Boulevard Paul Langevin – Cité Scientifique
59655 VILLENEUVE D'ASCQ CEDX

Tel : +33-(0)3-28-76-73-60
Fax : +33-(0)3-28-76-73-61

Tuteur de projet : M. REDON Xavier

REMERCIEMENTS

Je voudrai remercier M. REDON Xavier pour l'aide qu'il m'a apporté durant ce projet et l'apport du matériel qui n'avait pas été prévu initialement, M. BOE Alexandre pour m'avoir donné son avis et m'avoir apporté les corrections nécessaires à la conception de mes circuits imprimés et M. FLAMEN Thierry pour son assistance lors de l'impression de ces cartes.

TABLE DES MATIERES

INTRODUCTION

Le projet Centaure est un robot de taille h*L*l 160*90*35 cm^3 repris par différents élèves depuis 2005. Il se déplace sur trois roues et est équipé d'écrans et d'une caméra Kinect. Le robot avance à l'aide de deux moteurs 24 V continu (des moteurs équipant habituellement des fauteuils roulants) qui permettent de faire tourner les deux roues arrières du robot. La roue avant, elle, est une roue folle qui suit le mouvement donné par les roues arrière. Ce projet ayant été repris pas de nombreux élèves, il s'appuie donc sur une certaine base existante et le châssis est ainsi déjà monté.

Au début de ce projet, le robot a été récupéré vidé de toute l'électronique qu'il comportait. Quelques capteurs sont manquants et/ou cassés. Cependant, les moteurs et leurs codeurs sont en place, ainsi que les deux écrans et la Kinect qui nécessitent seulement d'être connectés à l'ordinateur/serveur. Sur une table à côté du robot, les variateurs de vitesse permettant de piloter les deux moteurs sont commandés par un Arduino Mega et sa carte d'extension "Arduino MEGA Sensor Shield". Une partie électrotechnique et électronique est donc à prendre en compte dans un premier temps pour parvenir à utiliser les moteurs avant d'effectuer un coffrage pour l'étage "puissance" du robot. Il sera nécessaire de prévoir un système de sécurité capable de couper la puissance du robot en cas de problème.

Concernant la partie commande, les fonctions développées en langage Arduino l'année dernière permettant aux deux moteurs de faire reculer le robot semblent fonctionner. Les ordres que le robot doit effectuer à la fin du projet sont d'avancer, de reculer et de tourner à gauche ou à droite en fonction des touches sur lesquelles appuie un utilisateur. Cette commande se fera, dans un premier temps, via une télécommande. Il sera donc nécessaire de programmer la lecture de la commande, la gestion des différents capteurs ainsi que la commande des deux moteurs en fonction de l'ordre reçu par l'utilisateur.

L'objectif final de ce projet étant d'intégrer un serveur au robot pour recevoir les ordres de l'utilisateur via le Wi-Fi, il est nécessaire de faire communiquer le cerveau du robot, l'Arduino, avec le serveur grâce à une connexion série. L'utilisateur pourrait alors se connecter sur son appareil (ordinateur, tablette, smartphone) et donner des ordres au robot sur une interface web plutôt que d'utiliser une manette reliée au robot par un câble. Afin de diriger le robot à distance, une caméra Kinect est intégrée au système pour transmettre à l'utilisateur l'image de l'environnement où se trouve le robot.

Ce projet ayant été travaillé par plusieurs équipes d'étudiants IMA, nous avons à notre disposition des archives :

- cinq rapports papiers de 2005 à 2009
- quatre CD-ROM
- un wiki étudiant de l'année 2017-2018

Ces archives regroupent le travail fait par les différentes équipes en intégrant aussi le code écrit et quelques documentations concernant les capteurs et les variateurs de vitesse.

CACHIER DES CHARGES

I. OBJECTIFS

Le premier objectif principal de ce projet est de remettre en état le robot. Ceci impliquant de refaire l'intégralité de l'étage de puissance comportant l'électronique de puissance pour déplacer le robot et les différents systèmes de sécurité (bouton d'arrêt d'urgence, fusibles). La visualisation du couple fournit par les moteurs ou encore l'état de charge des batteries pourront être des caractéristiques intéressantes à faire ressortir.

Une fois l'étage de puissance terminé, un second étage permettant de commander le robot sera mis au point : cet étage disposera de l'Arduino Mega reliée aux différents capteurs du robot ainsi que les variateurs de vitesse commandant les moteurs. Il devra de plus pouvoir accueillir l'unité centrale de l'ordinateur qui servira à l'avenir de serveur Wifi.

Le second objectif est d'établir un programme permettant de commander les différents modules du robot afin que ce dernier obéisse à une consigne de vitesse. Pour tester cette fonctionnalité, une manette filaire sera conçue et enverra des ordres au robot qui devra y répondre.

Pour finir, établir une connexion série fonctionnelle entre l'Arduino et l'ordinateur du Centaure afin d'anticiper l'échange d'informations entre ces deux appareils pour contrôler le robot via Wifi à l'avenir.

II. CONTRAINTES

• Le projet partant d'une base existante, il est nécessaire d'utiliser le matériel à disposition tel que les moteurs, les variateurs de vitesse ou encore les différents capteurs.

• Le châssis du robot ayant déjà été conçu, la solution apportée devra être contenu dans l'enceinte prévue à cet effet.

• Le projet devra être terminé avant le 21 décembre 2018.

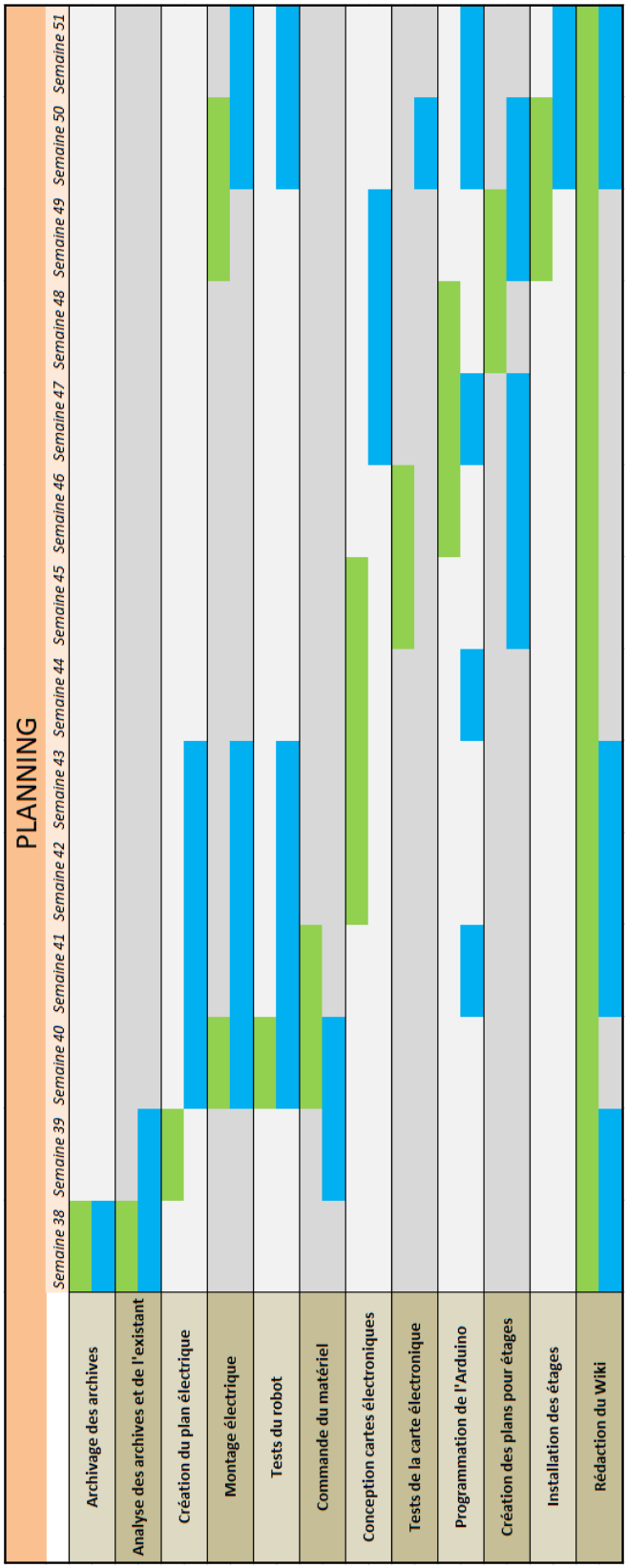
• Le robot devra être sécurisé et pourra être arrêté à n'importe quel moment grâce à un bouton d'arrêt d'urgence.

• L'esthétique du projet ne devra pas être négligée : des cartes électroniques et un coffrage devront être proposés.

III. MATERIEL A DISPOSITION

Matériel à disposition :			
Description	*Marque*	*Nb*	*Commentaire*
châssis du robot		1	Châssis monté avant le projet
batterie GF 12 22 Y (12V / 22 Ah (C5))	Sonnenschein	2	Environ 8V aux bornes des deux batteries initialement. 12 V après recharge
chargeur et testeur de batterie 12V GT6	ELEC	1	
moteurs SRG0131 24V / 15.5 A / 0.35 kW / 10 km/h	INVACARE	2	Déjà intégrés au châssis
roue		3	Une roue folle et deux roues de fauteuil roulant. Toutes déjà intégrées au châssis
codeur GC10K-04 11200	Baumer IVO	2	Déjà intégrés au châssis au niveau des roues arrière
variateur de vitesse 8CH2QM.2	Italsea	2	Sur la table, déjà reliés aux moteurs, aux batteries et aux relais
bouton d'arrêt d'urgence		1	En plusieurs parties en dehors du robot
Arduino Mega	Arduino	1	Déjà équipée de la carte d'extension, reliée aux relais et prête à être programmée
Arduino MEGA Sensor Shield	Arduino	1	Déjà fixé à l'Arduino et à la carte de relais
carte équipée de 4 relais KEYES ver. 4R1B	Funduino	1	La carte est déjà reliée aux variateurs de vitesse sur la table
relai 12 VDC Tongling		1	Neuf, encore emballé
capteur infrarouge 2Y3A003 F	SHARP	4	3 sont fixés sur l'avant du châssis, 1 détaché
capteur infrarouge 2Y0A02	SHARP	2	Un 2Y0A02 46 fixé à l'arrière du robot et un 2Y0A02 4X dont les fixations ont été détruites et les pins abimés
contacteur LC1D18	Telemecanique	1	
convertisseur Tel 5-2422 18-36 VDC -> +/- 12 VDC	TracoPower	1	
convertisseur TEN40-2411 24 VDC -> 5 VDC (8 A)	TracoPower	1	Légèrement abimé, dommages "esthétiques"
écran 4/3 ProLite E430 & E430S	iiyama	2	Deux écrans dont un fixé à l'avant du robot
caméra Kinect	Microsoft	1	Fixée à l'avant du robot en haut du mât

IV. CALENDRIER PREVISIONNEL



DEROULEMENT DU PROJET

I. Prémices : analyse de l'existant et achat de matériel

I.A Archives

Ce robot étant l'objet de plusieurs projets depuis une dizaine d'années, il existe donc de nombreuses traces écrites laissées par les anciens étudiants. Ces traces écrites sont principalement des rapports de projet traitant des capteurs de distance et des variateurs de vitesse. Si certains étudiants ont laissé un CD en plus de leur rapport permettant la numérisation facilitée de leurs sources, la plupart des rapports sont seulement sous format papier et n'ont pas été retranscrit sur un dépôt en ligne. Il existe aussi des disques non liés à un rapport papier. De plus, un étudiant ayant travaillé sur un wiki projet tel que celui-ci l'année dernière, nous avons ainsi accès au travail effectué par cette personne.

Afin de regrouper l'intégralité de ces informations sur un dépôt en ligne, un lien OneDrive permet d'accéder aux informations numérisées des anciens étudiants : https://1drv.ms/f/s!Agdnb608xp_RifFzM-b3jDG6AuN4-Q

Les archives sont contenues dans le dossier "Archives" et le dossier "Init" contient des photos du robot et de son équipement au début du projet. Afin de pérenniser ces archives, une archive au format ZIP de ce dossier sera disponible sur le Wiki de ce projet.

I.B Etude de l'existant : des archives au robot lui-même

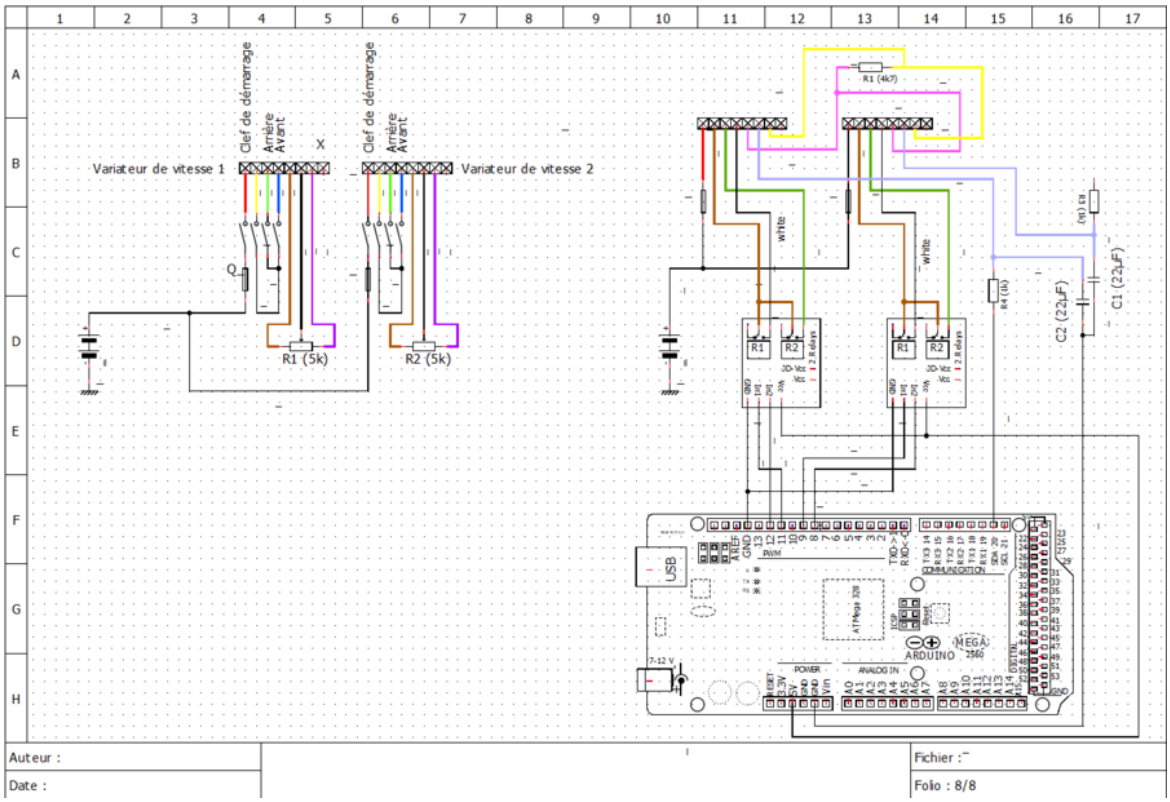
Une étude des archives écrites par nos prédécesseurs ainsi qu'une analyse de l'état du système actuel nous ont permis d'en apprendre plus sur le robot et ses constituants. Nous pouvons alors distinguer les différentes parties du robot comme le châssis, les batteries, les variateurs de

vitesse, les moteurs, les roues, les codeurs, les capteurs de distance, le microcontrôleur et les relais. Nous allons détailler chaque partie :

- Concernant le châssis du robot, il s'inscrit dans un volume de 160*90*35 cm^3. Le bloc permettant d'héberger les étages de puissance et de commande a pour volume 60*35*26 cm^3 = 54.6 dm^3. Les batteries ayant un volume de 16*13*17 cm^3, avec donc 13 cm de hauteur, l'étage de puissance fera occupera probablement entre 15 et 18 cm de hauteur.

- Le robot est alimenté par deux batteries à plomb 12 V continu. Mises en parallèles, elles permettent donc d'obtenir une tension continue de 24 V nécessaire à l'alimentation des moteurs.

- En ce qui concerne les variateurs de vitesse, il s'agit de deux plaquettes CH2QM.2 de ItalSEA. Parmi les archives, nous avons retrouvé une configuration manuelle différente de celle retrouvée en début de projet permettant le contrôle des moteurs via une PWM du microcontrôleur. Le schéma suivant montre à gauche le câblage théorique issu de la documentation d'une des archives, et à droite se trouve le montage réalisé par mon prédécesseur :



Différences de configuration du bornier des variateurs de vitesse entre le schéma de câblage théorique de la datasheet (à gauche) et le câblage réalisé par nos prédécesseurs (à droite)

Sur ce schéma, nous pouvons voir le bornier tel qu'il est documenté :

- Pin 1 : il s'agit de la clef de démarrage. Un fil doit donc relier cette entrée à une alimentation 24V continue protégée par un fusible de 3A. L'interrupteur permet donc de couper l'alimentation du variateur de vitesse.
- Pin 2 : ce pin est une sortie qui, grâce à deux interrupteurs, permet de récupérer un signal sur le pin 3 ou sur le pin 4.
- Pin 3 : si le signal en sorti du pin 2 est reçu par le pin 3, alors le sens de rotation des moteurs est tel que le robot reculera.
- Pin 4 : de même que le pin 3, si le signal reçu par ce pin correspond à la sortie du pin 2, alors le robot sera contraint de se déplacer en marche avant.
- Pin 5 et 7 : ces deux pins sont reliés à l'aide d'un potentiomètre de 5kOhm. Le pin 5 correspond à l'une des pattes externes du potentiomètre et le pin 7 correspond à la patte opposée;
- Pin 6 : ce pin est connecté à la patte intérieure du potentiomètre et le potentiel appliqué sur cette entrée dépend donc de la valeur du potentiomètre.

- Nous avons à notre disposition deux moteurs équipés de freins. Ces moteurs sont des machines à courant continu alimentées en 24 VDC et de puissance nominale 350 W. Au régime nominale, le courant est de 15.5 A et la rotation de l'arbre post réducteur est de 152 tours par minute.

- Le robot dispose de trois roues : deux roues motorisées à l'arrière de diamètre 34 cm et une roue folle à l'avant de diamètre 20 cm.

- Chaque roue motorisée dispose d'un codeur Baumer GCI0K. 0411200 alimenté en 5 V continu pouvant mesurer une vitesse de rotation de 37 500 tr/min.

- Afin de sécuriser le système, sont installés quatre télémètres infrarouges Sharp 2Y3A003 à l'avant et deux Sharp 2YOAO2 à l'arrière du robot pour mesurer la distance entre le robot et les obstacles et éviter tout danger. Les capteurs 2Y3A003 ont une portée théorique de 40 à 300 cm avec un temps de réponse d'environ 16.5 ms. Les deux types de télémètres sont analogiques.

- En guise de cerveau du système, un Arduino MEGA a été choisi pour récupérer les informations provenant des différents capteurs et contrôler la commande des variateurs de vitesse.

- De plus, nous disposons de divers contacts tels que deux relais et un contacteur pour contrôler l'alimentation des différents modules.

De plus, le robot embarquera une unité centrale en guise de serveur et un écran qu'il faut donc alimenter en 12 VDC. Nous profiterons de ce 12V pour alimenter l'Arduino. Il nous faut donc un convertisseur 24 VDC -> 12 VDC disposant d'une puissance et d'un courant de sortie suffisamment important pour alimenter tout cela. D'après la plaque signalétique de l'écran, ses caractéristiques sont les suivantes : 12 VDC, 3.5 A et 42 W. Le processeur i3-3220T de l'ordinateur, à lui seul, nécessite entre 38 et 71 W. Avec ces deux éléments seuls, le convertisseur 24/12 VDC à notre disposition n'est clairement pas suffisant.

Au moment de commencer le projet, le robot est donc en pièces détachées, et seuls les roues et les moteurs sont fixés au châssis. En effet, les variateurs de vitesse, reliés aux moteurs, sont sur une table adjacente au robot avec l'Arduino et les relais utilisés par la personne précédente. Le bouton d'arrêt d'urgence a aussi été retiré et se trouve en pièces distinctes. Quant aux capteurs de distance, trois sont encore fixés à l'avant du robot sur les quatre, et un sur les deux à l'arrière.

I.C Commande de matériel

Afin d'acquérir rapidement le matériel nécessaire au bon déroulement du projet, la question du matériel à commander s'est posée parmi les premières. Si une bonne partie du matériel était déjà à disposition, il restait une partie de l'équipement à acheter.

Dans le but de commander grâce à une manette le Centaure, il était nécessaire d'avoir de quoi réaliser une manette et nous avons donc commandé, pour la concevoir, un joystick. L'intérêt du joystick serait de commander la vitesse du robot au doigt directement et ce de manière progressive, en fonction de l'inclinaison du joystick et de permettre une rotation tout en avançant ou reculant. De plus, afin de transmettre les informations de la manette au robot, il est nécessaire d'avoir un câble suffisamment long. Que nous choisissions un joystick ou bien des boutons, il sera nécessaire d'avoir un câble constitué d'au moins 5 fils d'une longueur de quelques mètres (deux ou trois mètres suffiront amplement). Nous utiliserons donc pour ce projet un câble Ethernet de 5m avec des prises RJ11 aux extrémités, nous permettant d'avoir 6 broches.

Pour commander les variateurs de vitesse à l'aide de l'Arduino, il nous sera nécessaire d'utiliser un convertisseur numérique-analogique. De cette manière, nous pourrons envoyer une certaine tension directement sur le pin 6 du variateur de vitesse sur lequel était branché, d'après la documentation, le potentiomètre pour contrôler les moteurs. Nous choisirons donc les convertisseurs numérique/analogique MCP4725 12-bits pour commander les deux variateurs de vitesse.

Les télémètres ayant été malmenés par les années, en avoir quelques uns de réserve permettrait de couvrir le risque d'avoir des capteurs défectueux parmi ceux installés sur le robot. Il pourrait de plus être intéressant de mesurer l'état de la charge ou SoC (State of Charge en anglais) des batteries afin d'anticiper leur décharge et ainsi préserver leur durée de vie. Pour cela nous investissons dans un 'fuel gauge', mesurant le courant, la charge, la tension et la température pour estimer le pourcentage de décharge des batteries. Le capteur retenu est un LTC2944 qui semble avoir un excellent rapport qualité/prix. Il est compatible avec les batteries au plomb et peut déterminer la charge pour des batteries délivrant jusque 60 V à leurs bornes.

A l'aide d'un voyant lumineux tel qu'une LED RGB, nous mettrons alors au point un code couleur permettant de suivre l'évolution des batteries et d'anticiper leur recharge. Afin de mesurer le couple du moteur, il pourra être nécessaire de mesurer le courant transmis au moteur. En effet, nous savons que :

C = K.i

avec C le couple exercé par le moteur, K une constante liée au courant d'excitation du moteur et i le courant alimentant le moteur. De cette manière, si nous parvenions à connaitre la constante k du moteur, mesurer le courant permettrait de mesurer directement le couple exercé sur les roues.

Enfin, pour alimenter en 12V continu le serveur, l'écran et le microprocesseur, nous commandons un convertisseur 24 VDC -> 12 VDC. Bien qu'il y en ait de toutes sortes, le manque de disponibilité ou de puissance délivrée en sortie du convertisseur a restreint notre choix à un convertisseur permettant de délivrer, en sortie, jusqu'à 288W (24A). Ce convertisseur permettra donc bien d'alimenter tout les appareils nécessaires.

Matériel nécessaire au projet				
Description	*Marque*	*Nb*	*Prix*	*Référence*
bouton ou joystick	TE Connectivity OU SparkFun	4 (boutons) ou 1 (joystick)	2,24 € (paquet de 20 commutateurs) OU 3.36 € (joystick)	Achat sur RS OU Mouser
câble	Decelect Forgos	2,5 m * 5 (4 boutons + vcc) ou 2,5 m de câble ethernet	4,18 € (cordon ethernet (2m)) ou 4,91 € (5m)	RS
résistance		3*2kOhm + 1*50 mOhm		
capacité		1*1µF + 2*1nF		
batterie 12 V / 35 Ah	1er prix confiance de Norauto	2	99.9 € (2*49,95 €)	site web de Norauto
convertisseur PV24S : 24 VDC -> 12 VDC, 288 W et 24 A	ALFATRONIX	1	144 €	sur Farnell
fuel gauge LTC2944 24V	Analog Devices	1	5,28 €	achat sur Mouser
LED RGB 03069		1	0.75€	Achat sur Gotronic
capteur IR 2Y0A02	Sharp	2	14,86 €/unité	RS
capteur de courant	Allegro Microsyste	2	2*3 = 6 €	sur e.banana-pi.fr

ACS712ELCTR-20A-T	ms			
Plaque plexiglas 1*1*0,0025 m^3		1	20,70 €	Achat sur Leroy Merlin
2 convertisseurs numérique vers analogique MCP4725 12-bits	Adafruit	2	2*5 €	Achat sur Adafruit

II. Tests du robot et des variateurs de vitesse

Les deux moteurs disposent en tout de 7 câbles : des gros câbles de couleurs variées (noir, marron, bleu, blanc, vert) et deux câbles plus fins (violet et jaune). Après analyse, le gros câble noir est en fait la réunion de deux câbles noirs plus petits : chacun de ces câbles noirs accompagne un des deux autres câbles fins. Nous avons donc deux paires de câbles de petits diamètres qui sont violet et noir pour le frein du moteur droit et jaune et noir pour le frein gauche. Ces deux freins sont, par défaut, bloqués et il est nécessaire d'appliquer 24 V entre les câbles de chacune de ces paires si nous souhaitons faire tourner les roues du robot. Quant aux plus gros câbles, la paire de câbles marron/bleu permettent la mise sous tension du moteur droit lorsque la paire vert/blanc est liée au moteur gauche. Chacune de ces dernières paires sera donc reliée à l'un des variateurs de vitesse pour faire varier la vitesse de rotation des moteurs et donc la vitesse de déplacement du robot.

Afin de comprendre comment le variateur de puissance CH2QM.2 fonctionne, la première phase de tests sera réalisée sans la PWM de l'Arduino. Nous configurons donc le montage de la même manière qu'indiquée par les documents constructeurs, soit en plaçant une résistance variable de 5 kOhm entre les pins 5 et 7 du variateur de vitesse, le pin 6 étant relié à la troisième broche de la résistance variable. L'Arduino Mega nous permettra de changer l'état des relais afin de simuler un cycle de moteur : moteur à l'arrêt à l'état initial pour respecter les contraintes du variateur, avant d'appliquer une consigne de mouvement avant puis arrière.

Lors de ces tests, il est nécessaire que la valeur du potentiomètre soit de 5 kOhm à l'état initial, avant de faire varier la résistance lorsque le moteur est en phase de mouvement sans quoi les variateurs n'appliquera pas de tension aux bornes des moteurs. De plus, avant de provoquer la mise en mouvement des moteurs, il est très important d'appliquer une tension de 24 VDC aux bornes des freins moteurs sans quoi les freins continueraient d'appliquer une force très importante

sur les moteurs, les empêchant de tourner et provoquant un appel de courant très important risquant de faire chauffer le système si ce n'est le mettre hors service. En prenant toutes ces précautions, nous avons pu mettre en mouvement les deux moteurs en sens avant et en sens arrière.

Une fois le principe de fonctionnement des variateurs de vitesse compris et que nous avons pu faire tourner les deux moteurs, nous effectuons quelques tests sur l'un des moteurs pour être sûrs de la commande à appliquer en entrée des cartes CH2QM.2. Nous relevons, pour différentes valeurs du potentiomètre, les potentiels aux deux bornes du moteur et à celle du potentiomètre par rapport à la masse. Nous mesurons aussi la tension de la batterie pour avoir une valeur référence ainsi que celle aux bornes du moteur. Enfin, nous calculons de manière approximative le nombre de tours par minute que fait une roue pour cette valeur de potentiomètre. Cette valeur approximative vient du fait que, n'ayant pas encore câblé les codeurs, nous mesurons le temps pris par la roue pour faire dix tours par rapport à un repère fixe. Nous réalisons entre 5 et 10 fois cette mesure pour en tirer un temps moyen. C'est de ce temps moyen que nous calculerons le temps mis pour faire un seul tour de roue, et enfin la vitesse angulaire en nombre de tours par minute que le moteur applique à la roue.

Ces valeurs sont ensuite récupérées dans un tableur afin d'analyser l'incidence du potentiomètre que nous cherchons à remplacer sur le système.

TEST VARIATEURS DE VITESSE

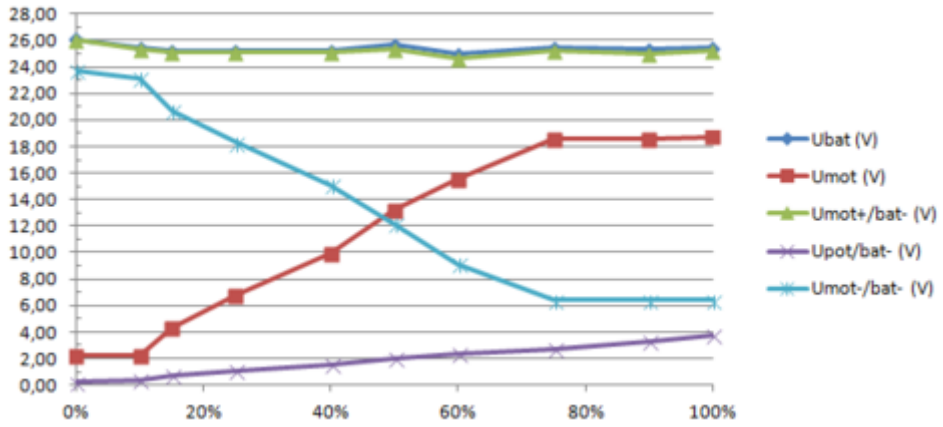
Tension de la batterie (V)	
A vide	24,7
En charge	22,9

Ordre tests
0%->100%->50%->25%->75%->10%->15%->40%->60%->90%

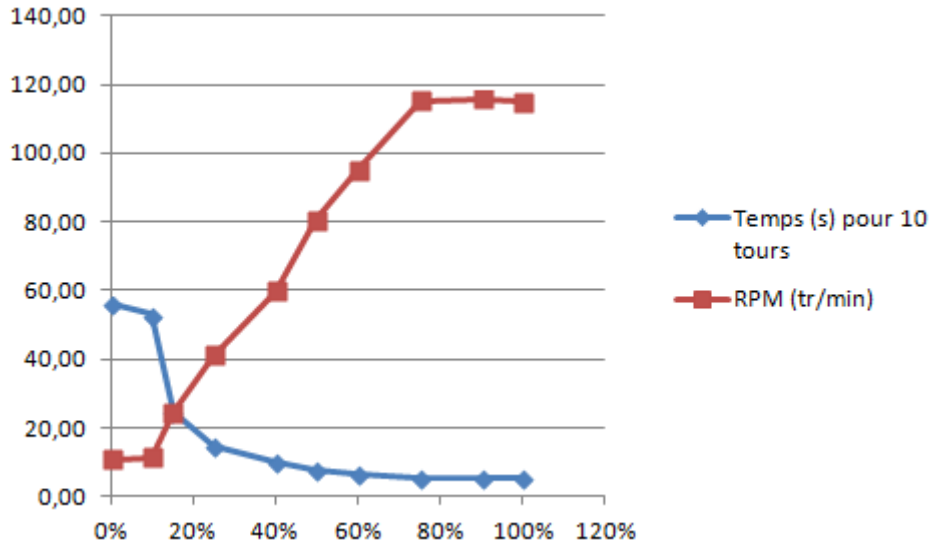
Valeur du potentiomètre 5K	Ubat (V)	Umot (V)	Umot+/bat- (V)	Umot-/bat- (V)	Upot/bat- (V)	Temps (s) pour 10 tours	Temps (s) pour 1 tour	RPM (tr/min)
STOP		0,00			0,17			0,00
0%	26,10	2,20	26,00	23,70	0,19	55,85	5,59	10,74
10%	25,50	2,20	25,40	23,10	0,33	52,91	5,29	11,34
15%	25,20	4,35	25,10	20,70	0,74	24,45	2,44	24,54
25%	25,20	6,81	25,10	18,30	1,08	14,57	1,46	41,19
40%	25,20	9,88	25,10	15,10	1,55	10,03	1,00	59,80
50%	25,70	13,24	25,40	12,20	2,02	7,44	0,74	80,65
60%	25,00	15,55	24,60	9,10	2,37	6,29	0,63	95,36
75%	25,50	18,60	25,20	6,40	2,70	5,21	0,52	115,27
90%	25,30	18,62	25,00	6,40	3,27	5,18	0,52	115,83
100%	25,50	18,74	25,20	6,40	3,72	5,22	0,52	114,92

Tableau des mesures effectuées lors du test de l'influence du potentiomètre sur le système

De ce tableau, nous pouvons constater que le potentiel du pin 6 du variateur de vitesse ne dépasse par les 4V par rapport à la masse. Il devrait donc être possible de retirer le potentiomètre et d'alimenter ce pin grâce à un signal analogique ou PWM issu de l'Arduino. De plus, nous tirons donc de ce tableau l'influence du potentiomètre sur les différentes tensions mesurées et sur la vitesse de rotation. Cette influence est mise en lumière par les deux graphiques suivants :



Graphique montrant l'influence du potentiomètre sur les différentes tensions



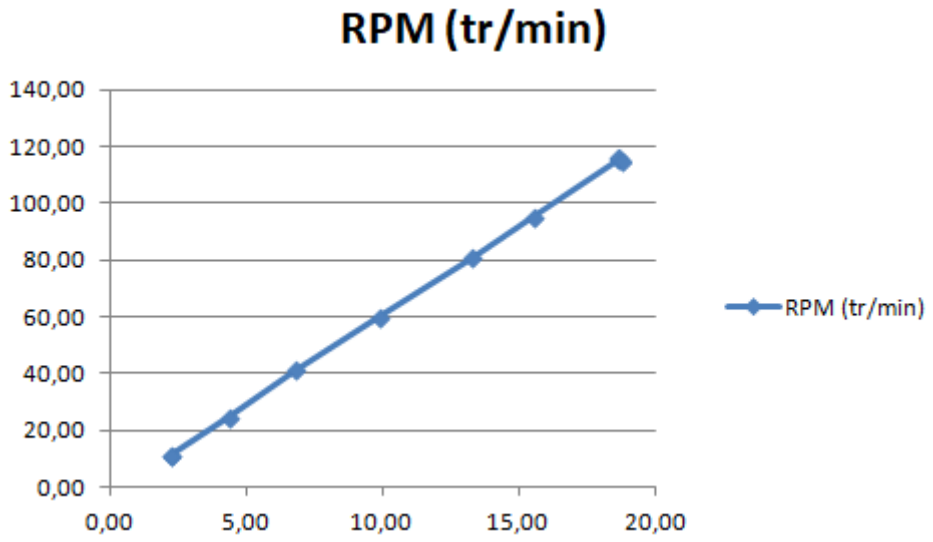
Graphique montrant l'influence du potentiomètre sur la vitesse du moteur en tours par minute

Nous pouvons voir sur ces deux schémas que le potentiel appliqué au pin 6 du variateur de vitesse, proportionnel à la valeur du potentiomètre, est la cause des variations du potentiel situé à la borne '-' du moteur. Cette variation de potentiel provoque donc une tension plus importante entre les deux bornes du moteur qui entraine alors la rotation du moteur. Nous pouvons constater deux paliers : lorsque le potentiomètre est à moins de 10% ou à plus de 75% de sa valeur, autrement dit

que le potentiel à sa borne est inférieur à 350 mV ou supérieur 2,50 V environ, la tension Umot est constante. Hormis ces cas particuliers, Umot semble linéaire en fonction du potentiel Vpot.

Concernant les valeurs obtenues, la plaque signalétique du moteur indique, une vitesse de rotation de 152 tours par minute en sortie du réducteur lorsqu'une tension de 24V est appliquée à ses bornes. De cette manière, une tension de 18,8 V appliquée à ses bornes, soit un rapport de 18,8/24 ~= 0.783 par rapport à la tension nominale, permet une vitesse de rotation de 0.783*152 = 119 tours par minute. Nous retrouvons donc bien presque les 116 tours par minutes mesurés approximativement, avec une marge d'erreur de 2.5%.

Nous pouvons d'ailleurs constater que, tel que le prévoit la théorie, la vitesse de rotation du moteur est proportionnelle à la tension à ses bornes :



Graphique mettant en lumière la relation linéaire entre la tension appliquée par le variateur aux bornes du moteurs et sa vitesse de rotation

Ces valeurs ont été mesurées en réglant le potentiomètre Vmax à son maximum et le potentiomètre Vmin à son minimum, Vmax et Vmin étant deux potentiomètres du variateur de vitesse. A son maximum, Vmax permet d'appliquer une tension maximale plus grande aux bornes du moteur. De même, à son maximum, Vmin permet au variateur d'appliquer une tension minimale plus faible. Cependant, nous avons pu remarquer que, lorsque nous cherchions à faire tourner le moteur à sa vitesse maximale, Vmin avait pour effet de décaler l'ensemble des tensions vers le haut ou le bas.

C'est pourquoi, dans le but d'obtenir la vitesse maximale pour le moteur, nous avons fixé les potentiomètres Vmax et Vmin à leur maximum et minimum respectivement. Sur le tableau suivant, nous mesurons la valeur de la tension appliquée aux bornes du moteur lorsque le potentiomètre est à sa valeur maximale : la colonne de gauche reprend les tensions lorsque le potentiomètre Vmax est à son minimum lorsque la colonne de droite a pour conditions un potentiomètre Vmax à son maximum. Ces deux colonnes sont ensuite divisées en 2 de la même manière en fonction de la position du potentiomètre Vmin.

TENSION MAXIMALE ATTEIGNABLE (V)			
Vmax : min		Vmax : max	
Vmin : min	Vmin : max	Vmin : min	Vmin : max
15,75	14,34	18,79	17,42

Tableau montrant l'influence des potentiomètres Vmin et Vmax du CH2QM.2 sur la tension maximale atteignable aux bornes du moteur

Nous pouvons constater que, contrairement à ce que nous pouvions nous attendre, la tension maximale mesurée aux bornes du moteur est Umot_max = 18,79 V et non 24 V. Ainsi, soit le potentiomètre 5 kOhm vu sur une documentation non référencée n'est pas optimal, soit le variateur de vitesse ne permet pas d'atteindre la tension aux bornes de la batterie. La tension minimale elle, peut osciller entre 0,93 V et 2,3 V continu en fonction de la position du potentiomètre Vmin.

Enfin, il est très important de noter que la tension sur le pin 6 du variateur de vitesse, celui connecté au potentiomètre, nécessite une tension de 165 mV lors du démarrage du système, sans quoi la plaquette se met en mode 'sécurité' et ne permet pas de piloter les moteurs. Il sera donc nécessaire d'appliquer cette tension minimale lors de l'initialisation du programme Arduino avant de l'augmenter pour faire tourner les moteurs, et donc faire bouger le robot.

III. Représentation du système

Dans le but de représenter le système à étudier, nous mettons au point deux forme de schéma :

- un schéma simplifié et visuel, reprenant les photos des différents éléments utilisés reliées par des liens.

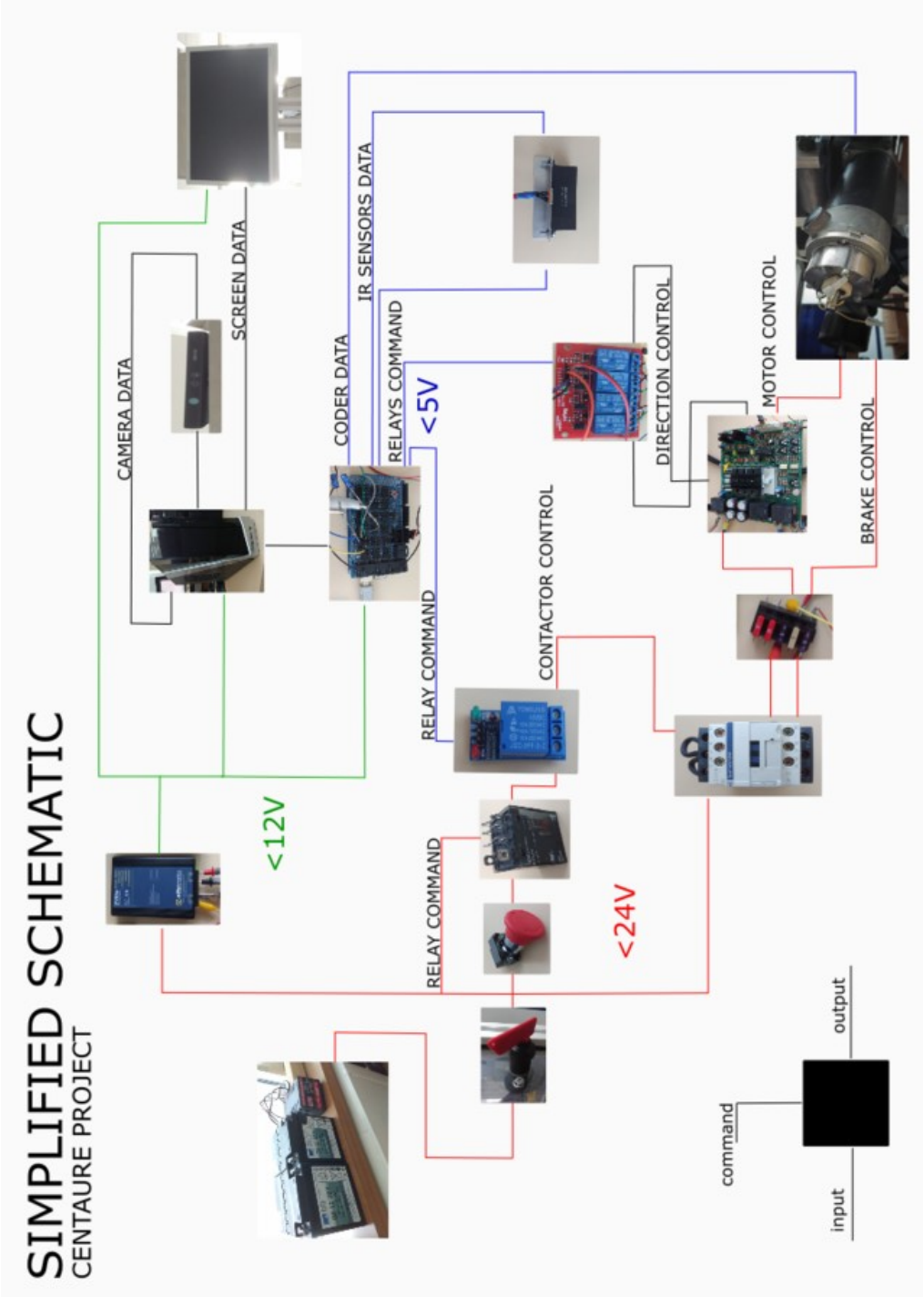


Schéma visuel du projet Centaure

Sur ce schéma, les différents éléments sont reliés à gauche par l'élément les alimentant et par en haut par les éléments lui fournissant des données. Sur leur droite se trouvent ensuite les liens allant vers les modules qu'ils alimentent ou commandent.

Nous pouvons voir que les deux batteries 12 V alimentent tout le système soit de manière directe, soit au travers d'un convertisseur 24 VDC -> 12 VDC. Cette alimentation peut facilement être

commandée manuellement grâce à un coupe batterie situé sur l'extérieur du robot. Suite à ce coupe batterie, nous pouvons distinguer quatre chemins d'alimentation distincts :

- le premier circuit est la partie "cerveau" alimenté en 12V continu. Nous avons donc le convertisseur en premier élément permettant d'alimenter en 12 VDC l'ordinateur, l'écran et l'Arduino MEGA. La caméra Kinect est ensuite connectée au serveur via une de ses sorties USB, lui permettant d'obtenir des informations sur l'environnement du robot. L'écran lui, est aussi relié au serveur pour y afficher des informations. Quant à l'Arduino, elle pilote ensuite différents relais et récupère les informations provenant des capteurs de distance et des codeurs situés derrière les roues. Il est important que cette partie du système ne soit pas en lien avec le bouton d'arrêt d'urgence : de cette manière, un appui sur l'arrêt d'urgence permettra de couper la partie puissance tout en gardant la partie commande. En effet, si l'alimentation du convertisseur était coupée, le serveur et l'Arduino ne seraient plus alimentés, ne permettant plus au système de communiquer sur le web et à l'Arduino de garder la main sur le système de sécurité.
- un second fil relie simplement l'alimentation en 24 VDC à la bobine du relai derrière le bouton d'arrêt d'urgence. Ce relai permet, dans cet état, d'alimenter la bobine du contacteur dés que le coupe batterie est fermé si les autres éléments ne sont pas ouverts. L'intérêt de ce relai réside dans la possibilité de mettre un bouton poussoir entre le 24V et la bobine : il sera alors nécessaire d'appuyer sur un bouton 'départ cycle' une fois la mise sous tension ou le repli en position de sécurité du système pour alimenter le circuit de puissance. Le contact sera maintenu grâce à un auto-maintien du relai. Cette fonction permet alors un gain de sécurité puisqu'il sera nécessaire d'appuyer sur un bouton pour piloter à nouveau le robot. Ce bouton physique peut aussi être remplacé par un bouton sur l'interface web lorsque le robot sera piloté par Wi-Fi pour éviter de se déplacer si le robot est loin. Ce bouton commandera alors le relai piloté par l'Arduino.
- la troisième voie, le circuit de commande, est composée du bouton d'arrêt d'urgence, du relai dont on a précédemment parlé et d'un autre relai piloté par l'Arduino. De la même manière que l'alimentation du circuit de puissance nécessite l'appui sur un bouton 'départ cycle', l'arrêt d'urgence coupe la commande du contacteur s'il a été enclenché et n'est pas revenu en position haute. Le relai piloté par l'Arduino permet à l'Arduino de sécuriser le système de manière logicielle : si un obstacle est détecté trop proche du robot, l'Arduino a la capacité de couper l'alimentation de la partie puissance et ainsi d'immobiliser le système pour qu'il n'entre pas en collision.
- La dernière ligne concerne la partie puissance du système. Si le contacteur est fermé grâce à la partie commande, alors il alimente les variateurs de vitesse et les freins des moteurs au travers de fusibles. Le variateur de vitesse, commandé par l'Arduino, va ensuite alimenter le moteur auquel il est rattaché. L'Arduino commande, grâce à des relais et à un signal analogique, la direction et la vitesse des moteurs. Il es très important que les freins des moteurs soient alimentés en 24V sans quoi le courant demandé à la batterie risquerait d'être très important et d'endommager le système lors de la commande des moteurs. Enfin, un codeur placé derrière chaque roue permet à l'Arduino d'obtenir des informations sur la vitesse de rotation du moteur, et donc d'adapter sa commande par rapport à la consigne reçue de l'utilisateur.

/!\ Non représenté sur ce schéma, un fusible en début de chaque lignes permettrait d'éviter tout problème électrique en cas de court-circuit.

- deux schémas électriques plus conventionnels :

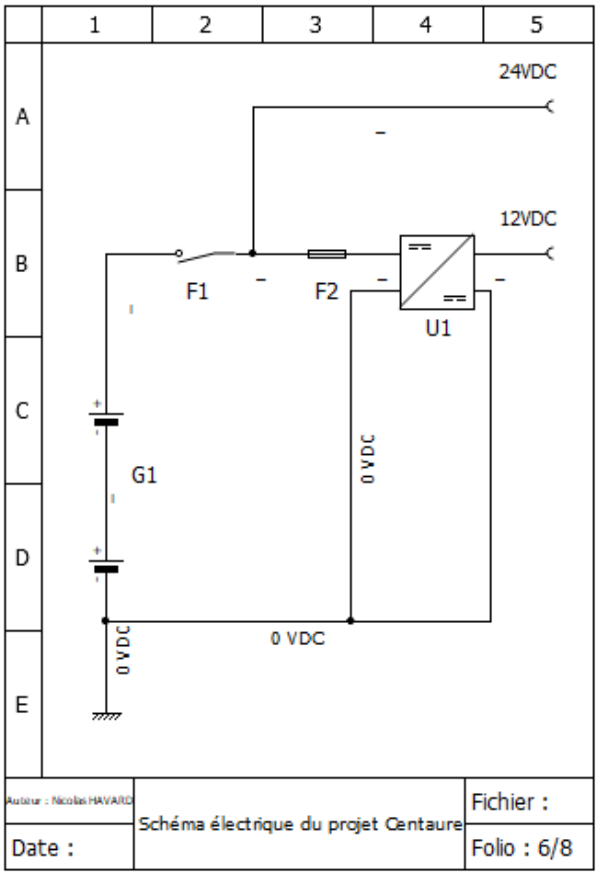


Schéma électrique du projet Centaure : partie 1

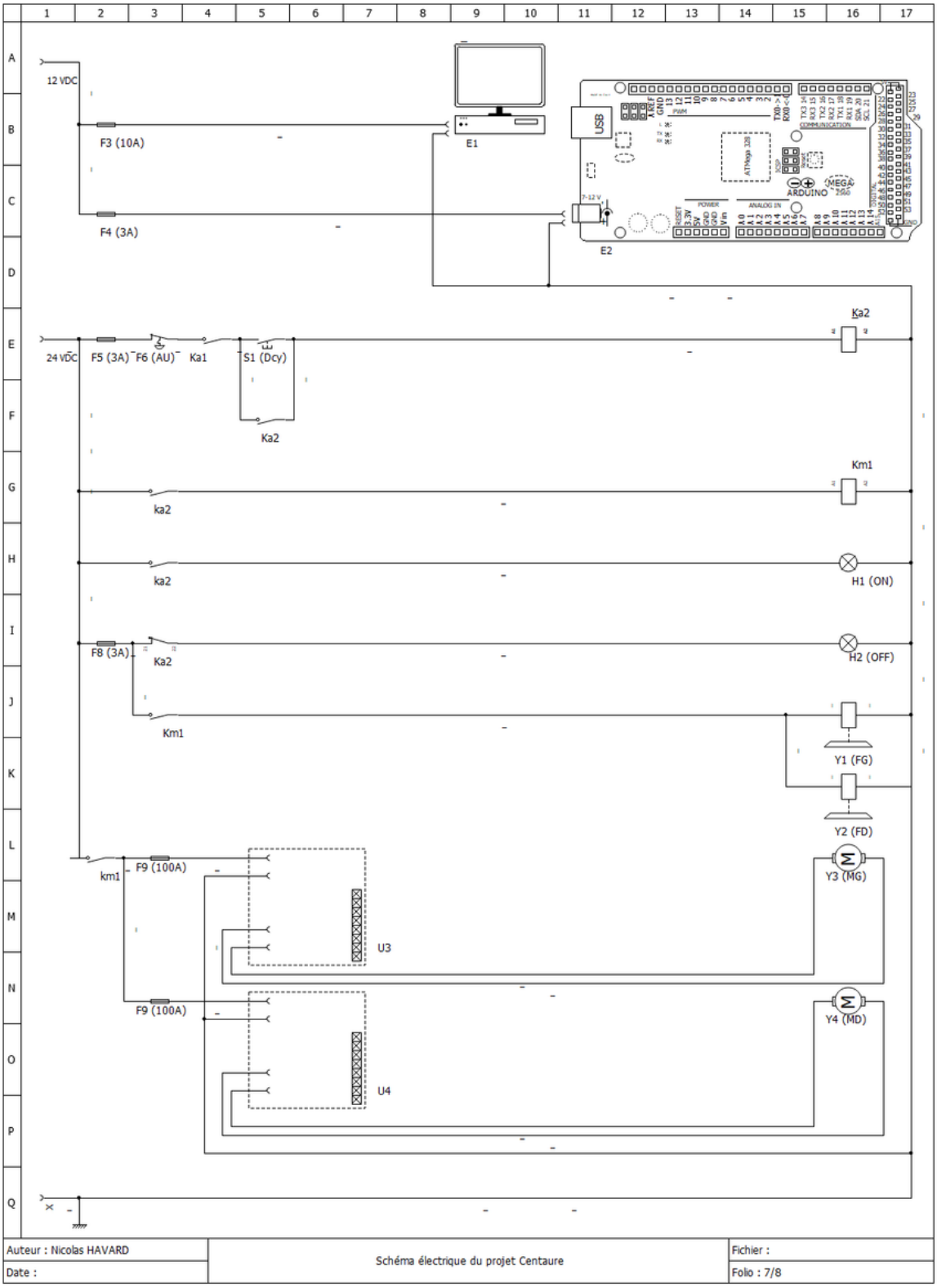


Schéma électrique du projet Centaure : partie 2

Sur le premier schéma, nous retrouvons la batterie, le coupe-batterie et le convertisseur 24 VDC->12 VDC. Les sorties de ce folio sont donc les alimentations en 12 V et 24 V continues. La suite

du système se trouve alors sur le second schéma de la même manière que nous l'avions vu sur le schéma précédent :

-
  - l'alimentation en 12 V permet d'alimenter le PC/l'écran ainsi que l'Arduino. Ne sont pas présents sur ce schéma les connexions liées à l'Arduino tels que les différents capteurs et la connexion série avec l'ordinateur : ces connexions seront l'œuvre d'un prochain folio lorsqu'il sera définitif.
  - la partie commande, alimentée en 24 VDC et composée du bouton d'arrêt d'urgence et des relais vu sur le schéma simplifié. Nous avons rajouté sur ce schéma électrique le bouton Dcy (S1) hypothétique. Il pourrait très bien être remplacé par un câble seul dans un premier temps. Au bout de cette ligne de commande ce trouvent les bobines du relai et du contacteur ainsi qu'un voyant lumineux H1 indiquant que le robot est actuellement sous puissance. Lorsqu'une des conditions n'est pas satisfaite (i.e suite à l'appui sur le bouton d'arrêt d'urgence ou qu'un des contacts des relais n'est pas fermé) alors le contact normalement fermé du relai 24V laisse passer le courant et alimente le voyant H2 signifiant la mise sous tension du robot bien qu'il ne soit pas sous puissance. De cette manière, un voyant est toujours allumé dés que le coupe-batterie est fermé afin de montrer que le robot est sous tension.
  - la partie puissance débute par le contacteur qui, s'il est fermé, permet l'alimentation des variateurs de vitesse et des freins.

IV. Conception de cartes électroniques

Maintenant que la grosse électronique est comprise, il nous est nécessaire de concevoir des cartes électroniques pour notre projet. En effet, nous disposons de quelques LEDs et résistances qu'il serait bon de fixer à un PCB. Ce PCB nous permettra, de la même manière, de fixer les différents fils tels que ceux liés aux codeurs, aux variateurs de vitesses et à la commande. Concernant la commande, nous avons choisi de piloter le Centaure à l'aide d'un joystick, il nous faut donc fixer ce dernier à un PCB et préparer la connexion avec la carte principale. Pour assurer cette communication, et puisque 5 pins sont nécessaires pour se servir du joystick, nous utiliserons un câble Ethernet avec deux prises RJ11 à 6 broches. Ainsi, il nous restera un pin non utilisé qui pourrait nous servir à l'avenir.

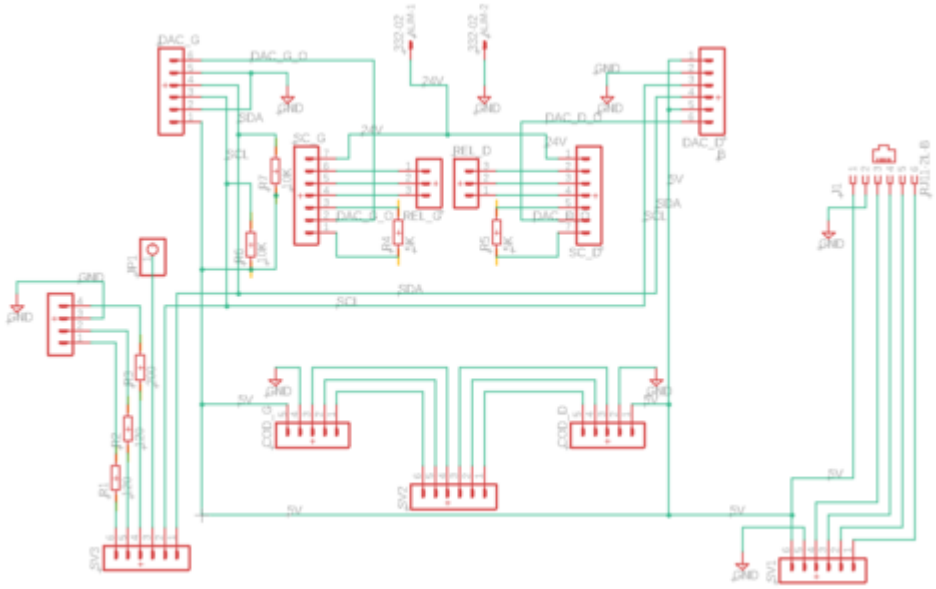
Afin de concevoir ces deux cartes, nous avons utilisé la version étudiante du logiciel Eagle. Dans un premier temps, il est nécessaire de concevoir le 'schématic' des cartes en récupérant ceux de chaque composant, ainsi que leur 'footprint' que nous utiliserons par la suite. Ces 'schématics' permettent d'indiquer le fonctionnement, le câblage du circuit pour les personnes souhaitant obtenir des informations sur la carte mais aussi, et surtout, de préparer le routage de la carte.

Concernant le routage, l'espace entre deux éléments (entre deux pistes, une piste et un pad, etc) ainsi que l'isolation ont été fixé à 12 mil (environ 0.3 mm) pour les deux cartes. La taille des pistes, elle, est de 24 mil (0.6 mm) sur les deux cartes à l'exception des pistes liées aux pins des prises RJ11 qui ont été réduites à 20 mil (0.5 mm) à cause du faible espace entre les pins des connecteurs.

Nous allons donc présenter la conception des deux cartes de manière distincte.

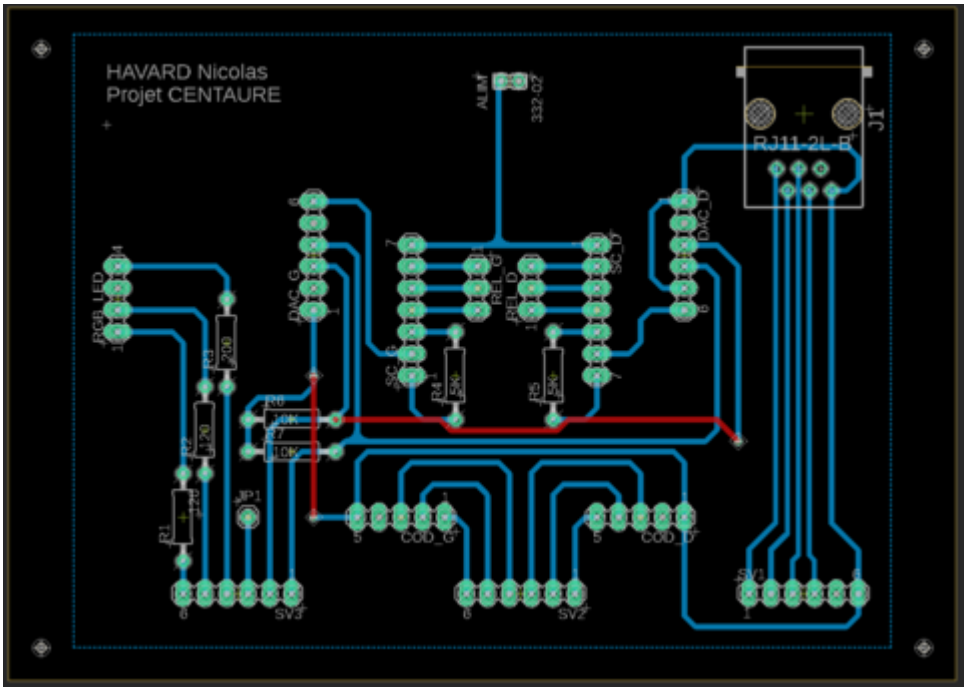
Une carte interfaces

La carte principale, devant servir d'interfaces entre l'Arduino, les codeurs incrémentaux et les différentes cartes déjà conçues, est majoritairement constituée de 'pin headers'. En effet, il nous est nécessaire d'avoir 17 pins pour l'Arduino (5V, GND, SDA, SCL, 11 I/O numériques et 2 I/O analogiques), 10 pins pour les codeurs (5 pins par codeur), 12 pins pour les convertisseurs numérique vers analogique (6 par DAC), 14 pins pour les variateurs de vitesse (7 par plaquette), 6 pour les relais, et 2 pour l'alimentation 24 VDC et la masse de la batterie. Se trouvent aussi sur cette carte électronique une LED RGB et ses résistances, les résistances de 5 kOhm nécessaires au bon fonctionnement des variateurs de vitesse, ainsi qu'un pin qui a été ajouté en cas de besoin. De plus, deux résistances de 10 kOhm ont été ajoutée en pull-up des signaux SDA et SCL pour la communication I²C sur conseils de M. Boé.



Schematic de la carte servant d'interface entre l'Arduino et les autres modules

Concernant le routage, nous avons essayé de garder une carte en simple couche. Cela a été possible mais nécessite deux vias permettant de relier, au travers de fils, le signal SCL entre les deux convertisseurs numérique vers analogique et le 5V entre les headers pour le codeur gauche et le convertisseur numérique/analogique gauche, indiqué en rouge sur la photo suivante :



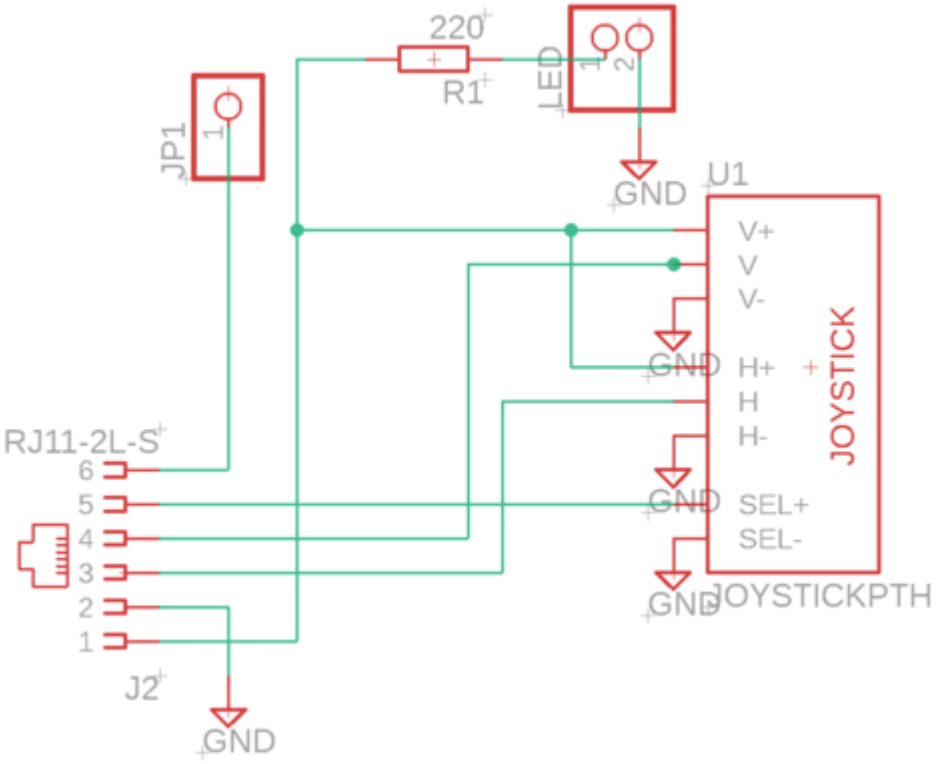
Routage de la carte servant d'interface entre l'Arduino et les autres modules

Critiques : Cette carte fait preuve de nombreux défauts de conception. En effet, concernant les relais couplés aux variateurs de vitesse, ceux-ci ne sont pas gérés par cette carte. Seuls 3 pins leur sont dédiés conformément à la documentation des variateurs de vitesse et nous n'avons donc pas pris en compte la gestion des relais sur cette carte : des fils devront relier directement les bobines des relais à l'Arduino pour être pilotées

De plus, la taille des convertisseurs numérique/analogique n'a pas été prise en compte, et celui de droite se retrouve ainsi bien trop proche de la prise RJ11, malgré la place disponible sur la carte. Enfin, la conception de cette carte s'est basée sur la documentation des modules et n'a donc pas pris en compte les liens entre les modules provoqués par les équipes ayant travaillé sur ce projet précédemment tels que des fils des variateurs soudés entre eux ou encore sur les contacts des relais.

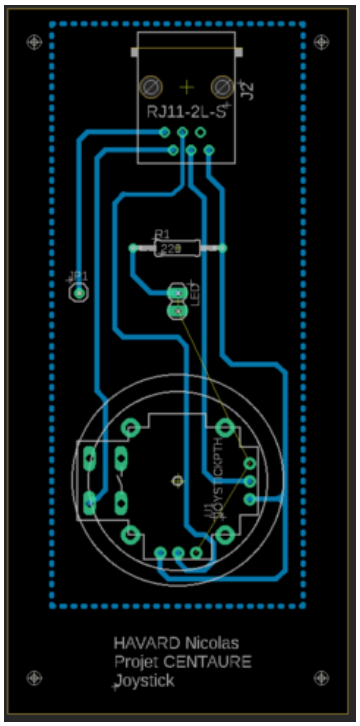
Une manette

Concernant la manette, sa carte fut simple à concevoir. En effet, celle-ci est constituée du joystick et de la prise RJ11 pour envoyer les données facilement à la carte principale. Etant donné que cette prise disposait de 6 broches, nous avons tiré une piste supplémentaire, reliée à un pin pouvant servir à l'avenir. Afin d'indiquer facilement à l'utilisateur que la manette est bien connectée, nous avons aussi rajouté une LED et sa résistance directement connectées sur la broche Vcc alimentant la carte.



Schematic de la manette

A l'image du schematic, le routage de cette carte a été relativement aisé et est contenu sur une seule face du PCB.



Routage de la manette

Critiques : Le plan de masse a été isolé au milieu de la carte, et il a été nécessaire de souder un fil reliant les deux plans de masse pour que la carte fonctionne correctement. Dans le cas d'une réédition, ce problème pourrait facilement être corrigé en vérifiant l'espace entre les pins et en routant différemment la carte.

Critiques générales : Concernant les cartes de manière générale, la taille des pads et celle des vias ont été réduites au strict minimum. Afin de faciliter le soudage, il pourrait être intéressant d'agrandir ces tailles légèrement. Concernant la connexion entre l'Arduino et le joystick, si le câble vient à être débranché, l'Arduino ne reçoit plus la tension correspondant à la position d'origine du joystick, et le robot est donc bloqué en marche avant, ce qui est assurément un problème de sécurité non négligeable. Le plus simple, d'après M. Boé, serait de forcer une tension de 2.5 V en cas de problème, coupée par un transistor FET canal P dont la grille est commandée par la tension du joystick : quand elle est déconnectée, le transistor devient passant et alimente en 2.5 V.

V. Programmation des différents modules

L'Arduino MEGA étant connectée à deux convertisseurs numérique vers analogique, aux capteurs infrarouges, aux codeurs, aux relais et au joystick, il va nous falloir développer un programme permettant de gérer ces différents modules et les adapter pour qu'ils fonctionnent ensemble selon la logique suivante :

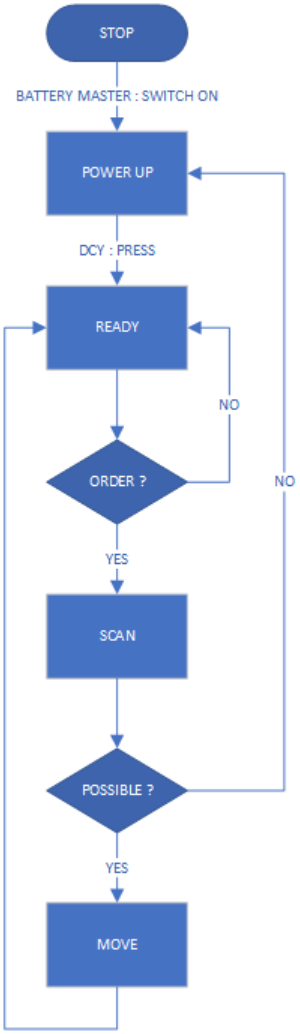


Schéma illustrant la logique de programmation que devra suivre le robot

Nous utilisons l'IDE d'Arduino et des bibliothèques constructeurs lorsqu'elles sont disponibles pour concevoir le programme du robot. Afin de tester les différents modules, nous testons les programmes de manière isolée dans un premier temps.

Convertisseur Numérique vers Analogique (ou Digital to Analog Converters) :

Nous utilisons deux convertisseurs MCP4725 12-bits pour commander les variateurs de vitesse par le biais de l'Arduino et donc se passer d'une commande manuelle. Le principe de fonctionnement de ces convertisseurs est assez simple : la carte va appliquer une tension sur le pin Vout comprise entre 0V et la différence de tension entre VCC et GND dépendamment de l'information reçue sur 12 bits via le protocole I²C (**I**nter-**I**ntegrated **C**ircuit). Dans notre cas, nous relions les broches VCC et GND du convertisseurs aux broches 5V et GND de l'Arduino MEGA. Afin de permettre le démarrage du contrôle des moteurs par les variateurs de vitesse, il était nécessaire

d'appliquer une tension d'environ 165 mV sur le pin 6 du variateur de vitesse. Après une mesure de tension du pin 5V de l'Arduino, il s'avère que la tension est exactement de 5.15 V. 165 mV correspondant à 3.2% des 5.15 V délivrés par l'Arduino, l'information envoyée au convertisseur doit donc correspondre à 3.2% de la valeur maximale prise en compte par le convertisseur, soit 3.2% de 4095 (2^12 valant 4096). L'Arduino devra donc envoyer '131' comme information lors du paramétrage du convertisseur au démarrage du robot grâce à la méthode

dac.setVoltage(131, true) ;
// voltage is 131/4095 * 5.15 = 0.165 V, required to pass speed controllers' checking

qui permet d'envoyer l'information sur le bus I²C (à conditions d'avoir chargé la librairie *<Wire.h>* en plus de celle d'Adafruit propre au convertisseur)). Le flag *true*, dans l'exemple, permet de charger cette valeur dans l'EEPROM comme valeur par défaut, intéressant ici puisque cette tension doit être appliquée sur le pin du variateur de vitesse au démarrage du système.

NB : Les masses des convertisseurs (et donc de l'Arduino) et celles des variateurs de vitesse doivent absolument être les mêmes sans quoi les 165 mV émis par les convertisseurs ne seront jamais perçus comme tel par les variateurs de vitesse. Le bornier n'ayant pas de pins 'GND' sur le variateur de vitesse, la masse de l'Arduino devra donc être liée à la borne moins de la batterie sur laquelle sont branchés les variateurs.

Programme test du convertisseur numérique vers analogique :

```
#include <Wire.h>
#include "Adafruit_MCP4725.h"

Adafruit_MCP4725 dac_motorD ;
Adafruit_MCP4725 dac_motorG ;


void setup(void) {
  Serial.begin(9600);
  Serial.println("DAC test :");


  // For Adafruit MCP4725A1 the address is 0x62 (default) or 0x63 (ADDR pin tied to VCC)
  // For MCP4725A0 the address is 0x60 or 0x61
```

```
  // For MCP4725A2 the address is 0x64 or 0x65
  dac_motorD.begin(0x62);                    // A0 -> GND
  dac_motorG.begin(0x63);                    // A0 -> Vin

  dac_motorD.setVoltage(131, true) ;                    // voltage is 131/4095 * 5.15 = 0.165 V,
required to pass speed controllers' checking
  dac_motorG.setVoltage(131, true) ;                // flag 'true' allows Arduino writing this value
in the EEPROM of the MCP4725 : this value will be the default value sent by the MCP4725 on restarts


}

void loop(void)
{
  //dac_motorD.setVoltage(2048, false);
  delay(2000) ;
}
```

Dans cet exemple, nous avons commenté la ligne nous permettant de paramétrer le DAC pour qu'il envoie 50% des 5.15 V afin de tester si le variateur de vitesse recevait correctement les 165 mV.

Joystick

La consigne de mouvement du robot étant donnée par la position d'un joystick, il est primordial de récupérer la position horizontale et verticale du joystick. Cette position sera récupérée par deux pins analogiques de l'Arduino grâce aux résistances variables du joystick. En effet, le joystick est composé d'une résistance variable pour la position horizontale et d'une autre pour la position verticale. Cette position, initialement entre 0 et 1023, est ensuite multipliée par un coefficient pour le moteur droit et un autre pour le moteur gauche. Ces deux coefficients dépendent de la position horizontale du joystick et permettent au robot de tourner. Ces deux vitesses, propres à chacun des moteurs, sont ensuite ramenées sur une échelle allant de 0 à la vitesse maximale du robot, qui correspondra dans notre cas aux 3V que peut envoyer les convertisseurs numérique-analogique aux variateurs de vitesse.

Afin de représenter la consigne transmise au moteur, nous réalisons un schéma indiquant, en fonction de la position du joystick, la consigne en puissance émise :

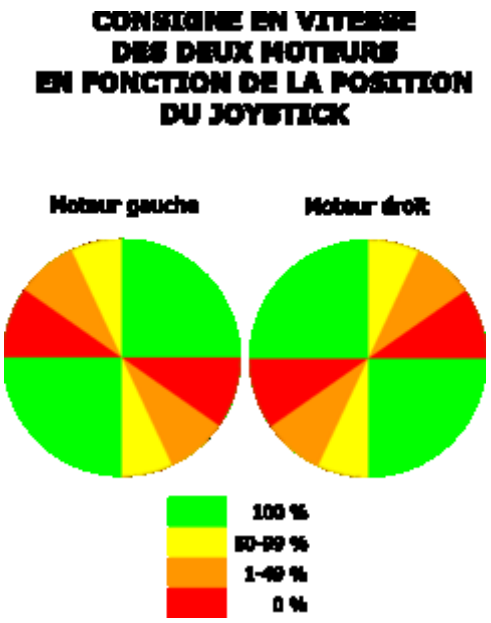


Schéma simplifié représentant la vitesse de consigne des deux moteurs en fonction de la position du joystick

Sur ce schéma, la couleur verte indique que le moteur tournera à vitesse nominale, la couleur rouge indiquera que le moteur est arrêté tandis que les couleurs jaune et orange indiquent des consignes en vitesse intermédiaires.

Programme du joystick :

```
#define JOYSTICK_X A6 // X -> droite / gauche
#define JOYSTICK_Y A7 // Y -> vitesse / direction
#define JOYSTICK_THRESHOLD 20
#define MAX_SPEED 2147          //
int calX, calY, vitesse, direction ;
float coefG, coefD ;
#define MOTEUR_STOP 0
#define MOTEUR_AVANCE 1
#define MOTEUR_RECULE 2

void setup()
{
  //joystick_INIT_() ;
  calX = analogRead(JOYSTICK_X);
```

```
  calY = analogRead(JOYSTICK_Y);
}

void loop()
{
  joystick_getData(calX, calY, &coefG, &coefD, &vitesse, &direction);

  // Affichage de la vitesse du moteur gauche
  Serial.print("Moteur Gauche : ");
  Serial.println((int)(vitesse * coefG));
  vitesse_gauche =vitesse * coefG ;

  // Affichage de la vitesse du moteur droit
  Serial.print("Moteur Droit : ");
  Serial.println((int)(vitesse * coefD));
  vitesse_droit = vitesse * coefD ;

  // Affichage de la direction des moteurs
  //Serial.print("Direction : ");
  if(direction == MOTEUR_AVANCE) // Vers l'avant
  {
    //Serial.println("AVANT");
    digitalWrite(led_g, HIGH) ;
    digitalWrite(led_r, LOW) ;
    digitalWrite(led_b, LOW) ;
    avance_gauche(vitesse_gauche) ;
    avance_droit(vitesse_droit) ;
  }
  else if(direction == MOTEUR_RECULE) // Vers l'arriére
  {
    //Serial.println("ARRIERE");
    digitalWrite(led_r, HIGH) ;
    digitalWrite(led_g, LOW) ;
    digitalWrite(led_b, LOW) ;
    recule_gauche(vitesse_gauche) ;
    recule_droit(vitesse_droit) ;
  }
  else
  {
    //Serial.println("ARRET"); // Aucun mouvement
    digitalWrite(led_b, HIGH) ;
    digitalWrite(led_r, LOW) ;
    digitalWrite(led_g, LOW) ;

    digitalWrite(PINAVMOTD, LOW) ;
    digitalWrite(PINARMOTD, LOW) ;
    digitalWrite(PINAVMOTG, LOW) ;
    digitalWrite(PINARMOTG, LOW) ;
  }
```

```
 // Delai de 500ms pour pouvoir lire la console
 delay(750);


joystick_getData(calX, calY, &coefG, &coefD, &vitesse, &direction)
  // rawX : valeur brute en X du joystick centrée sur calX
  // rawY : valeur brute en Y du joystick centrée sur calY
  int rawX, rawY;

  // Mesure des valeurs brute en X et Y
  rawX = -(analogRead(JOYSTICK_X) - calX);
  Serial.print("valeur X : ") ; Serial.println(rawX) ;
  rawY = -(analogRead(JOYSTICK_Y) - calY);
  Serial.print("valeur Y : ") ; Serial.println(rawY) ;

  // Si -THRESHOLD < rawY < THRESHOLD
  if(rawY > -JOYSTICK_THRESHOLD && rawY < JOYSTICK_THRESHOLD)
  {
    // Les moteurs sont marqués comme arrêtés, et vitesse = 0
    *direction = MOTEUR_STOP;
    *vitesse = 0;
  }
  // Si rawY >= 0
  else if(rawY >= 0)
  {
    // Les moteurs sont marqués en mode "avance"
    *direction = MOTEUR_AVANCE;

    // La vitesse est égale à map(rawY) depuis 0 ~ (1023 - calY) vers 0 ~ MAX_SPEED
    *vitesse = map(rawY, 0, 1023 - calY, 0, MAX_SPEED);
  }
  // Si rawY < 0
  else
  {
    // Les moteurs sont marqués en mode "recule"
    *direction = MOTEUR_RECULE;
    // La vitesse est égale à map(rawY) depuis 0 ~ calY vers 0 ~ MAX_SPEED
    *vitesse = map(-rawY, 0, calY, 0, MAX_SPEED);
  }

  // Si rawX < -THRESHOLD alors coefG = -rawX / calX sinon coefG = 1
  *coefG = (rawX < -JOYSTICK_THRESHOLD) ? (1023/2+rawX)/(float)calX : 1;

  // Si rawX > THRESHOLD alors coefD = rawX / calX sinon coefD = 1
  *coefD = (rawX > JOYSTICK_THRESHOLD) ? (1023/2-rawX)/(float)calX : 1;
}
```

Codeurs incrémentaux

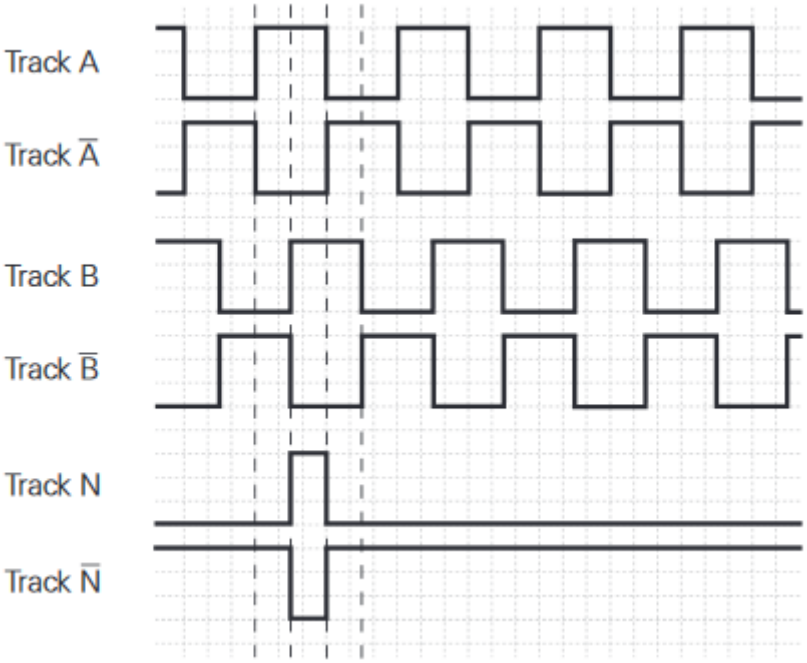
Afin d'asservir le robot en vitesse et vérifier que celui-ci suit bien une ligne droite lorsque nous lui ordonnons d'aller tout droit, nous devons récupérer des informations sur ses déplacements. Pour cela, nous récupérons les impulsions émises par les codeurs situés derrière les roues du robot. Ces codeurs, des GCI0K. 0411200 de Baumer, sont constitués de deux parties : une partie fixe placée sur le châssis du robot et relié grâce à 8 câbles permettant d'envoyer les informations sur la position des roues, et une roue placée derrière la roue du robot tournant donc à la même vitesse que cette dernière.

Ainsi, lorsque la roue du robot tourne, elle entraine cette petite roue placée sur son axe de rotation et qui permet à la partie fixée sur le châssis d'envoyer un signal au microcontrôleur indiquant la position de la roue et donc, en mesurant le temps entre deux positions connues, sa vitesse.

Mais comment ces codeurs permettent-ils de déterminer la position de la roue ?

En réalité, la roue placée sur l'axe de rotation de la roue du robot est constituée d'une multitude d'aimants à polarité opposée à ses voisins. Lorsque la roue tourne, la partie fixe voit donc défiler des pôles nord et sud successivement, provoquant un champ magnétique variant. A titre d'exemple, la partie fixe va donc émettre un signal carré compris entre 0V lorsque l'aimant devant le détecteur est un pôle sud et 5V lorsque l'aimant est un pôle nord. Le microcontrôleur peut donc être alerté de chaque changement de position angulaire de la roue du robot et, en connaissant le diamètre de cette roue, déterminer la distance parcourue et la vitesse du robot.

Cependant, la précision de ce mécanisme étant donc directement reliée au nombre de pôles de la roue, et celui-ci ne pouvant évidemment pas être important, ce genre de codeur double le nombre de pistes afin d'avoir, par exemple, deux tours constitués de 64 aimants sur sa périphérie. L'intérêt est qu'en doublant le nombre de pistes, de signaux, et en les déphasant d'un quart de période, nous multiplions par deux la précision puisqu'au lieu d'avoir seulement des 0 et des 1 pour deux aimants, nous pourrons avoir 00, 01, 11 et 10 pour le même nombre d'aimant et donc le même angle. Ceci requiert cependant de consulter l'état des deux signaux ce qui peut prendre un peu plus de temps.



Signaux des codeurs

Les codeurs utilisés comportent deux voies appelées A et B. Un troisième, la voie N, permet de savoir si la roue a fait un tour complet et donc de resynchroniser les compteurs s'il le faut. De plus, nous avons à notre disposition les signaux contraires de ces 3 voies : non A, non B et non N, qui servent à détecter de potentielles erreurs et donc à gagner en précision. Nous nous en servirons pas ici. Afin de suivre chaque changement, nous connectons les différentes voies des deux codeurs à des interruptions externes de l'Arduino MEGA. En effet, cette dernière possède 6 interruptions externes, numérotées de 0 à 5 et correspondant aux pins 2, 3, 21, 20, 19 et 18 respectivement.

Programme test des codeurs incrémentaux :

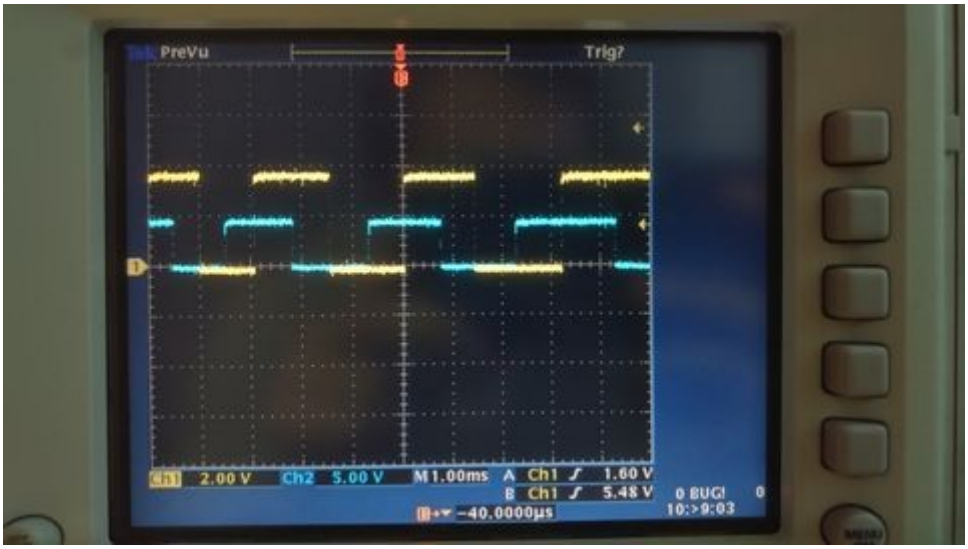
```
#define pinA  0  // interruption 0 correspond au pin 2
#define pinB  1  // interruption 1 correspond au pin 3
#define pinN  2  // interruption 2 correspond au pin 21


byte state = 0x00 ;
int compteurA = 0;
int compteurTot = 0;

void interruptA()
{
  //state = 0x01 ;
  compteurA = compteurA+1 ;
  compteurTot = compteurTot+1 ;
}

void interruptB()
{
```

```
  //state = 0x02 ;
  compteurB = compteurB+1 ;
  compteurTot = compteurTot+1 ;
}

void interruptN()
{
  //state = 0x03 ;
  compteurN = compteurN+1 ;
}


void setup()
{
  Serial.begin(115200) ;
  Serial.println("TEST CODEUR INCREMENTAL") ;
  init_motor() ;
  //pinMode(21, INPUT_PULLUP);
  //pinMode(2, INPUT_PULLUP);
  //pinMode(pinN, INPUT_PULLUP);

  /* accroche les ISR aux pins */
  attachInterrupt(pinA, interruptA, RISING);      // Interruption sur front montant
  attachInterrupt(pinB, interruptB, CHANGE);      // Interruption sur front montant et descendant
  attachInterrupt(pinN, interruptN, CHANGE);

}

void loop()
{
  avance_gauche(50);
  avance_droit(50) ;

  Serial.print("Voie A : ");
  Serial.println(compteurA) ;
  Serial.print("Voie B : ");
  Serial.println(compteurB) ;
  Serial.print("Changements : ");
  Serial.println(compteurTot) ;
  Serial.print("Voie N : ");
  Serial.println(compteurN) ;
  Serial.println() ;

  delay(1000) ;
}
```

Ce code exemple ne sert qu'à incrémenter des compteurs à chaque fois qu'une interruption est détectée. Il a permis de valider le fait que les voies A et B disposaient chacune de 1024 impulsions, soit 1024 fronts haut et 1024 fronts bas chacun. En se contentant de suivre seulement les fronts haut de la voie A, cela nous permet donc une précision de 0.35° sur la roue, ce qui est

amplement suffisant. La voie B nous servira donc à analyser le mouvement de la roue, et de voir si le robot avance ou recule : en effet, lors d'un front haut sur la voie A, si la voie B est à l'état bas, alors le robot avance. A l'inverse, si elle est à l'état haut, cela signifie que le robot recule.

Critiques : Ayant eu des problèmes avec les voies N, nous avons analysé les signaux émis par les codeurs incrémentaux. Si nous n'avions pas de problème à détecter les voies A et B, de forme carrée et d'amplitude 4V environ, il a été impossible de visualiser l'impulsion de la voie N lorsqu'un tour est fait. Il est possible qu'il y ait donc un problème de branchement avec cette voie.



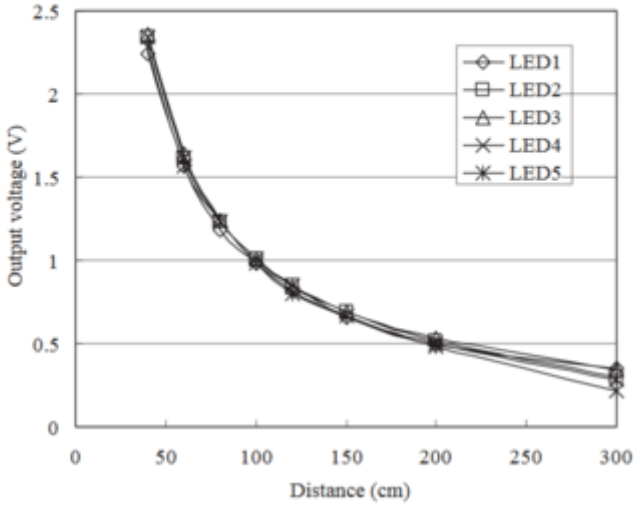
Signaux des voies A (en jaune) et B (en bleu) visualisés à l'oscilloscope

Télémètres infrarouges

Pour les capteurs infrarouge permettant de détecter les obstacles devant et derrière le robot, nous avons procédé de différentes manières : à l'aide d'un simple relevé de la valeur analogique grâce à une commande Arduino, et via des fonctions écrites dans des bibliothèques dédiées aux capteurs Sharp. Comme nous pouvons le voir d'après la documentation du constructeur, la caractéristique tension en fonction de la distance mesurée croit fortement jusqu'à une distance relativement proche (quelques dizaines de centimètres) avant de décroitre lentement jusqu'à une distance de plusieurs mètres :

SHARP

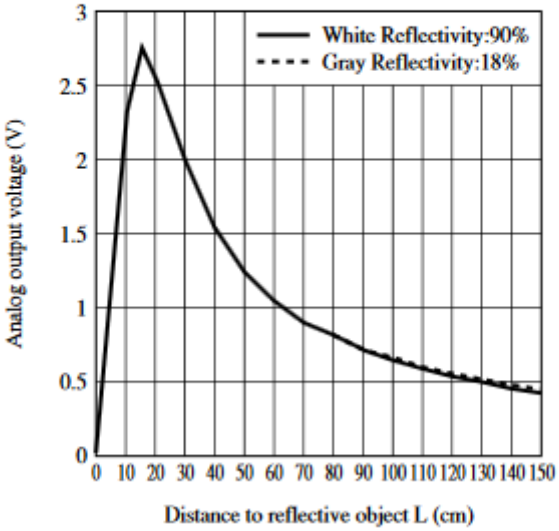
Fig. 2 Example of distance measuring characteristics (output)



Caractéristique de la distance en fonction de la tension émise par le capteur GP2Y3A003K0F, placé à l'avant du robot

SHARP

Fig.3 Analog Output Voltage vs. Distance to Reflective Object



Caractéristique de la distance en fonction de la tension émise par le capteur GP2Y0A02YK, placé à l'arrière du robot

Les capteurs Sharp étant destinés à mesurer des objets à distance moyenne voire longue, nous nous intéresserons aux parties décroissantes des courbes : celles-ci sont continues et strictement décroissante. Afin d'être sûr qu'un objet n'a pas été détecté trop proche du robot, il suffira donc de choisir un seuil de tension pour lequel le robot entre en mode sécurité tel que 2V. De cette manière, si un objet se rapproche à moins de 30 cm ou 50 cm, suivant le capteur utilisé, l'Arduino va détecter un tension supérieure à ce seuil et alimenter le relai permettant d'ouvrir le circuit de commande du contacteur qui alimente les moteurs.

N'ayant cependant pas eu de résultat concluant au travers de ces deux méthodes, il serait nécessaire de revenir sur cette partie à l'avenir.

Critiques : Ces capteurs sont conçus pour détecter des objets à distance relativement longue et sont inexploitables à courte distance. Afin d'être sûr qu'un objet n'est pas à moins de 50 cm à l'avant et de 20 cm à l'arrière (distance correspondant aux pics de tension en fonction des capteurs utilisés), il serait bon de coupler ces capteurs à un capteur à ultrason tel qu'un HC-SR04.

VI. Test du système

Une fois le programme conçu, nous connectons les cartes électroniques à l'Arduino, branchons les variateurs de vitesse aux batteries et aux moteurs puis testons l'algorithme. Pour cela, nous fermons le coupe-batterie et branchons l'Arduino au PC afin de récupérer les données lors du test. Nous attendons ensuite l'initialisation des différents modules avant d'envoyer un ordre de mouvement au robot : une fois la LED de la carte principale allumée en bleue, il nous est alors possible d'envoyer une commande au robot à l'aide de la manette.

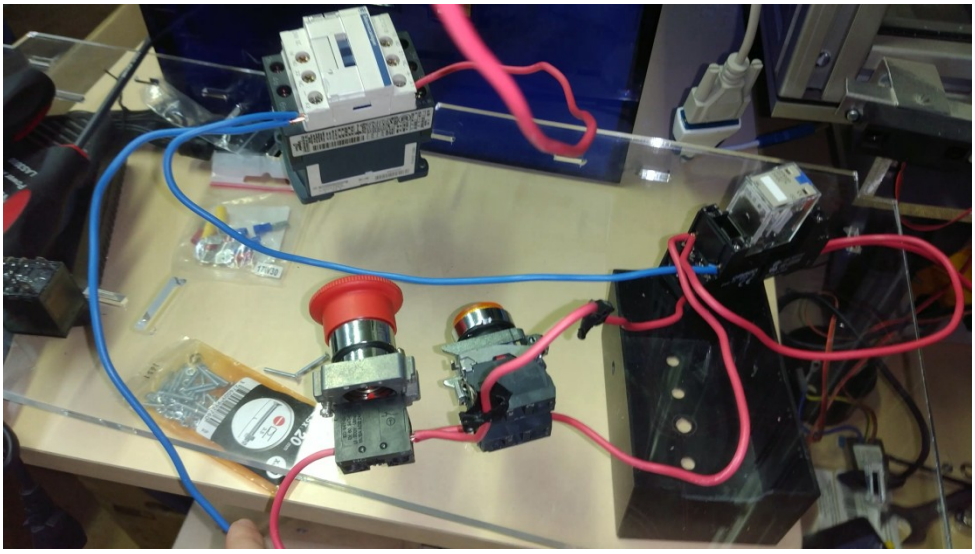
A l'issu de cette phase de tests, nous avons pu constater que le Centaure obéissait correctement à la consigne donnée par l'utilisateur à savoir avancer, reculer, tourner à droite ou à gauche, et ce, à l'image de la force exercée sur le joystick. Le test est donc concluant, et, puisque la partie commande du robot est terminée, nous pouvons passer à la sécurité du système.

VII. Sécurisation du robot

Une fois les tests concluant, nous nous chargeons de sécuriser le système en rendant opérationnel le bouton d'arrêt d'urgence et en obligeant l'utilisateur à initialiser le robot en appuyant sur un bouton avant de le faire fonctionner.

Nous branchons donc, en sortie du coupe-batterie, le bouton d'arrêt d'urgence qui est normalement fermé. Derrière ce bouton d'arrêt d'urgence se trouvent un bouton normalement ouvert et un des communs d'un relai 24 V. La bobine de ce relai est alimenté par la sortie du bouton, et un retour du contact normalement ouvert du relai qui permet de maintenir le relai alimenté : on

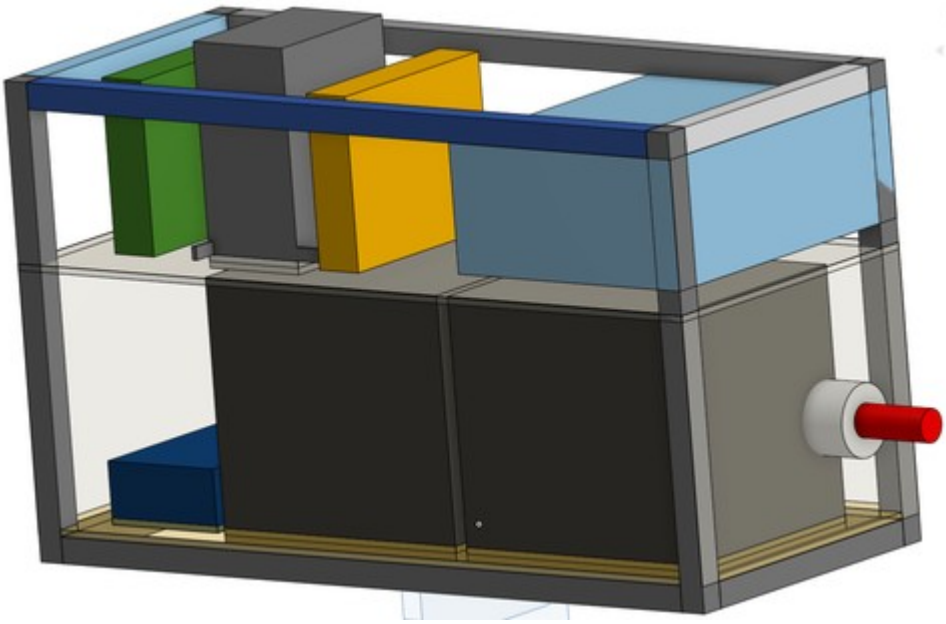
parle alors d'auto-maintien. En sortie du contact normalement ouvert du relai se trouve aussi la bobine du contacteur pilotant le circuit de puissance lorsqu'elle est a une tension de 24 V à ses bornes.



Avec ce mécanisme, si le bouton d'arrêt d'urgence est en position haute, il est nécessaire d'appuyer sur le bouton de départ de cycle pour initialiser le robot : la bobine du relai est alors alimentée et alimente à son tour la bobine du contacteur qui va permettre aux batteries d'appliquer une tension de 24 V sur les variateurs de vitesse. L'auto maintien bloquant le système en état de marche, il est alors possible d'arrêter le robot en appuyant sur le bouton d'arrêt d'urgence : il sera alors nécessaire d'amorcer à nouveau le système en appuyant sur le bouton de départ de cycle.

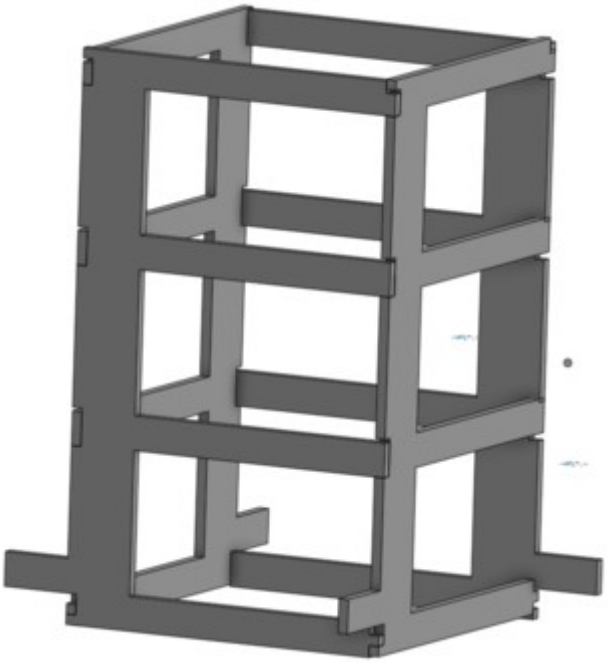
VIII. Conception des étages de puissance et de commande

Durant ce projet, nous devions aussi nous pencher sur la partie conception des étages de puissance et de commande et gérer l'intégration des différents modules. Afin de concevoir ces deux étages et d'organiser au mieux les différents éléments à l'intérieur du robot, nous avons réalisé un modèle 3D simplifié de l'arrangement prévu dont voici une capture :



Modèle 3D des deux étages du Centaure

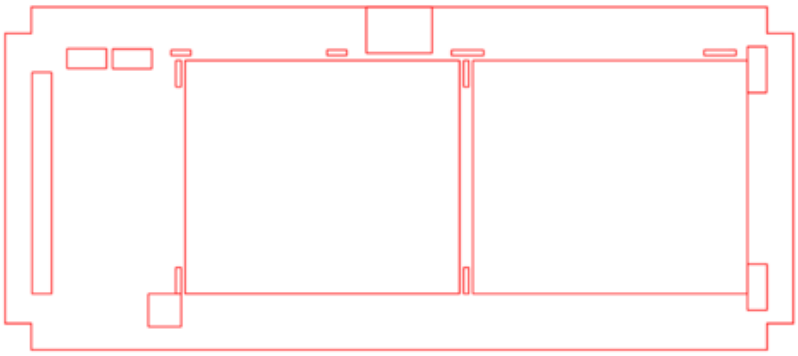
Nous pouvons voir, sur l'étage du bas, les deux batteries représentées par deux gros blocs gris. Cette hauteur des blocs comprenant la batterie elles-mêmes et leurs cosses. A gauche, en bleu, se trouve le convertisseur 24VDC -> 12VDC, posé sur la plaque permettant de surélever l'étage de puissance de 2cm, hauteur des profilés, afin de faire passer les différents câbles en dessous. Dans l'étage de commande, au dessus, nous retrouvons les deux variateurs de vitesses placés en position verticale, correspondant aux blocs vert et jaune. En gris, entre les deux, se trouve un bloc permettant d'accueillir les différentes cartes électroniques, à savoir l'Arduino MEGA, les quatre relais, ainsi que la carte électronique conçue pour ce projet. Nous retrouvons un visuel de ce bloc ci-dessous :



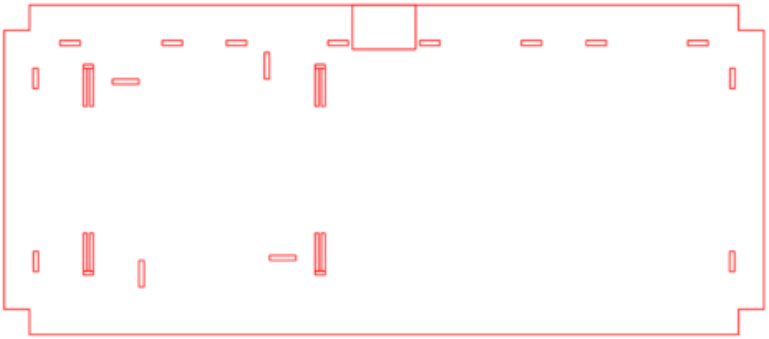
Modèle 3D du bloc hébergeant les différentes cartes électroniques

A droite, nous retrouvons en bleu l'emplacement de la carte mère de l'ordinateur. Le boîtier étant trop large pour le robot, il sera nécessaire à l'avenir d'ôter la carte mère et de lui confectionner un nouveau boitier plus adapté.

Nous obtenons donc les deux plans suivants qui nous serviront à créer les plaques en PMMA pour les deux étages grâce à la découpeuse laser du Fabricarium :



Plan de découpe de l'étage de puissance

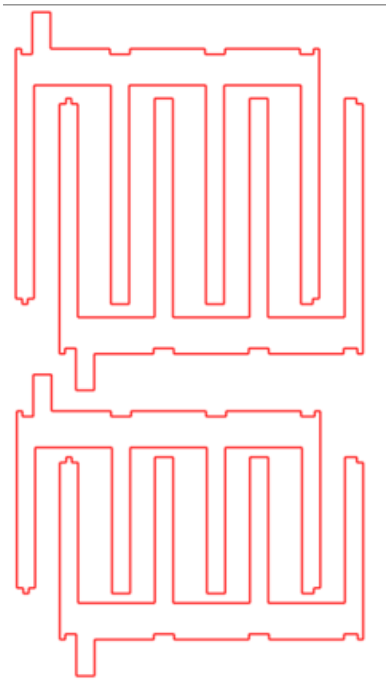


Plan de découpe de l'étage de commande

Lors de l'installation de ces plaques pour la confection des étages, nous avons pu nous apercevoir que des minces épaisseurs telles que la barre entre les deux trous accueillant les batteries ou encore la partie à droite reposant sur le profilé, pour l'étage de puissance, étaient des endroits très fragiles qui risquent fortement de casser lors de la mise en place du matériel.

Concernant l'étage de commande, la partie basse devrait être supprimée puisque, ne reposant pas sur un profilé, celle-ci empêche la fermeture de la porte.

Pour confectionner le bloc où seront placées les cartes électroniques, nous utilisons les deux schémas suivants :



Plan de découpe des différentes pièces servant de façades

Plan de découpe des 3 étages de la tour

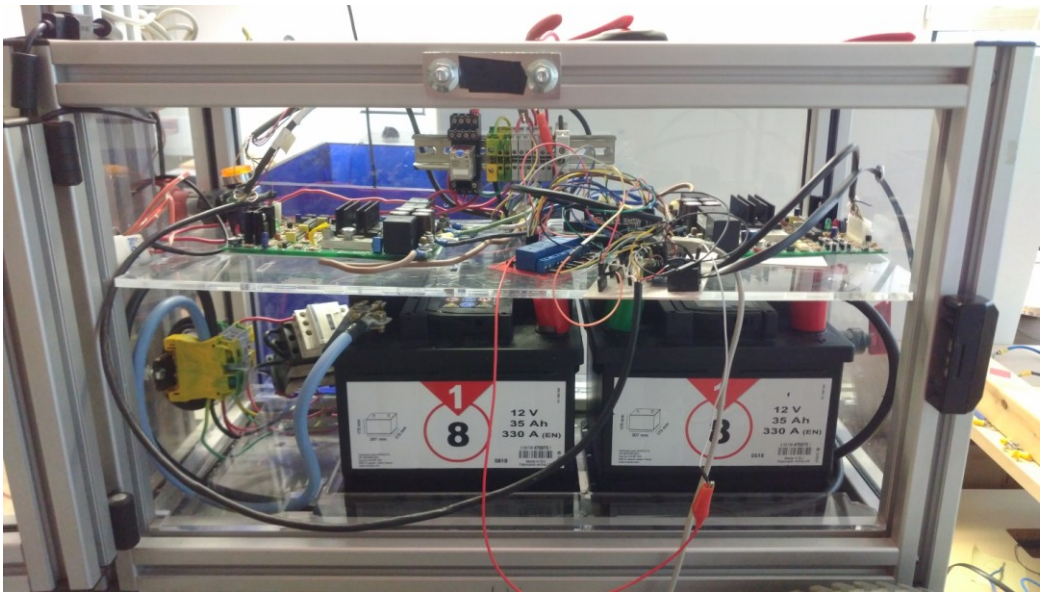
Après découpe, nous montons donc cette tour, fixée ici à l'aide de quelques bouts de chatterton en attendant la colle pour le modèle définitif :

Photo du prototype de la tour, fixée avec du chatterton

Bien qu'adaptée aux dimensions initialement prévues de l'étage de commande, la taille imposante des batteries récupérées en cours de projet ne permet plus d'utiliser un modèle de tour verticale comme celui-ci : il suffira d'adapter cette tour pour la faire pivoter de manière à ce que la longueur de la tour se trouve être contenue par la largeur de l'étage de commande.

Enfin, il sera nécessaire de percer le PMMA afin de fixer les différentes cartes électroniques et de faire passer des fils entre les différents étages.

Même si l'étage de commande n'était donc pas au point, nous avons tout de même tenté de mettre en place le système tel qu'il aurait été prévu, et avons donc câblé tout ce qui pouvait l'être. Le rendu final de ce projet est donc visible sur la photo suivante :



CONCLUSION

Ce projet m'a permis d'apprendre beaucoup de choses sur la robotique, la motorisation et la gestion d'énergie dans les systèmes embarqués. En effet, ce projet fut ma première expérience en ce qui concerne les systèmes embarqués à forte puissance, et mener ce projet par moi-même fut l'occasion de me retrouver face à des difficultés que je n'avais jamais rencontrées auparavant.

En effet, travailler sur ce genre de projet demande une attention toute particulière sur la sécurité, et donc une certaine maitrise des différents éléments constituants le robot. De plus, ce projet m'a demandé une certaine pluridisciplinarité propre au département IMA étant donné qu'il m'a fallu montrer des compétences en électrotechnique, en microélectronique et en informatique.

Durant ces mois passés sur le Centaure, j'ai pu apporter pour les prochaines équipes travaillent sur ce projet une nouvelle approche de contrôle du robot étant donné que j'ai apporté l'idée de contrôler ce robot via une commande analogique grâce à un convertisseur numérique vers analogique. Bien que l'état du robot ne permette, finalement, de tester le programme. Ce dernier a pu être testé auparavant et fonctionne correctement. Il conviendra donc de remettre en état le robot avant de continuer à développer les prochains projets.

REFERENCES

Sites web

- Documentation pour plans électriques :
  - Exemple de schémas et liste des repères :
    http://techelec.e-monsite.com/medias/files/memo-schema-electrotechnique.pdf
  - Schématisation de certains éléments
    https://slideplayer.fr/slide/10493093/

- Logiciel pour plan électriques :
  - Site du logiciel libre Qelectrotech
    https://qelectrotech.org/

- Logiciel pour conception de cartes électroniques :
  - Eagle
    https://www.autodesk.com/products/eagle/overview

- Ordinateur :
  - Consommation du processeur
    https://www.59hardware.net/articles/processeurs/intel-pentium-g2100t,-i3-3220t-et-i3-3240t:-ivy-bridge-%C3%A0-petits-prix-2012102813135.html?iframe=true&width=90%25&height=90%25all%2Fall%2Fall%2Fall%2Fall%2F&limitstart=5&fbclid=IwAR0BUp6l3hkl8rtiptlvkQf2ePrvchrK2J5k8huY6fdQtsT-DsYMPW133Fs

- Convertisseur 24 VDC vers 12 VDC :
  - Documentation constructeur
    http://www.farnell.com/datasheets/1772927.pdf?_ga=2.260083470.2081688051.1538052439-927647027.1537431483

- Convertisseur Numérique-analogique (DAC) MCP4725 :
  - Tutoriel Adafruit
    https://learn.adafruit.com/mcp4725-12-bit-dac-tutorial?view=all
  - Bibliothèque
    https://github.com/adafruit/Adafruit_MCP4725
  - Informations sur le bus I²C
    https://f-leb.developpez.com/tutoriels/arduino/bus-i2c/

- Manette :
  - Exemple de modèle 3D
    https://cults3d.com/fr/mod%C3%A8le-3d/gadget/diy-controller
  - Joystick
    https://itechnofrance.wordpress.com/2013/04/08/utilisation-dun-joystick-partir-de-larduino/
  - Joystick
    https://www.brainy-bits.com/arduino-joystick-tutorial/

- Codeurs :
  -

- Interruptions Arduino :
  - Tutoriel sur les interruptions des cartes Arduino
    http://www.locoduino.org/spip.php?article64

- Capteurs Sharp :
  - Documentation du modèle GP2Y0A02YK
    https://www.erasme.org/IMG/gp2y0a02_e.pdf
  - Documentation du modèle GP2Y3A003K0F
    https://media.digikey.com/pdf/Data%20Sheets/Sharp%20PDFs/GP2Y3A003K0F.pdf
  - Principe de fonctionnement
    http://www.societyofrobots.com/sensors_sharpirrange.shtml
  - Librairie pour certains capteurs Sharp
    https://github.com/guillaume-rico/SharpIR

- Joystick :
  - Github vers design d'une carte avec joystick
    https://github.com/adafruit/2-axis-joystick-breakout-board-with-mounting-holes/blob/master/joystick.brd

- LED RGB :
  - Site fournisseur
    https://www.gotronic.fr/art-led-rgb-5-mm-led5rvb-2103.htm
  - Documentation sur l'alimentation de LEDs
    https://www.astuces-pratiques.fr/electronique/led-et-calcul-de-la-resistance-serie

- Capteur de courant Allegro™ ACS712 :
  - Documentation constructeur
    https://docs-emea.rs-online.com/webdocs/0d88/0900766b80d885f5.pdf

- Capteur d'état de la batterie Analog Devices LTC2944 :
  - Documentation constructeur
    https://www.mouser.fr/datasheet/2/609/2944fa-1279557.pdf