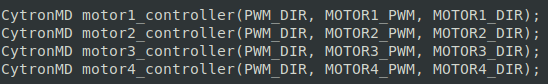
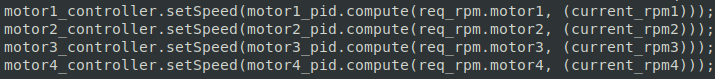
-Ludovic Chavalarias, Lab-STICC, Lorient

# Passage à la version aluminium

* Télécharger la librairie pour les nouveaux drivers de moteurs :
  + https://github.com/CytronTechnologies/CytronMotorDriver
* Copie de la librairie pour les nouveaux drivers dans « Linorobot/teensy/firmware/lib/motor/ »
* Modifier le fichier « firmware.ino » :
  + % roscd linorobot/teensy/firmware
  + % nano src/firmware.ino
  + Rajouter « #include « CytonMotorDriver.h » »
  + Déclaration des contrôleurs :

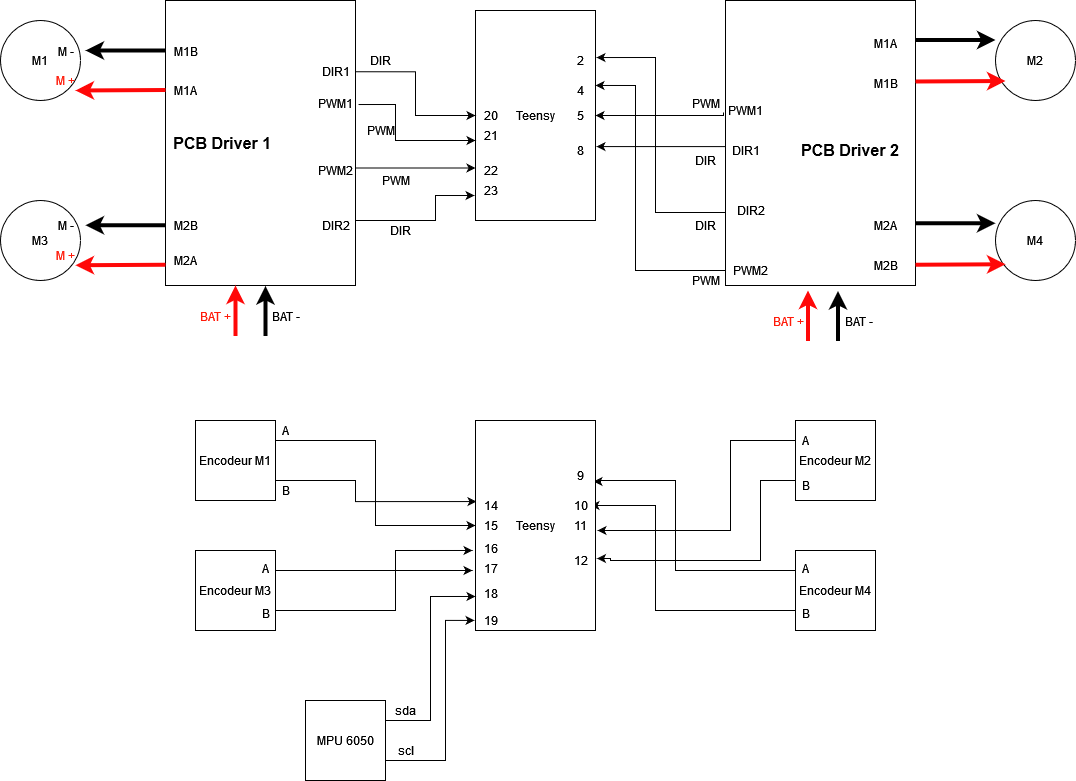


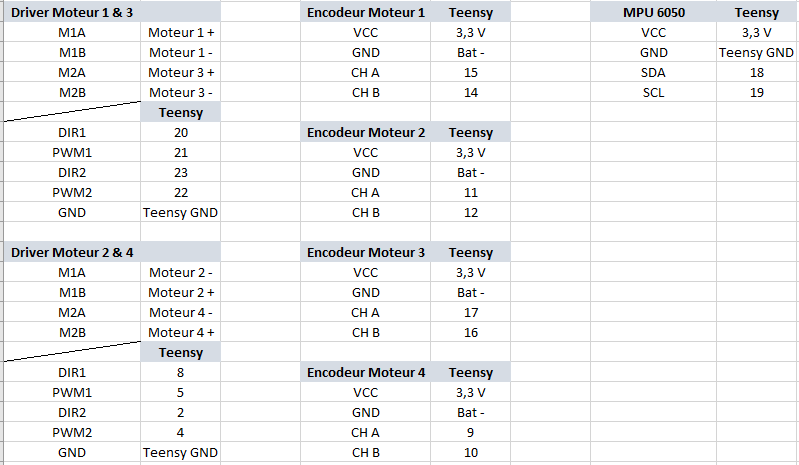
* + Modifier l’appel de la fonction d’envoi de commande au driver (ces appels sont dans la fonction moveBase ()) :



* Modifier le fichier lino\_base\_config.h
* % roscd linorobot/teensy/firmware/lib/config/
* KP = 1.2
* KD = 0.01
* KI = 0.3
* LINO\_BASE = SKID\_STEER
* USE\_L298\_DRIVER
* USE\_MPU6050\_IMU
* DEBUG = 0
* MAX\_RPM = 312
* COUNTS\_PER\_REV = 537.6
* WHEEL\_DIAMETER = 0.121
* PWM\_BITS = 8
* LR\_WHEELS\_DISTANCE = 0.34
* FR\_WHEELS\_DISTANCE = 0.27
* Dans ce même fichier, inverser les connections pour les encodeurs des moteurs 2 & 4
  + Motor2\_encodeur\_A = 12 ; motor2\_encodeur\_B = 11
  + Motor4\_encodeur\_A = 10 ; motor4\_encodeur\_B = 9
* Brancher les moteurs 2 & 4 de façon inverse aux moteur 1 & 3 (connexion aux drivers, voir schéma)

Schéma des connections :





* Les transformes sont les suivantes :
  + Dans « laser.launch » : -0.159 0 0.205 0 0 0
  + Bringup.launch : 0 0 0.12 0 0 0
  + imu.launch : 0.144 0.0 0.105 -1.5708 0 1.5708

Il peut y avoir un problème de transforme lorsque l’on souhaite afficher les données du lidar en saisissant « map » comme frame fixe dans Rviz (unkown tranform error). Cela est dû à un bug de ydlidar :

* % roscd ydlidar/src/
* % nano ydlidar\_node.cpp
  + Changer :
    - Scan\_msg.scan\_time = scan.config.scan\_time;
    - Scan\_msg.time\_increment= scan.config.time\_increment;
  + En :
    - Scan\_msg.scan\_time = 1/\_frequency;
    - Scan\_msg.time\_increment= 1/((float)samp\_rate\*1000) ;
* Revenir dans le workspace catkin :
* % cd ../..
* Compiler le packet
* % catkin\_make ydlidar\_node

# Configuration du PID

Sur l’ordinateur du robot :

* % roslaunch linorobot minimal.launch
* % rosrun lino\_pid pid\_configure

Sur le pc de dev :

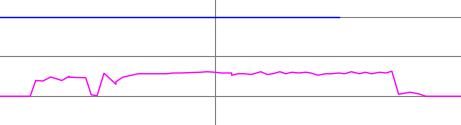
* % rosrun teleop\_twist\_keyboard teleop\_twist\_keyboard.py
* Ouvrir rqt
* Importer la perspective lino\_pid.perspective
* Cliquer sur pid\_config et configuer le PID

Dans rqt :

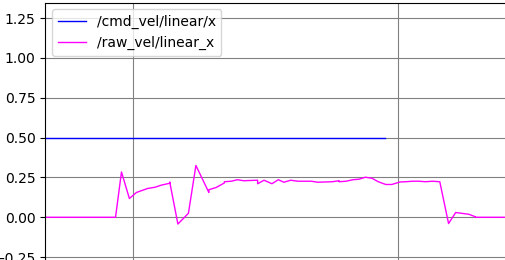
* Rajouter le topic /raw\_vel/linear\_x pour la vitesse actuelle
* Rajouter le topic /cmd\_vel/linear/x pour la vitesse demandée

**Proportionnel**

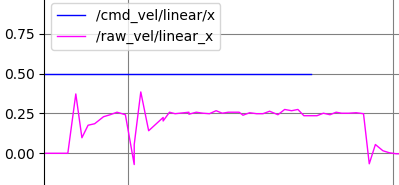
P = 0.6



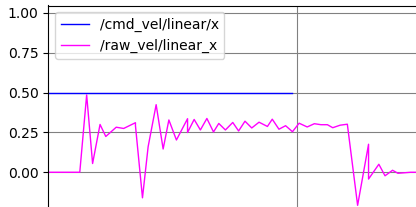
P = 1



P=1.2



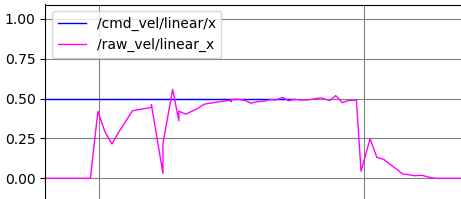
P=1.6



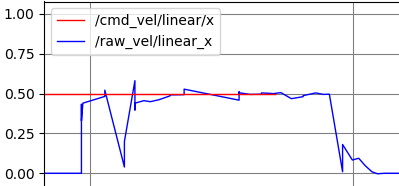
On prend P = 1.2

**Intégral** Avec P = 1.2

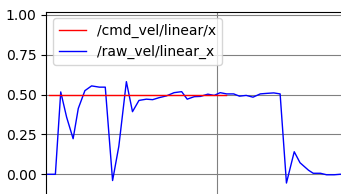
I = 0.2



I = 0.3



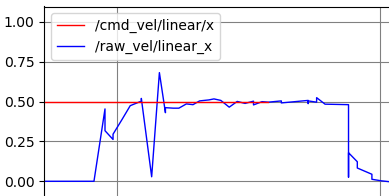
I = 0.5



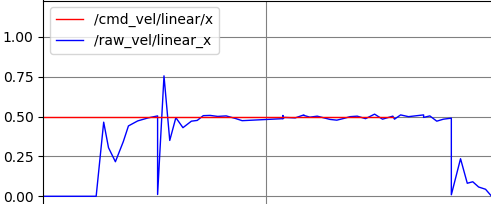
On prend I = 0.3

**Dérivative** P = 1.2, I = 0.3

D = 0.01



D= 0.05



On prend D = 0.01

# Problème de connexion

Il se peut qu’il y est un problème de connexion entre le pc vers la raspberry. Cela est dû au type de connexion.

* Dans la configuration réseau de la VM, passer de NAT à connexion en pont
* Dans la VM, aller dans paramètre réseau et donner une IP fixe à la VM dans le réseau correspondant.
* Changer la variable ROS\_IP de la VM pour lui donner l’IP choisie
  + % export ROS\_IP=ip\_choisie

Le problème devrait être résolu.

# Effectuer une mission SLAM

**Sans waypoints :**

Sur le robot :

* % roslaunch linorobot bringup.launch
* % roslaunch linorobot navigate.launch

Lancer Rviz et donner un objectif avec l’outil 2D nav goal.

**Avec waypoints :**

Installation du paquet de waypoints sur le pc (il est aussi possible de l’installer sur la raspberry) :

* Aller dans le workspace catkin
* % cd ~/linorbot\_ws/src
* % git clone <https://github.com/halejo-as/waypoints.git>
* % cd ..
* % catkin\_make
* % source devel.setup.bash

Sur le robot :

* % roslaunch linorobot bringup.launch
* % roslaunch linorobot navigate.launch

Sur le pc :

* % rosrun waypoints waypoints\_server

Lancer Rviz :

* Dans la barre des outils, appuyer sur le signe + et ajouter l’outils waypoints
* Ensuite faire Panels -> AddNewPanels et ajouter waypointPanel
* Ajouter ensuite la visualisation du topic waypoints\_rviz

On peut ensuite ajouter des waypoints, créer des groupes et lancer le parcours d’un groupe de waypoints en boucle

**Utiliser le paquet de waypoints sans connexion réseau**

Il faut avoir fait une sauvegarde de point de passage avec rviz au préalable.

Il faut aussi dans « .bashrc » passer la variable *offline* à « true ». Cela indique à ros que le ROS\_MASTER se trouve au local host et il ne passe donc pas par le réseau pour le ping. Il est ensuite possible de lancer les commandes suivantes via ssh, mais il n’est pas possible de lancer rviz sur le pc portable.

Le principe est de faire appel aux services proposé par le paquet waypoints via des commandes de terminal :

Charge des waypoints et les groupes :

* % rosservice call /waypoint\_server/load\_wp « file\_name : ‘ ‘ »

Appuyer sur tab pour la complétion automatique

Le nom du fichier est celui inscrit lors de la sauvegarde des points (sans les suffixes “\_wp” ou “\_gr”). Les fichiers sont présents le dossier *files* du paquet waypoints.

Pour lancer une mission de groupe, il faut faire appel au service run\_wp de la même façon. Pas besoins de preciser le champ « wp\_name »

Fichier « etage\_clockwize » :

* Parcours de l’étage dans le sens des aiguilles d’une montre
* Deux groupes :
  + etage\_clockwize\_bureau : fait le tour de l’étage et revient dans le bureau (premier point : a)
  + etage\_clockwize : fait uniquement le tour de l’étage (premier point : b)
  + clockwize\_less\_point

Un fichier « set\_pose.bh » est présent à la racine de la raspberry. Il envoie un message au topic */initial\_pose* pour indiquer la position initiale du robot si elle a besoins d’être modifier (ne pas oublier d’indiquer les covariances dans la commande !)

## Exemple des commandes à effectuer pour une mission autour de l’étage en offline

% roslaunch linorobot bringup.launch

% roslaunch linorobot navigate.launch

% rosrun waypoints waypoints\_server

(Les trois commandes au-dessus peut être remplacer en lançant autonomous\_nav.launch)

% bash set\_pose.bh #optionnel

% rosservice call /waypoint\_server/load\_wp “file\_name: ‘etage\_clockwize’”

% rosservice call /waypoint\_server/run\_wp “wp\_name :’’

gr\_name: ‘etage\_clockwize\_bureau’

loop: false

index: 0”

# Synchronisation des horloges

Il est possible qu’il faille synchroniser les horloges de la VM et de la raspberry.

Pour ce faire :

* Dans la VM :
* % sudo nano /etc/ntp.conf
* A la fin du fichier ajouter :
  + Server 127.127.1.0
  + Fudge 127.127.1.0 stratum 1
* Décommander la ligne « broadcast 192.168.\*\*\*.255 » et ajoute la bonne adresse de broadcast
* Dans la raspberry :
* % sudo nano /etc/ntp.conf
* Décomenter les lignes :
  + Disable auth
  + broadcastclient

# Bloc relais pour le lidar

Il a fallu rajouter un relai pour le lidar car la raspberry de démarrais pas lorsque l’on branchait la batterie. Cela était dû au fait que tous les périphériques était mis sous tension au même moment. Une solution était donc l’ajout d’un relais 5V contrôler par la teensy.

Comme les GPIO de la teensy ne sont capable que de sortir du 3.3V, le relais a besoins d’un transistor pour être piloté.

# Annexe

Document methodo de configuration de la stack de navigation :

<https://kaiyuzheng.me/documents/papers/ros_navguide.pdf>