Construction d’un robot 4x4

Pierre Bomel, Université de Bretagne Sud, Lab-STICC, Lorient, France

V2 par Ludovic Chavalarias, Université de Bretagne Sud, Lab-STICC, Lorient, France

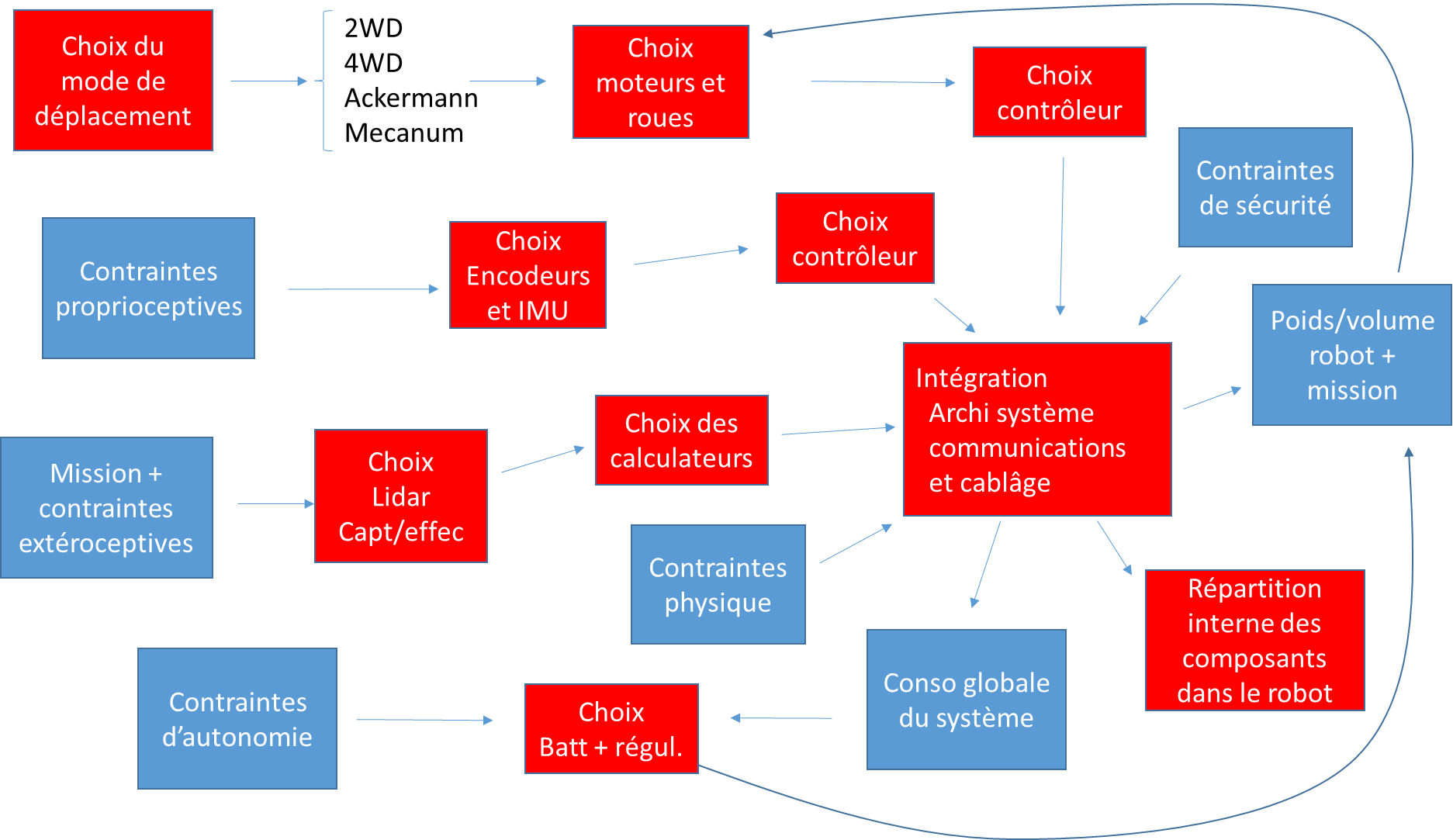
# Introduction

Ce document a pour but de préciser, par écrit, un processus de conception, de fabrication et de mise en œuvre d’un robot 4x4 autonome, piloté avec ROS. Un robot mobile étant une machine mécanique, électrique, électromécanique, électronique et informatique qui communique et interagit avec des êtres humains, toute décision (cahier des charges) concernant les fonctions à réaliser et les contraintes à respecter peut impacter tous ces domaines en même temps. De plus, l’interdépendance entre ces domaines complexifie encore la tâche. Le processus ici décrit tente d’être aussi simple et générique que possible de façon à permettre à des étudiants et des ingénieurs débutants de rapidement créer des robots dérivés pour divers cahiers des charges sans trop de difficultés. Nous espérons que ce sera une bonne base pour concevoir par la suite des robots plus complexes.

La mission du robot consiste à être une plateforme d’essais pour divers algorithmes mêlant perception, navigation et prise de décisions de façon autonome ou assistée par un être humain. Il s’agit d’un robot mobile, doté de batteries électriques, de roues motrices, de capteurs et de calculateurs qui communiquent potentiellement avec un ordinateur de contrôle distant. La puissance des calculateurs embarqués est telle qu’ils permettent aussi d’exécuter des algorithmes d’IA (Deep-Learning en particulier au moment de la rédaction) afin de traiter des images et/ou d’observer et d’apprendre sur le terrain. Le robot est donc à la fois un base robotique mobile, un objet communiquant (un objet connecté) et un outil intégrant de la technologie IA dynamiquement déployable sur le terrain.

# Principes de conception, généralités

La conception d’un robot est un sujet multidimensionnel, dont le schéma suivant offre une vue d’ensemble simplifiée. Dans ce chapitre nous n’allons pas concevoir un robot particulier mais plutôt mettre l’accent sur les généralités et les variations autour de ce thème.



Sont coloriées en rouges les activités nécessitant des choix et donc des arbitrages, sont coloriées en bleu les contraintes et les fonctions que le robot doit satisfaire sous peine de ne pas respecter le cahier des charges. Il faut aussi remarquer que **ce processus de conception n’est pas linéaire**. En effet, il est potentiellement rebouclé sur lui-même car, à la suite d’un nombre de choix divers (moteurs, roues, capteurs proprioceptifs, capteurs extéroceptifs, effecteurs, calculateurs, contrôleurs), le bilan énergétique conditionne le choix des batteries (hélas un des éléments les plus lourds et volumineux du robot). Ces dernières vont à leur tour vont conditionner le poids et le volume du robot, donc éventuellement remettre en cause le choix initial des moteurs. Le processus de conception d’un tel robot est donc nécessairement itératif, les experts ayant besoin de moins d’itérations que les débutants … L’essentiel étant surtout de ne rien oublier.

## Choix du mode de déplacement

De très nombreux modèles de robot mobiles au sol existent, ils peuvent être classés en quatre principales catégories qui se différencient par le mode de transmission du couple des moteurs et donc les mouvements possibles. Tous ne sont pas autonomes (ils sont alors télécommandés par un opérateur distant) ce sont, succinctement :

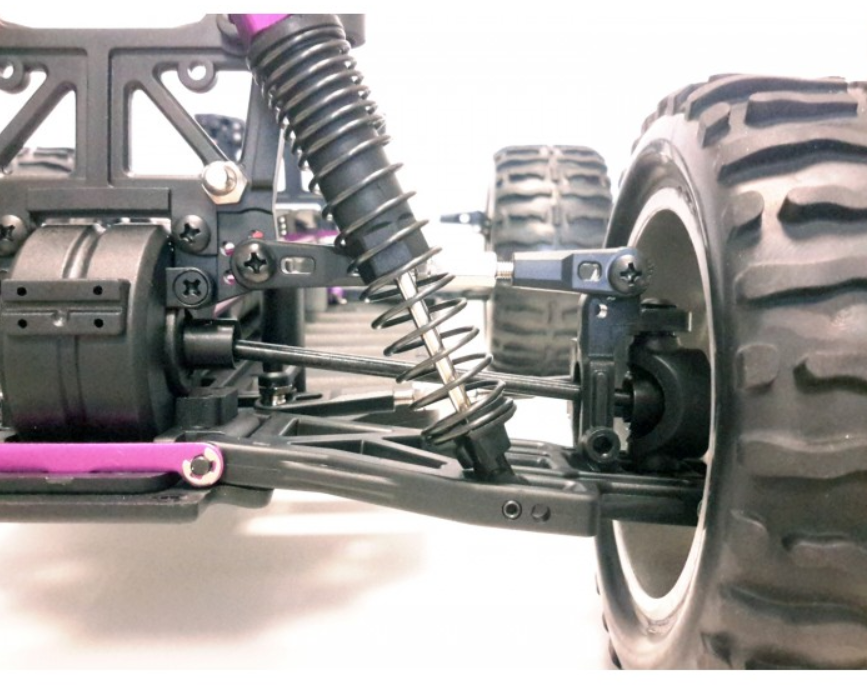
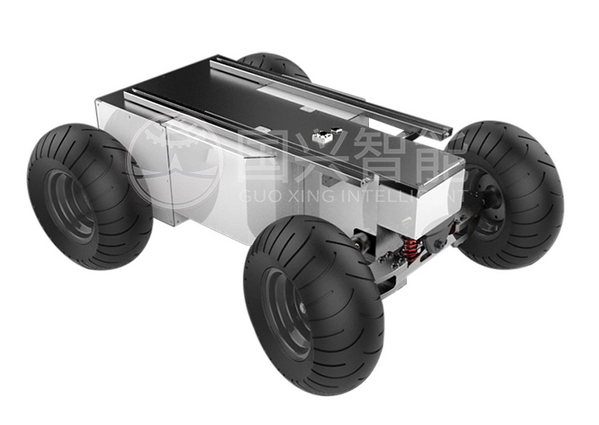
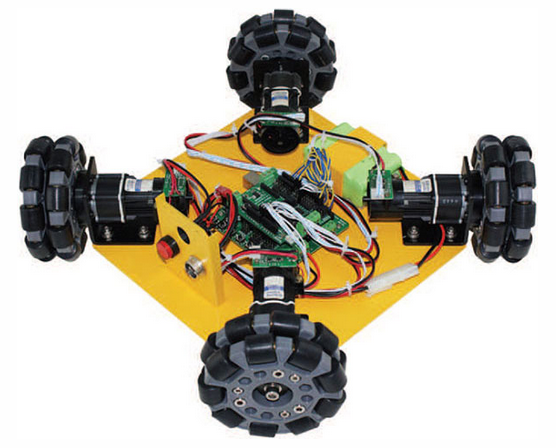
* Les robots à roues (ex, le robot livreur d’Amazon)
* Les robots à chenilles (ex, le Camper Trolley de la société Kronings)
* Les robots à pat
* tes (ex, le Wildcat de la société Boston Dynamics)
* Les robots sphériques (ex, le Guardbot de la société Guardbot)
* Il y en a même qui rampent (ex, le robot serpent de Carnegie Melon)

Les plus simples sont les robots à roues, ce que nous retenons. Les robots à roues, en plus de leur simplicité, ont aussi l’avantage par rapport aux robots à chenilles d’avoir un meilleur rendement en terme de transmission de la puissance moteur aux points de contact des roues avec le sol. En effet, les chenilles sont elles-mêmes des consommatrices d’énergie importantes : à cause de leur mode de fonctionnement intrinsèque, mais aussi lors des rotations à cause de leur grande surface de contact (et donc de résistance à la rotation) avec le sol. Parmi les robots à roues, différents types de robots existent à nouveau, selon le nombre, le type, le mode de suspension et la position des roues ainsi que la méthode de giration.

Quelle que soit la combinaison de tous ces paramètres, on peut les résumer à quatre qui sont :

* Le type de mouvement possible : ce sont soit des robots holonomes soit non-holonomes. Holonome signifient qu’ils peuvent se déplacer dans le plan horizontal sans contrainte, c’est-à-dire qu’ils peuvent avancer, reculer et tourner sur eux-mêmes. Non holonome signifient qu’ils ont un mécanisme de rotation qui ne leur permet pas de tourner sur eux-mêmes. Citons quelques exemples de directions holonomes et non-holonomes :
  + Holonome : La rotation sur place est toujours possible avec un robot à roues mecanum ou avec 4 roues motrices
  + Non-holonome : type « chariot tracté » avec une seule roue directrice ou direction de voiture Ackermann (c’est-à-dire propulsion et direction séparées)
* Le nombre de roues motrices : 2 (2WD + roues folles) ou 4 (4WD), voire plus, mais cela nous semble un peu superflu dans un premier temps.
* Les roues standards et les roues mecanum
* Le type des suspensions



Holonome, 4WD, mecanum ackermann avec roues standard ex de suspensions

Les robots que nous proposons seront donc capables 1) d’utiliser des roues standards ou bien des roues « mecanum », 2) d’avoir de 2 à 4 roues motrices et enfin 3) d’être holonomes ou de type Ackermann. Ces trois paramètres sont des éléments très importants car ils **doivent être supportés par toute la chaîne logicielle** de contrôle/commande du robot (bas niveau) et de navigation/pilotage (haut niveau).

Par contre, il nous semble que le type de suspension n’a aucun impact autre que mécanique. On devrait pouvoir choisir librement tout type de suspension, voire aucune si l’on dispose de roues suffisamment souples qui absorbent les chocs dus aux irrégularités du sol et les vibrations dues aux moteurs. Mais cela dépend du contexte d’utilisation : certains terrains nécessiteront des suspensions ou bien des moteurs avec des axes très résistants. Cela existe, ce sont les moteurs et les châssis destinés aux robots nommés « robots de combats ». Mais, dans ce document, nous n’en sommes pas là.

## Forme du robot

A partir du moment où le choix des roues et le type de giration sont choisis, la forme du robot importe relativement peu, à moins qu’il soit imposé par le cahier des charges. Il suffit de disposer d’un châssis sur lequel on pourra fixer les moteurs et les roues (plus tout ce qui sera nécessaire à terme, batteries, calculateur, capteurs, effecteurs, …). Néanmoins, il faut considérer qu’il existe une contrainte forte issue des logiciels de planification qui considèrent bien souvent le « pire cas » en terme de dimensions extérieures du robot. Pour simplifier, le logiciel assimile le robot à une forme symétrique dont le diamètre correspond à la plus grande longueur du robot. Il est donc pertinent de concevoir le robot le plus compact possible, pour avoir un diamètre minimal. L’idéal en 3D est une sphère, et un cercle en 2D. Pour des raisons de simplification de la fabrication, nous suggérons des formes simples de type cube ou pavé. Mais nous laissons toute liberté au lecteur.

# Conception électromécanique d’un exemplaire du robot

## Choix du mode de déplacement

Prenons comme exemple le cas d’un robot holonome 4x4, ayant 4 roues motrices (4WD).



Cette photo est celle d’un prototype réalisé par l’auteur dans son garage avec du contre-plaqué.

## Forme et matière du châssis du robot

Le robot sera contenu dans une enceinte de forme « pavé » ou « cube » ayant une surface plane libre à son sommet pour pouvoir y installer divers équipements. Le châssis sera construit avec des plaques d’aluminium de 2mm d’épaisseur, boulonnées entre elles. Ce matériau permet en effet de construire des châssis de robots jusqu’à quelques dizaines de kg sans avoir besoin de faire une étude mécanique.

## Choix des moteurs et des roues

On ne peut choisir les moteurs que si l’on a déjà une idée du poids du robot, des dimensions des roues et des performances attendues et terme de vitesse maximale, d’accélération et de pente qu’il pourra gravir. Pour cela, il existe de nombreux tutoriaux, vidéos et d’excellents cours sur Internet sur ce sujet. Nous avons néanmoins rédigé un document séparé[[1]](#footnote-1) qui explique notre méthode de calcul ainsi que les fondements physiques et mathématiques sur lesquels elle repose. Notre méthode diffère un peu des cours standards dans la mesure où l’on part de l’accélération requise dans le pire cas plutôt que du théorème de la variation de l’énergie cinétique. Bien sûr, les résultats sont les mêmes, nous avons vérifié. Dans ce document nous ne ferons qu’en exploiter les résultats.

Notre cahier des charges est donc :

* La vitesse maximale = Vmax = 5 km/h = 1,39 m/s
* L’accélération maximale, ou TVmax = 2 secondes pour atteindre la vitesse maximale
* La pente maximale = ThetaMax = 5° = 0,087 rad
* Le poids du robot = 13 kg (détail suit), mr = 13 kg, + 7 kg de charge utile, mc = 7 kg
  + 7,5 kg de batteries au plomb (1 x 24 Ah ou 2 x 12 Ah, ou 4 x 6 Ah, 12V)
  + 1,5 kg de moteurs
  + 2 kg de châssis
  + 2 kg d’électronique
* Dimension des roues[[2]](#footnote-2) (ces roues sont celles qui sont utilisées par plusieurs robots tous-terrains[[3]](#footnote-3) du commerce) diamètre = 120 mm, donc rayon = r = 0,06 m.
* 

Le résultat du calcul nous donne les valeurs suivantes :

* Roues motrices *N = 4*
* Accélération
* Couple total = 0,834 + 1,02 N.M = 1,86 N.m
* Couple par moteur *Cm = C / N = 1,86 / 4 = 0,465 N.m = 4740 g.cm = 65,9 oz.in*
* Vitesse angulaire max RPM =
* Puissance
* Voltage nous choisissons des batteries standard au plomb de 12 V, 12 Ah

Il est donc nécessaire de trouver un moteur électrique ayant ces caractéristiques. Nous avons rédigé un document séparé[[4]](#footnote-4) qui explique comment choisir les moteurs. Nous retiendrons que :

* dans le pire cas, lors de l’accélération sur un plan incliné, le couple ne doit pas dépasser 50% du couple bloqué. Il faut donc un moteur avec un couple de décrochage supérieur ou égal à 9480 g.cm (0,93 N.m ou 131 oz.in).
* en mode continu, bien que l’accélération et la pente soient nulles, il faut encore vaincre les forces de frottement ainsi que les irrégularités du sol. Bien évidemment, le robot fonctionnera toujours en deça des contraintes du pire cas. La mesure de la consommation électrique permettra de vérifier à quelle distance du point de meilleur rendement on se trouve. Ce mode de fonctionnement devrait se trouver de préférence dans le mode « nominal ». Il faut alors trouver un moteur dont la vitesse nominale est au moins de 221 RPM.

Les quelques moteurs électriques que l’on peut trouver chez Robotshop[[5]](#footnote-5) et qui se rapprochent des spécifications[[6]](#footnote-6), sont listés dans le tableau qui suit. Tous sont prévus pour fonctionner sous 12 V.

Robotzone #638280 RB-Sct-948

* A vide courant 0,52 A, vitesse 313 RPM +- 31
* Nominal courant 2A, couple 4,5 kgf.cm (62 oz.in), vitesse 276 RPM +- 28, puissance électrique consommée 24W, mécanique 13 W
* Bloqué courant 20A, couple 30 Kgf.cm (416 oz.in)
* Charge radiale 3,5 kg max (4 axes => 14 kg sans engrenages supplémentaires)
* Charge axiale 2,5 kg max
* Poids 300g
* Arbre 6 mm, forme en D

Robotzone #638282 RB-Sct-1013 (même moteur, engrenages différents, plus rapide en RPM, mais couple plus faible)

* A vide courant 0,52 A, vitesse 437 RPM +- 44
* Nominal courant 2A, couple 3,2 kgf.cm (44 oz.in), vitesse 389 RPM +- 39, puissance électrique consommée 24W, mécanique 13W
* Bloqué courant 20A, couple 20 Kgf.cm (306 oz.in)
* Charge radiale 3,5 kg max (4 axes => 14 kg sans engrenages supplémentaires)
* Charge axiale 2,5 kg max
* Poids 300g
* Arbre 6 mm, forme en D

FIT0493

* A vide courant 0,02 A, vitesse 350 RPM
* Efficacité max 0,65 A, 2 kg.cm (27,7 oz.in), 285 RPM, 5 W (ou nominal)
* Puiss max 1,65 A, 5,8 kg.cm (80,5 oz.in), 180 RPM, 9 W méca
* Bloqué courant ?, couple 12 Kgf.cm (306 oz.in)
* Poids 100 g
* Arbre 4 mm, forme en D

SPG30HP-30K

* A vide courant 0,03 A, vitesse 380 RPM+-10%
* Efficacité max 1 A, 1,4 kg.cm (19,4 oz.in), 300 RPM+-10% (ou nominal)
* Bloqué 3,7 A, 6,5 kg.cm (90,3 oz.in)
* Poids 160 g
* Arbre 6 mm, forme en D
* Courant dém. 3,7 A

RB-Cyt-90, IG32E-14K[[7]](#footnote-7)

* A vide courant 0,15 A, vitesse 520 RPM
* Efficacité max 1 A, 1,1 kg.cm (15,3 oz.in), 430 RPM (ou nominal), 7W méca
* Bloqué 5 A, 8,6 kg.cm (119 oz.in)
* Poids 240 g
* Arbre 6 mm, forme en D

Le moteur qui semble le plus approprié est le RB-Cyt-90 car :

* Sa zone de fonctionnement nominal se situe à environ un ¼ du couple calculé « pire cas »  
  15,3 oz.in ~= ¼ de 66 oz.in
* Sa zone de fonctionnement « pire cas » se situe à environ ½ du couple bloqué. On ne devrait donc pas avoir de fonctionnement en zone dangereuse pour le moteur.  
  66 oz.in ~= ½ 119 oz.in
* Le diamètre de l’arbre est de 6mm (4mm nous semble en effet un peu faible). Dans ces conditions chaque moteur devrait pouvoir porter jusqu’à 3,5 kg (soit 14 kg pour le robot). C’est un peu moins que les 20 kg prévus et cela correspond au robot sans charge utile. On prend le risque de partir avec l’accouplement direct des roues avec l’arbre des moteurs sans engrenage intermédiaire dans un premier temps.
* La consommation électrique de chaque moteur est de l’ordre d’un ampère en nominal, et sera probablement le double en « pire cas ».
* En terme de puissance mécanique fournie, elle est un peu inférieure (7W) à la spec (11W), mais il est préférable d’avoir des moteurs qui consomment moins (enfin, c’est le choix fait aujourd’hui) et pèsent moins, quitte à gravir un peu moins vite que prévu les plans inclinés.

Les calculs concernent le « pire cas » d’une accélération sur un sol en pente que le robot doit gravir. Mais, ce cas de figure est loin de représenter la totalité des situations que le robot va rencontrer lors de ses déplacements. Pour estimer l’autonomie du robot, il faut avoir une idée du comportement des moteurs dans le temps. C’est-à-dire qu’il faut pouvoir combiner la consommation des phases d’accélération et du régime de croisière : donc disposer d’un profil de comportement qui indique pendant quel pourcentage du temps de la mission le robot fonctionne en mode « accélération ».

Soient

* Pa la puissance dissipée en phase d’accélération = 2 A x 12 V = 24 W
* Pc la puissance dissipée en régime de croisière = 1 A x 12 V = 12 W
* α le pourcentage du temps pendant lequel le robot fonctionne en mode « accélération » = 5%

La puissance moyenne[[8]](#footnote-8) électrique consommée par chaque moteur est :

Soit Pm x N = 50,4 W pour les quatre moteurs

Ce qui représente un courant moyen de 50,4 W / 12 V = 4,2 A

Soit une autonomie d’environ 5,72 heures avec une batterie de 24 Ah.

Retenons une autonomie de 5 à 6 heures. Les données techniques d’une batterie au plomb[[9]](#footnote-9) pour usages cycliques confirment que :

* Pour un poids de 7,2 kg et une capacité de 24 Ah
* On a une autonomie de l’ordre de 5 h avec un courant de décharge de l’ordre de 4 A.

## Choix des contrôleurs des moteurs

La spécification est de piloter, via des signaux PWM, des moteurs 12V avec un courant moyen de 1 A (régime de croisière) et maximal de 2 A (pire cas). Il faut aussi remarquer que les courants de démarrage sont parfois importants mais de courte durée. Ils sont de 5 A pour le moteur choisi.

Le contrôleur de moteurs « Contrôleur de deux moteurs DC 10A 5-30V Cytron » de Robotshop tient toutes ces contraintes. Il peut :

* piloter deux moteurs balais séparément avec des PWM
* pour un total de 10A en continu et un total de 30 A en crête
* à partir d’une source de tension de 5 à 30 V
* dimensions 84,5 x 62 mm
* il permettra de piloter des moteurs plus puissants par la suite
* il est le mieux placé en prix à la date de la rédaction

Il en faut deux pour piloter les quatre moteurs du robot. Ce contrôleur remplacera sans grandes difficultés le L298 que nous utilisions jusqu’à présent, mais qui ne peut pas supporter plus de 2A en continu et qui risque donc d’être endommagé lors des démarrages (5A).

## Choix du calculateur de contrôle du robot

Ce sera une carte Teensy[[10]](#footnote-10), que nous connaissons très bien, et qui est très peu onéreuse. Il y a plusieurs modèles, nous avons expérimenté la Teensy 3.1/3.2. Elle contient un ARM Cortex-M4 cadencé à 72 MHz. Elle dispose de 256 Ko de flash et de 64 ko de RAM. Elle est supportée par l’IDE Arduino sous le nom Teensyduino. Ses 34 entrées-sorties sont compatibles 5V/3.3V. Elle possède un régulateur 3.3 V pour alimenter la centrale inertielle et évite ainsi d’ajouter un régulateur de plus. Le site PRJC ne donne pas la consommation électrique de la carte mais on peut penser que c’est de l’ordre de 100 à 200 mA au grand maximum. Une consommation de 200 mA pendant 4 heures représente 0,8 Ah.

## Choix de la centrale inertielle

Ce sera une simple centrale inertielle MPU6050 à 6 axes (pas de magnétomètre) que nous connaissons bien. Elle consomme de l’ordre de quelques dizaines de mA. Elle communique avec la Teensy via un bus I2C (2 fils).

## Choix du calculateur ROS

Le robot doit impérativement embarquer une partie de ROS sur un calculateur apte à l’exécuter dans de bonnes conditions. Il s’agit d’un calculateur sur lequel tourne Linux et dans lequel on peut installer ROS (ce qui n’est pas évident du tout pour toutes les versions de Linux).

Nous avons testé avec succès une carte Jetson-nano (destinée à embarquer des fonctions « IA ») et une carte Raspberry PI 4. Ces cartes ont les consommations suivantes (pires cas) :

* Jetson-nano 5A, 5V, 25 W, soit l’équivalent de 2 moteurs de plus
* RaspBerry PI4, 3A, 5V, 15 W

La carte Jetson-nano nécessite manifestement d’ajouter des batteries pour que la durée de fonctionnement du robot ne descende pas en dessous de 4h. On peut considérer qu’alors, les fonctions « IA » feront partie de la charge utile et contiendront leur source d’énergie. Mais ce n’est pas un bon choix pour le pilotage du robot avec ROS. Nous retenons donc la PI4 comme calculateur ROS.

## Choix du Lidar

Ce n’est pas un véritable choix, car nous prenons ce que nous avons « sous la main ». Il s’agit du YDLIDAR X4[[11]](#footnote-11) qui sera relié par USB au calculateur ROS. C’est un lidar 360°, d’une portée de 12 cm à 11m, fréquence de rotation de 6 à 12 Hz, max 5000 points/s, précision de mesure de l’ordre de 1%, il fonctionne à 5V. Mais tout autre modèle, compatible ROS et ne consommant pas plus d’énergie pourra le remplacer si besoin.

## Bilan des consommateurs d’énergie sur une durée de 4h et pour une batterie de 24 Ah

4 x Moteur 4 x 1,05 A = 4,2 A 4,2 A x 4 h = 16,8 Ah 70%

Jetson-nano 5V, 5A 25 W, 50% des moteurs 8 Ah 35% (c’est trop)

Raspberry PI 4 5V, 3A 15W, 30% des moteurs, 5 Ah 21% (ça passe)

YDLIDAR 350 mA 350 mA x 4h = 1,4 Ah 5,8%

2 x drivers moteurs ? ? ?

1 x Teensy 3.1 200 mA 200 mA x 4 h = 0,8 Ah 3,3%

IMU 6050 20 mA 20 mA x 4 h = 0,08 Ah 0,33 %

Régulateurs ? ? ?

Total 100,43

Tous les éléments n’ont pas été listés, il manque en effet la consommation des drivers des moteurs et des régulateurs 5V et 12V. Néanmoins, cette sélection de composants tient à peu près la contrainte d’autonomie. Ces éléments favorisent un peu l’autonomie au détriment des performances « pire cas ». Enfin, on remarquera que la puissance est répartie en ¾ pour la motricité et ¼ pour le calcul et la communication.

## Choix des régulateurs/stabilisateurs de tensions et des batteries

Ce sont les derniers éléments à choisir car ils dépendent de tous les calculateurs, contrôleurs, drivers et moteurs qu’il faut alimenter. On a donc besoin :

* D’un régulateur 5V (PI4 + lidar + Teensy + IMU) = (3 + 0,350 + 0,200 + 0,02) = 3,57 A, disons 4 A sous 5V, donc 20W
* D’un régulateur de tension 12V, 5A en moyenne, 20A max, donc 240W

La tension 12V est exclusivement destinée à l’alimentation des moteurs. Ils sont plus tolérants aux variations de tension que les dispositifs numériques (tous alimentés sous 5V ou 3V). Il y a trois façons de produire cette tension :

* Directement à partir de batteries 12V, lorsqu’elles sont correctement chargées, c’est à dire entre 11,5 et 12,5 V.
* 12V : A partir d’une seule batterie 24Ah 12V munie d’un stabilisateur de tension[[12]](#footnote-12) 12V. La spec du stabilisateur prévoit que la tension délivrée soit de 12V +-10%, ce qui revient au même que de brancher les batteries en direct.
* 24V : A partir de deux batteries en série de 12Ah 12V en série (24V), avec le même stabilisateur de tension.

L’association de batteries en parallèle est à déconseiller car elle introduit des déséquilibres en courant lors des charges et des décharges et nuit au parc entier. Nous ferons des essais pour décider quelle est la meilleure option.

Les principaux inconvénients du stabilisateur de tension sont :

* son poids (520g)
* et son volume (157x90x47 mm).

En ce qui concerne le régulateur 5V (20W) un seul modèle correspond à ces contraintes chez RS, il s’agit du PSR54-9G[[13]](#footnote-13).

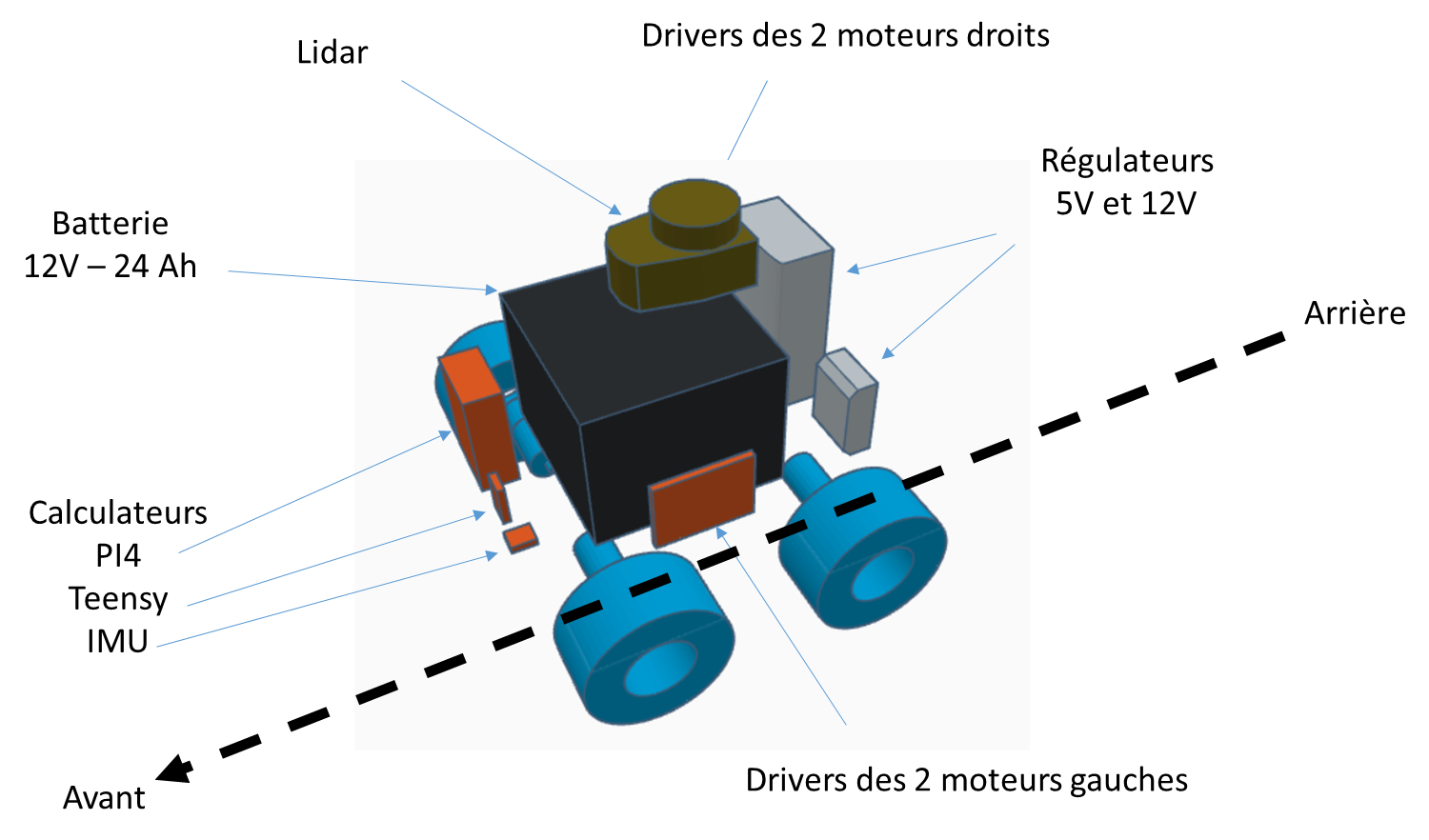
## Choix de la méthode de rechargement des batteries

Il sera nécessaire de recharger régulièrement les batteries. Pour cela le robot rejoindra automatiquement (ou manuellement) un emplacement conçu pour cette fonction et où se trouveront un chargeur de batteries ainsi les contacts électriques (des pinces ou des plots) auxquels le robot sera relié le temps de la charge. En ce qui concerne (par exemple) la batterie 12V 24Ah retenue, le temps de charge dépend bien sûr de sa décharge mais, dans le pire cas, 12 heures (c’est-à-dire une nuit) suffisent à les recharger complètement.

En terme de disponibilité on constate que, dans ces conditions, le robot ne fonctionne que 4h toutes les 12h. C’est-à-dire qu’il n’est opérationnel que ¼ du temps. Il est fort probable que pour un usage professionnel ce ne soit pas suffisant. Si c’est le cas, il faudra alors envisager un mécanisme de remplacement des batteries « vides » par des « pleines », ou plusieurs robots (4 en l’occurrence) dont on alterne la mise en route pour être capable d’assurer un service permanent. On sort du cadre de ce document qui ne concerne qu’un robot « académique », mais il nous semblait pertinent d’aborder aussi ce sujet quelques instants.

## Etude d’implantation

Ayant déterminé les différents éléments constitutifs du robot, nous connaissons leurs dimensions et nous pouvons maintenant créer un modèle 3D du robot. Ce modèle 3D n’a pas pour but de devenir le dossier de fabrication pour une sous-traitance mécanique, mais seulement une ébauche (au centimètre près, cela suffit) pour avoir une idée de son volume et de sa forme avant de prendre la décision de poursuivre dans cette voie.



Une rapide étude d’implantation[[14]](#footnote-14) (avec des boîtes dont les volumes représentent celui des différents éléments) nous donne un robot ayant les caractéristiques suivantes :

* La batterie est au centre car c’est l’objet le plus lourd, donc celui qui va influer le plus fortement sur la position du centre de gravité du robot
* La zone des calculateurs est à l’avant
* La zone des régulateurs est à l’arrière
* Les drivers des moteurs sont placés sur les côtés
* Le lidar est fixé à l’extérieur, au centre sur le « toit »

Les dimensions sont approximativement pour le corps du robot (une enceinte en tôle d’aluminium de 2 mm d’épaisseur)

* 32 cm de longueur
* 22 cm de largeur
* 15 cm de haut

Le lidar dépasse d’environ 6 cm en haut.

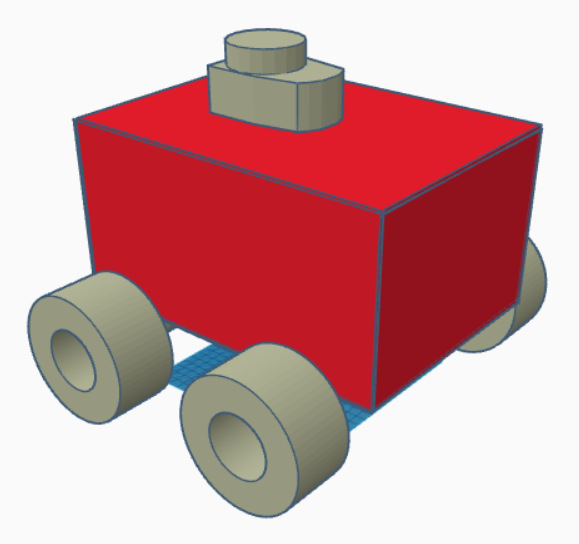
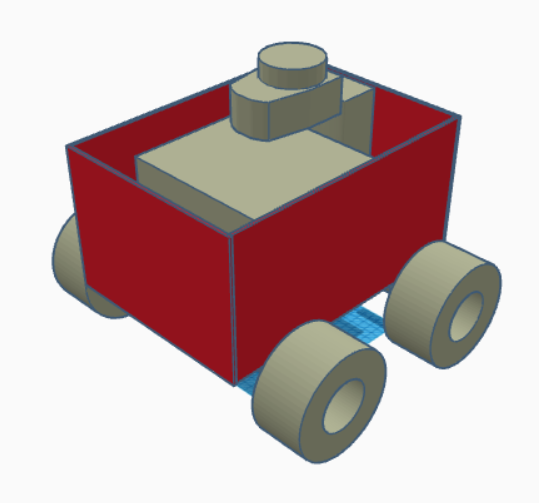
Les roues soulèvent le robot d’environ 9 cm du sol et dépassent sur les côtés d’environ 7 cm.

On a donc les dimensions « hors tout » suivantes :

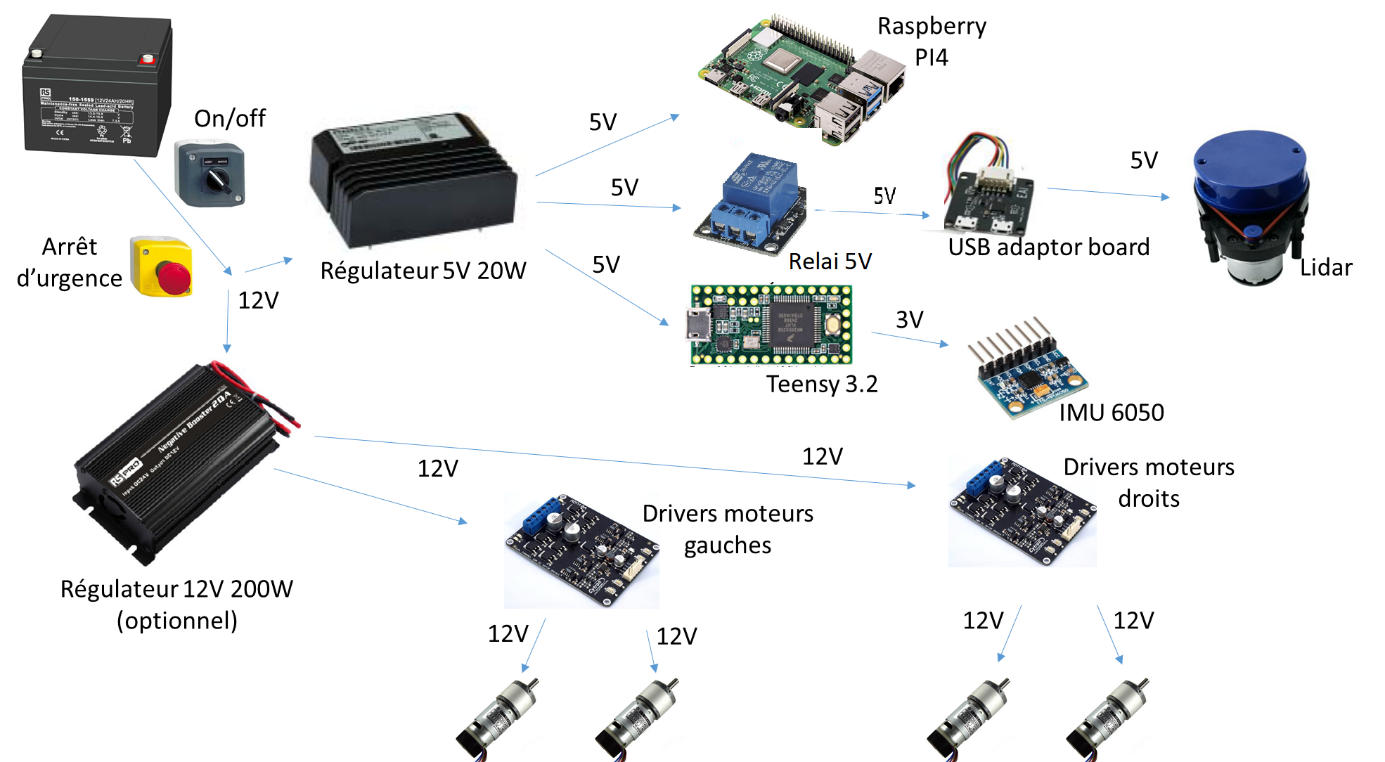
* Longueur 32 cm
* Largeur 22 + 2 x 7 = 36 cm
* Hauteur 9 + 15 + 6 = 30 cm

Soit 0,35 m3 dans un volume proche d’un cube, comme nous l’espérions en début de document.

L’aspect du robot caréné sera le suivant.



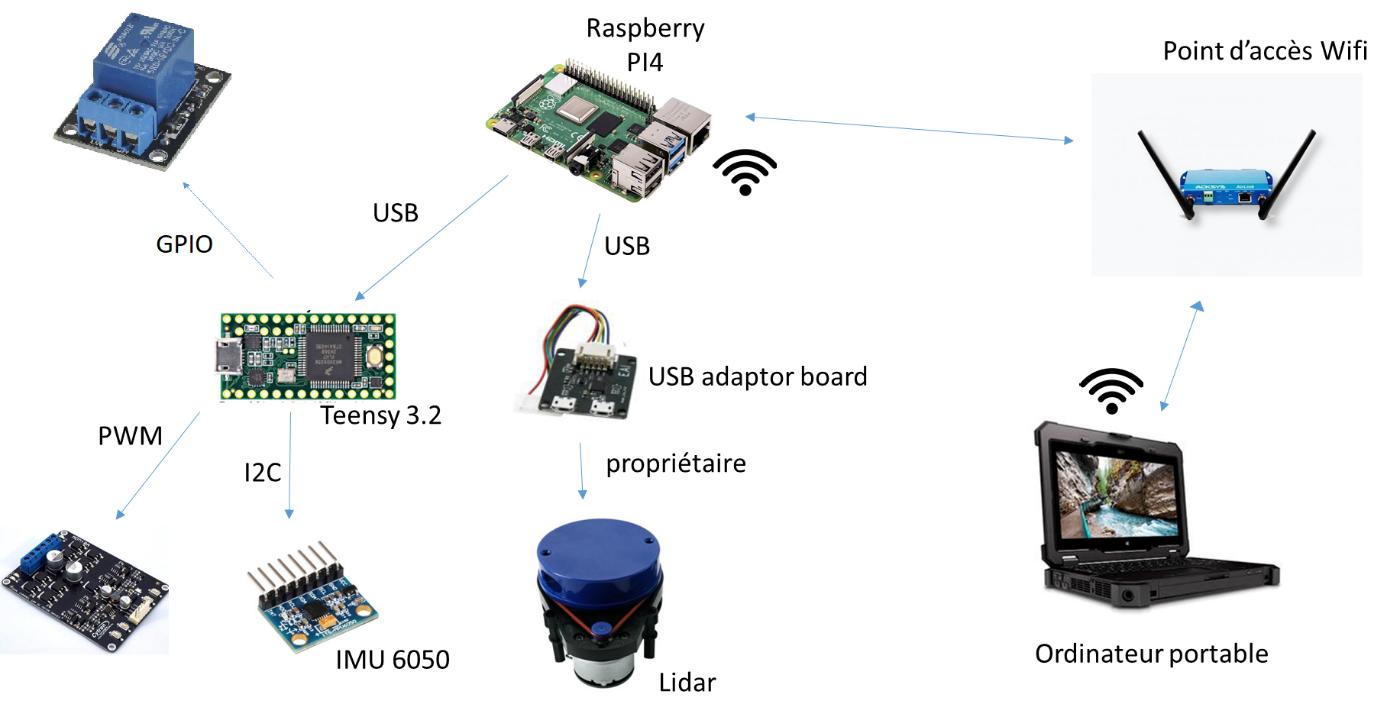
## Distribution électrique



V2 : Un relai est ajouté pour piloter l’alimentation de Lidar.

# Informatique

## Architecture des communications



V2 : Le relai est commandé par une simple GPIO qui passe à l’état haut lors de l’initialisation.

# Installation de ROS sur la Raspberry PI4[[15]](#footnote-15)

Il faut installer la version « melodic » car le Linux de la Raspeberry Pi est équivalent à un Linux Buster (version 10). Détail suit :

Mettre à jour les repositories pour télécharger les fichiers de ROS

% sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu $(lsb\_release -sc) main" > /etc/apt/sources.list.d/ros-latest.list'

Mettre à jour les clés pour avoir le droit de télécharger

% sudo apt-key adv --keyserver hkp://ha.pool.sks-keyservers.net:80 --recv-key C1CF6E31E6BADE8868B172B4F42ED6FBAB17C654

OU

% sudo apt install curl # if you haven't already installed curl

% curl -s https://raw.githubusercontent.com/ros/rosdistro/master/ros.asc | sudo apt-key add -

Mettre à jour le Linux si ce n’est pas déjà fait (normalement cela a été déjà fait lors de l’installation de Raspberry Pi OS).

% sudo apt-get update

% sudo apt-get upgrade

Télécharger les packages Linux qui sont nécessaires pour l’installation de ROS

% sudo apt install -y python-rosdep python-rosinstall-generator python-wstool python-rosinstall build-essential cmake

Télécharger les mêmes packages pour python3

% sudo apt install -y python3-rosdep python3-rosinstall-generator python3-wstool python3-rosinstall

Initialiser rosdep

% sudo rosdep init

% rosdep update

Créer le workspace catkin dans lequel ROS sera compile

% mkdir –p ~/ros\_catkin\_ws

% cd ~/ros\_catkin\_ws

* **Option 1** : Installer le ROS-variant « ROS-comm »

% rosinstall\_generator ros\_comm --rosdistro melodic --deps --wet-only --tar > melodic-ros\_comm-wet.rosinstall

% wstool init src melodic-ros\_comm-wet.rosinstall

Résoudre les dépendances, s’il y en a (C’est pour un Linux Buster !!!)

% rosdep install -y --from-paths src --ignore-src --rosdistro melodic -r --os=debian:buster

Compilation des sources

% sudo ./src/catkin/bin/catkin\_make\_isolated --install -DCMAKE\_BUILD\_TYPE=Release --install-space /opt/ros/melodic

* **Option 2** : Installer le ROS-variant « desktop »

% rosinstall\_generator desktop --rosdistro melodic --deps --wet-only --tar > melodic-desktop-wet.rosinstall

% wstool init src melodic-desktop-wet.rosinstall

% wstool merge -t src melodic-desktop-wet.rosinstall

% wstool update -t src

Résoudre les dépendances, s’il y en a (C’est pour un Linux Buster !!!)

% rosdep install -y --from-paths src --ignore-src --rosdistro melodic -r --os=debian:buster

Compilation des sources

% sudo ./src/catkin/bin/catkin\_make\_isolated --install -DCMAKE\_BUILD\_TYPE=Release --install-space /opt/ros/melodic

# Installation de Linorobot sur la Raspberry PI4

% source /opt/ros/melodic/setup.bash (Ajouter le source au fichier .bashrc)

Créer un workspace catkin

% mkdir –p ~/catkin\_ws/src

% cd ~/catkin\_ws

% catkin\_make

Récupérer le code des packages suivants (pour LINOROBOT sous ROS melodic) sous ros\_catkin\_ws/src (« -b melodic-level » indique la branche de développement)

% cd ~/ros\_catkin\_ws/src

% git clone https://github.com/ros/roslint.git

% git clone -b melodic-devel https://github.com/ros-drivers/rosserial.git

% git clone -b melodic https://github.com/ccny-ros-pkg/imu\_tools.git

% git clone -b melodic-devel https://github.com/ros-perception/slam\_gmapping.git

% git clone -b melodic-devel https://github.com/ros-planning/navigation.git

% git clone -b melodic-devel https://github.com/cra-ros-pkg/robot\_localization.git

% git clone -b melodic-devel https://github.com/ros-perception/openslam\_gmapping.git

% git clone https://github.com/ros-planning/navigation\_msgs.git (supprimer le dossier navigation\_msgs déjà présent)

% git clone https://github.com/ros-geographic-info/geographic\_info.git

% git clone https://github.com/ros-geographic-info/unique\_identifier.git

% git clone -b melodic-devel https://github.com/ros/geometry2.git (supprimer le dossier geometry2 déjà présent)

Source les nouveaux fichiers :

% source ./devel\_isolated/setup.bash

Résoudre les dépendances, s’il y en a (C’est pour un Linux Buster !!!)

% rosdep install -y --from-paths src --ignore-src --rosdistro melodic -r --os=debian:buster

Si la commande rosdep n’est pas reconnu :

% sudo apt-get install python-rosdep

Compilation des sources

% sudo ./src/catkin/bin/catkin\_make\_isolated --install -DCMAKE\_BUILD\_TYPE=Release --install-space /opt/ros/melodic

# Récupérer les sources de LINOROBOT

% cd

% git clone <https://github.com/linorobot/lino_install>

% cd lino\_install

(modifier .install)

Ligne 88 retirer l’installation du package python-gudev qui n’existe plus

Ligne 90 retirer l’installation de pip avec easy\_install, car pip est maintenant dans python3

Ligne 91 remplacer python2.7 par python3

Ligne 101 retirer l’installation des packages Linux qui n’existent plus

% ./install 4wd ydlidar (Faire % sudo apt-get install python3-pip si une erreur avec pip apparait)

Règles UDEV

Aller à <http://raspbian.raspberrypi.org/raspbian/pool/main/p/python-gudev>

Récupérer le fichier python-gudev\_147.2-3\_armhf.deb

Cliquer sur le fichier téléchargé, répondre « oui » à la demande d’installation

ATTENTION : Cela ne fonctionne que sur une raspberry pi avec un processeur ARM.

Pour une VM, python-gudev peut être téléchargé ici : <https://packages.debian.org/stretch/amd64/python-gudev/download>

Ou

% sudo apt-get install python-gudev

% rosrun lino\_udev lino\_udev.py

Les filtres USB doivent être inscrit dans la VM (dans Virtual Box : configuration -> USB -> ajouter filtre). Ils devraient apparaitre automatiquement quand la carte est branchée.

Plugger la teensy dans un des ports USB de la Raspberry

Répondre « Y » à la question « Create uDev name ? »

Répondre « linobase » à la question « What do you want to name this port ? »

Plugger le lidar dans un des ports USB de la Raspberry

Répondre « Y » à la question « Create uDev name ? »

Répondre « linolidar » à la question « What do you want to name this port ? »

Taper CTRL-C pour sauvegarder le fichier 58-lino.rules

Recopier le fichier dans les règles uDev

% sudo cp 58-lino.rules /etc/udev/rules.d

Redémarrer le service uDev (ou rebooter)

% sudo service udev reload

% sudo service udev restart

Customiser les codes sources[[16]](#footnote-16) pour les adapter au robot (les données sont celles du 1er prototype)

% roscd linorobot/teensy/firmware/lib/config/

Modifier le fichier lino\_base\_config.h

KP = 3.0

KD = 2.5

KI = 0,001

LINO\_BASE = SKID\_STEER

USE\_L298\_DRIVER

USE\_MPU6050\_IMU

DEBUG = 0

MAX\_RPM = 21

COUNTS\_PER\_REV = 5462

WHEEL\_DIAMETER = 0.12

PWM\_BITS = 8

LR\_WHEELS\_DISTANCE = 0.28

FR\_WHEELS\_DISTANCE = 0.22

% roscd linorobot/launch/include/imu

Modifier le fichier imu.launch (mettre le 2ème paramètre à faux)

<param name="use\_mag" value="false"/>

Recompiler les codes (et flasher le programme dans la teensy à la fin)

% roscd linorobot/teensy/firmware

% platformio run -t upload

Si le programme demande de reset la carte, vérifier dans la VM en bas à droite sur l’icône USB que la carte est bien sélectionnée.

Calibrer la centrale inertielle

Dans teensy/firmware/lib/config/lino\_base\_config.h, sélectionner le MPU6050.

Si il y a une erreur en lancent move\_base\_4wd alors retirer le « / » dans les fichiers global\_costmap\_params.yaml et local\_costmap\_params.yaml

# Test Lidar

% roslaunch linorobot laser.launch

Puis lancer Rvis pour visualiser les données :

% rviz

Dans rvis, passer FixedFrame de « map » à « laser ».

En bas, appuyer sur « Add » puis sélectionner « Laser scan » et le topic /scan. Les données du lidar devraient apparaitre.

# Création d’une carte de l’environnement

Lancer les packets de base :

% roslaunch linorobot bringup.launch

Lancer les packets de mapping:

% roslaunch linorobot slam.launch

Piloter le robot:

% rosrun teleop\_twist\_keyboard teleop\_twist\_keyboard.py

Faire rouler le robot dans l’environnement. La construction de la carte peut être visualiser dans Rviz.

Une fois la cartographie terminée, enregistrer la carte :

% rosrun map\_server map\_saver -f ~/linorobot\_ws/src/linorobot/maps/map

Pour utiliser la carte créée, il faut le préciser dans navigate.launch.

# Navigation autonome

Lancer les paquets de base :

% roslaunch linorobot bringup.launch

Lancer les packets de navigation:

% roslaunch linorobot navigation.launch

Ouvrir Rviz avec le fichier de configuration « navigate.rviz »

Si la data de la raspberry et l’ordinateur ne sont pas synchronisé, il risque d’y avoir un bug.

# Pour compiler un paquet perso

% catkin\_make

% source devel/setup.bash (Faire le source à chaque nouveau build de paquet)

La commande suivante permet de compiler seulement un paquet :

% sudo ./src/catkin/bin/catkin\_make\_isolated --pkg <package\_name>

On peut faire même chose avec catkin\_make.

Il est aussi possible de rajouter les mêmes options que plus haut dans ce document pour compiler le paquet en « Release »

# Connecter ROS avec plusieurs machines

ROS master ne doit tourner que sur une seule machine. Ce sera le robot.

Robot : 192.168.202.103

PC : 192.168.202.101

Sur le robot :

% export ROS\_IP=192.168.202.103

% export ROS\_MASTER\_URI=http://192.168.202.103 :11311

Sur le PC à distance

:

% export ROS\_IP=192.168.202.101

% export ROS\_MASTER\_URI= http://192.168.202.103 :11311

Dans le fichier /etc/hosts du PC il faut ajouter les IP et les noms des hôtes :

% nano /etc/hosts

Ajouter deux lignes :

192.168.202.101 debian

192.168.202.103 raspberrypi

Les noms des hôtes sont obtenus via la commande hostname.

# Annexe 1 – Trucs et astuces

Pour faire une sauvegarde d’une SDcard sous windows : win32diskImager,

<https://sourceforge.net/projects/win32diskimager/>

Pour faire une sauvegarde d’une SDcard sous Linux : dd

% fdisk –l pour lister les disques montés

% dd if=/dev/<sdcard-device> of=<image.file> sauvegarde

% dd if=<image.file> of=/dev/<sdcard.device> restauration

Attention /dev/<sdcard.device> ne doit pas être monté pour la restauration

1. « Calcul de couple des moteurs électriques (à balais, CC) pour un robot mobile », Pierre Bomel, Lab-STICC, Université de Bretagne Sud, Lorient, avril 2021 [↑](#footnote-ref-1)
2. ## <https://www.lextronic.fr/roues-et-chenilles-3853> ou bien « [Ensemble de Quatre Roues Tout Terrain Dagu](https://www.robotshop.com/be/fr/ensemble-quatre-roues-tout-terrain-dagu.html) »  chez Robotshop. Ces roues peuvent être accouplées à des arbres en D de diamètres 4 mm avec les moyeux livrés. Pour un arbre en D de 6 mm de diamètre, il faut commander d’autres moyeux. Par exemple le modèle « [Moyeu de Montage Hexagonal HUB-12 Lynxmotion - 6mm » chez Robotshop.](https://www.robotshop.com/be/fr/moyeu-montage-hub-12-lynxmotion.html)

   [↑](#footnote-ref-2)
3. # 4WD Wild Thumper de la société Dagu, ou bien chez generationrobots.com

   [↑](#footnote-ref-3)
4. « Choix des moteurs électriques (à balais, CC) pour un robot mobile », Pierre Bomel, Lab-STICC, Université de Bretagne Sud, Lorient, mai 2021 [↑](#footnote-ref-4)
5. [www.robotshop.com](http://www.robotshop.com) (l’auteur n’a aucun intérêt personnel à recommander cette société ☺ [↑](#footnote-ref-5)
6. Rappel : 1 oz.in = 7,06 10e-3 N.m, 1 gf.cm = 9,81 10e-5 N.m [↑](#footnote-ref-6)
7. Vendu avec les pièces de montage pour moteurs IG32 et les moyeux hexagonaux de 12 mm (robotshop) [↑](#footnote-ref-7)
8. On peut très facilement généraliser à n phases ayant chacune une probabilité (ou une fréquence d’apparition relative) et dont la somme des probabilités est égale à 1. Cela permettrait de modéliser un profil de consommation ayant plus de deux phases distinctes. [↑](#footnote-ref-8)
9. Ref 150-1559, 12V, 24 Ah, chez RadioSpares [↑](#footnote-ref-9)
10. www.pjrc.com [↑](#footnote-ref-10)
11. www.ydlidar.com [↑](#footnote-ref-11)
12. # Convertisseur CC/CC, 18 32V c.c. vers 12V c.c., 240W, RS ref 179-3349

    [↑](#footnote-ref-12)
13. RS ref 466-7590 [↑](#footnote-ref-13)
14. Réalisée avec Thinkercad, www.tinkercad.com [↑](#footnote-ref-14)
15. <http://wiki.ros.org/ROSberryPi/Installing> ROS Melodic on the Raspberry Pi [↑](#footnote-ref-15)
16. Un document séparé à été rédigé pour permettre le test et le réglage du PID. « Réglage du PID », Pierre Bomel, Université de Bretagne Sud, Lab-STICC, Lorient, France, mai 2021 [↑](#footnote-ref-16)