



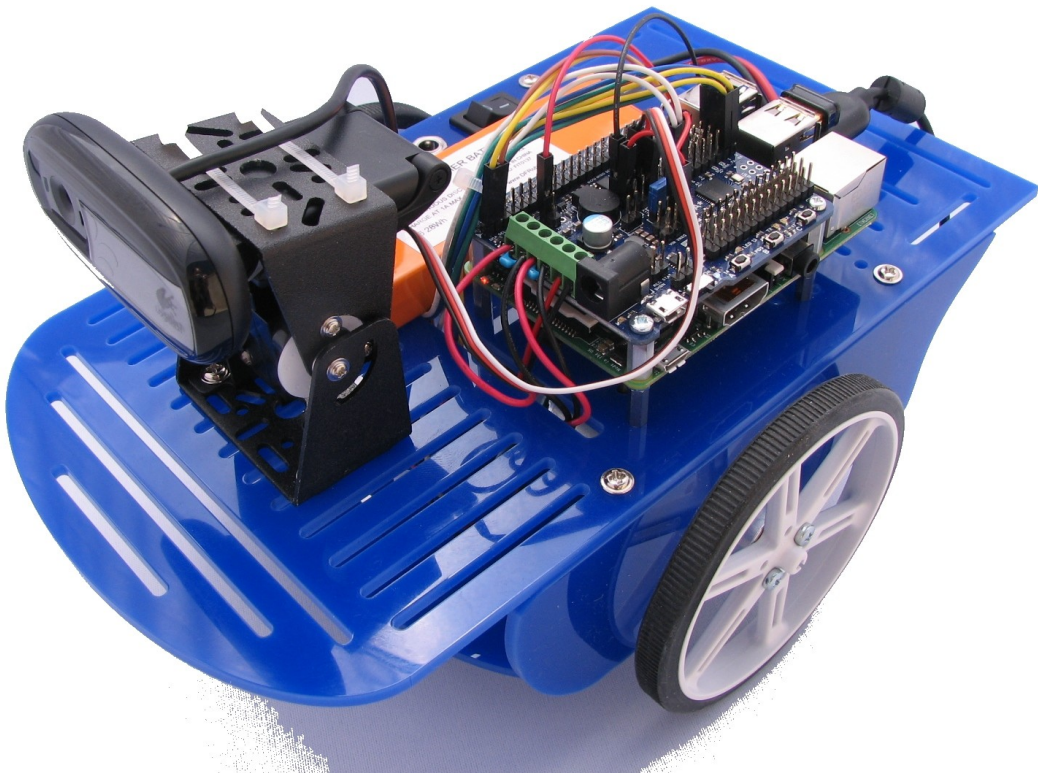
---

# Geeros Raspberry C / Python

---

## Documentation complète

03/11/2020  
Version 2.3



## Table des matières

1 Introduction.....	5
2 Matériel inclus.....	6
3 Conformité.....	6
4 Présentation du matériel.....	7
4.1 Carte contrôleur de robot Pololu A-Star 32u4.....	11
4.2 Moteurs électriques.....	13
4.2.1 Équations et fonctions de transfert.....	13
4.2.2 Calcul de la vitesse de rotation.....	14
4.2.3 Remarque sur le comptage des impulsions.....	16
4.3 Batterie.....	16
4.4 Communication Wifi.....	17
5 Installation des logiciels.....	18
5.1 Installation de l'environnement de développement Arduino.....	18
5.1.1 Installation principale.....	18
5.1.2 Installation du support de la carte A-Star.....	19
5.1.3 Installation de la bibliothèque complémentaire FlexiTimer2.....	19
5.1.4 Installation de la bibliothèque complémentaire EnableInterrupt.....	20
5.1.5 Installation de la bibliothèque complémentaire MPU9250.....	20
5.1.6 Installation de la bibliothèque complémentaire SoftwareServo.....	20
5.1.7 Choix de la carte Arduino dans l'IDE.....	21
5.1.8 Programmation de la carte Arduino.....	21
5.2 Installation et première utilisation du logiciel MyViz.....	22
6 Mise en œuvre de l'ensemble.....	23
6.1 Précautions d'emploi.....	23
6.1.1 Connexions d'alimentation sur la carte A-Star.....	23
6.1.2 Recharge de la batterie.....	23
6.1.3 Précautions d'utilisation.....	24
6.2 Première utilisation.....	25
6.2.1 Pilotage par ordinateur.....	25
6.2.2 Pilotage par smartphone ou tablette.....	29

7 Détails techniques.....	31
7.1 Asservissements de type PI.....	31
7.2 Asservissements de type PID.....	32
7.3 Fonctionnement d'un gyropode.....	35
7.3.1 Maintien en équilibre.....	35
7.3.2 Mouvement.....	36
7.3.3 Démarrage.....	36
7.3.4 Arrêt.....	37
7.3.5 Équations.....	37
7.3.6 Asservissements.....	38
7.4 Tracés graphiques.....	39
8 Activités réalisables avec le système en mode Raspberry Pi / Python.....	41
8.1 Contrôle de vitesse avec asservissement de verticalité et pilotage en Wifi.....	42
8.1.1 Présentation.....	42
8.1.2 Expérimentations possibles.....	43
8.2 Commande des moteurs en tension.....	44
8.2.1 Présentation.....	44
8.2.2 Expérimentations possibles.....	45
8.3 Asservissement des moteurs en vitesse.....	46
8.3.1 Expérimentations possibles.....	48
8.4 Pilotage du robot sans asservissement de verticalité.....	49
8.4.1 Expérimentations possibles.....	50
8.5 Pilotage du robot et visualisation des images de la Webcam.....	51
8.5.1 Expérimentations possibles.....	52
8.6 Asservissement d'angle.....	53
8.6.1 Présentation.....	53
8.6.2 Expérimentations possibles.....	54
8.7 Pilotage par programme Python.....	55
8.7.1 Principe de fonctionnement.....	55
8.7.2 Fonctions de l'API.....	56
8.7.3 Exemples d'utilisation de l'API.....	60
9 Activités réalisables avec le système en mode Arduino / C.....	63
9.1 Commande des moteurs en tension.....	64

9.2 Asservissement des moteurs en vitesse.....	64
9.3 Trajectoire pré-programmée avec asservissement de verticalité.....	65
9.3.1 Présentation.....	65
9.4 Trajectoire pré-programmée sans asservissement de verticalité.....	66
10 Support technique.....	67

# 1 Introduction

Geeros Raspberry C / Python est un robot gyropode ouvert et open-source, un concentré de technologie vous permettant de faire de nombreuses expériences. Il est basé sur une carte Raspberry Pi, une carte compatible Arduino, différents capteurs et deux moteurs électriques permettant de le mettre en mouvement tout en assurant son maintien en équilibre.

Il intègre une liaison Wifi vous permettant de le piloter à distance à partir d'un ordinateur, d'un smartphone ou d'une tablette.

Il embarque enfin une Webcam capable de faire de la transmission vidéo en temps-réel des images prises par le robot.

Les programmes associés à ce système sont téléchargeables sur notre site Web à l'adresse suivante :

[https://www.3sigma.fr/geeros4raspberrypi/fichiers/Geeros\\_Raspberry\\_C\\_Python.zip](https://www.3sigma.fr/geeros4raspberrypi/fichiers/Geeros_Raspberry_C_Python.zip).

Cette archive contient la présente documentation et 3 sous-répertoires :

- Arduino : programmes Arduino permettant de réaliser des expériences utilisant cette carte exclusivement (voir chapitre 9) ou en liaison avec la Raspberry Pi (voir chapitre 8)
- MyViz : tableaux de bord des différentes expériences réalisables avec le robot (voir les chapitres 5 pour l'installation du logiciel, 8 pour les expériences utilisant la carte Raspberry Pi et 9 pour celles utilisant la carte compatible Arduino seule)
- programmes\_python : programmes stockés sur la carte Raspberry Pi (dans le répertoire /root/programmes\_python) pour les expériences mettant en œuvre le robot programmé en Python (voir chapitre 8)

## 2 Matériel inclus

Ce robot est livré monté et **fonctionnel (testé par nos soins avant la livraison)**. Il est composé des éléments suivants:

- Le robot gyropode lui-même
- 1 chargeur de batterie
- 1 câble mini USB pour la programmation de la carte compatible Arduino
- 1 clé allen pour visser ou dévisser les roues
- 2 boules omnidirectionnelles permettant de stabiliser Geeros afin de faire des expériences sans se préoccuper de l'asservissement de verticalité

## 3 Conformité

Le robot Geeros, **dans sa configuration livrée aux clients**, est conforme à la directive 1999/EC.

## 4 Présentation du matériel

Les différents éléments constituant Geeros Raspberry C / Python sont présentés ci-dessous :

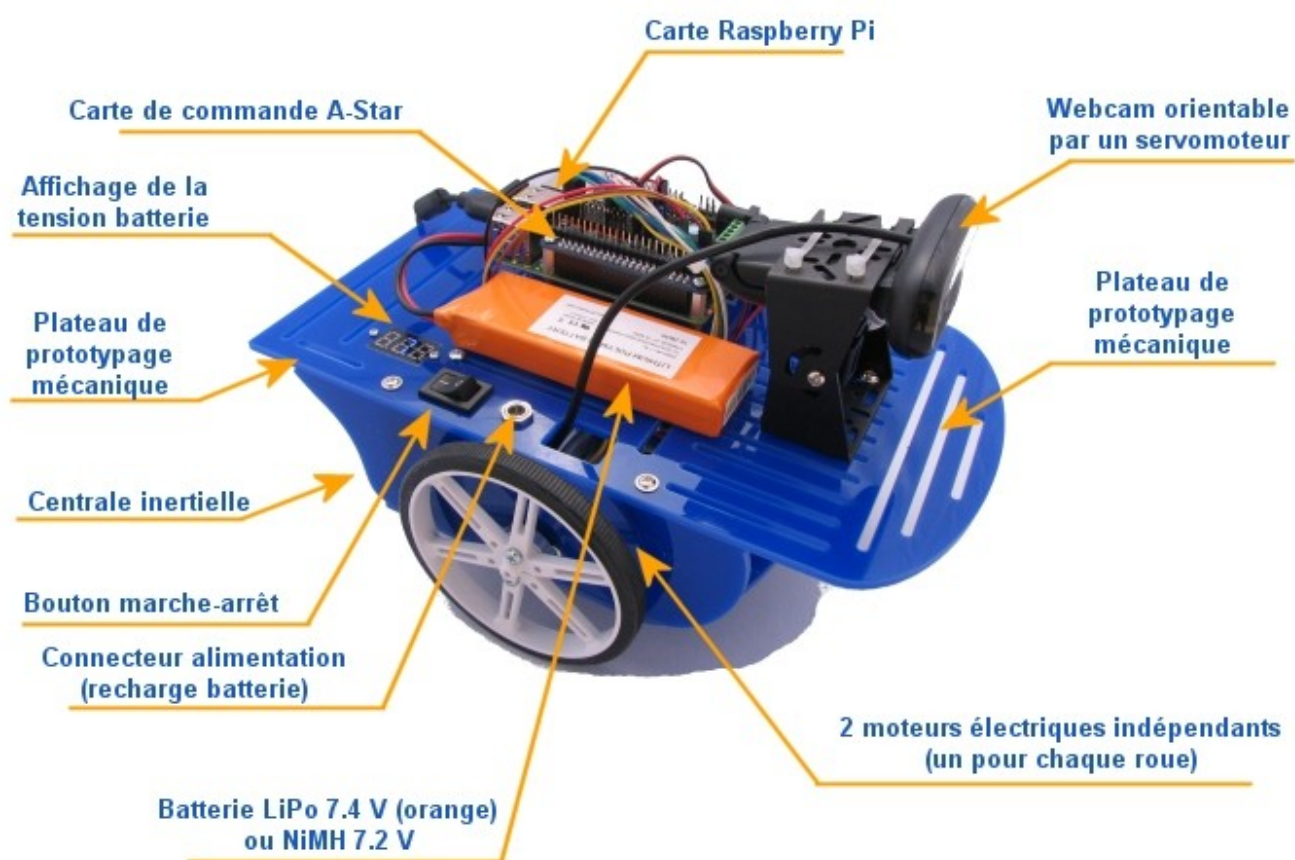
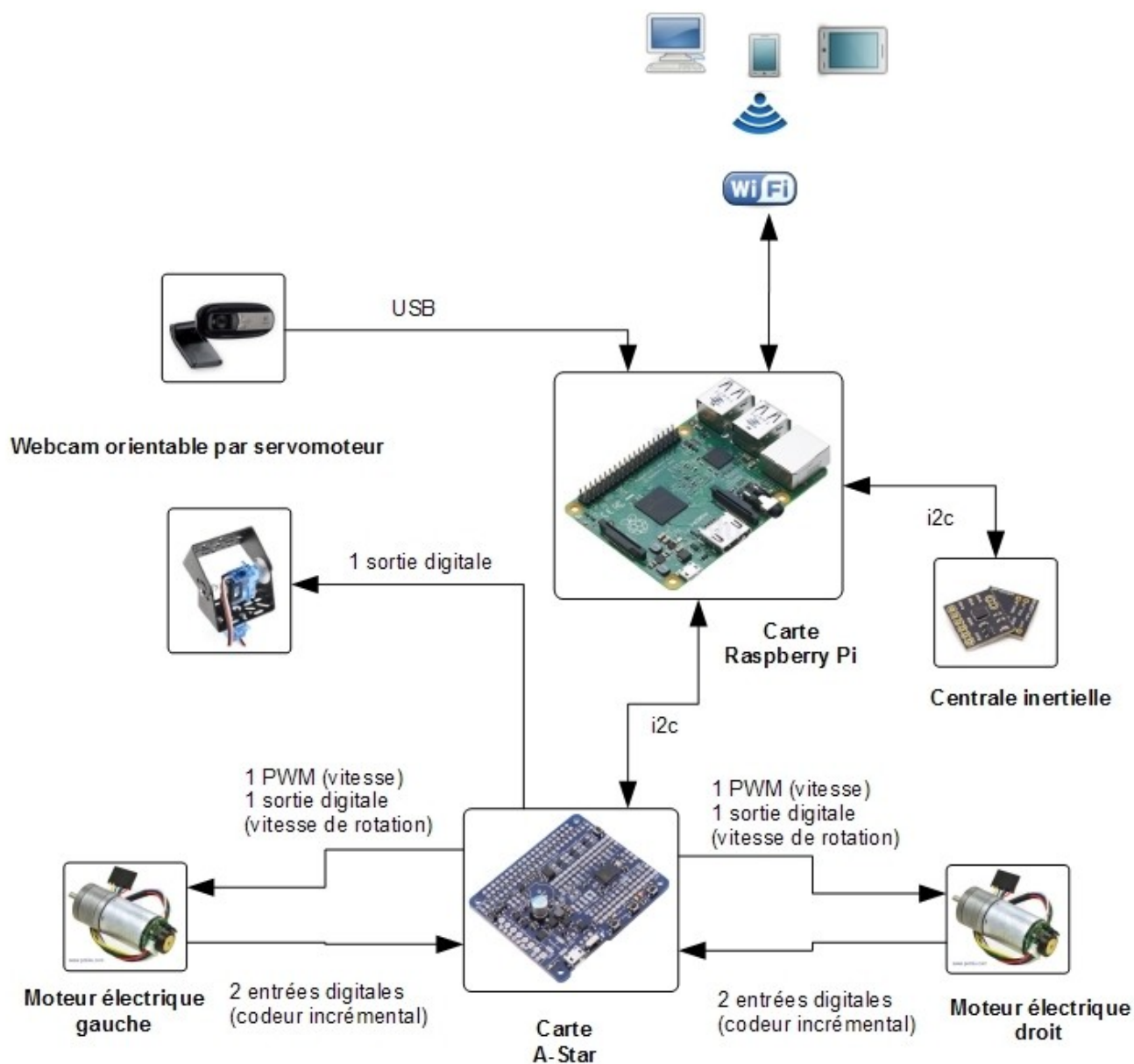
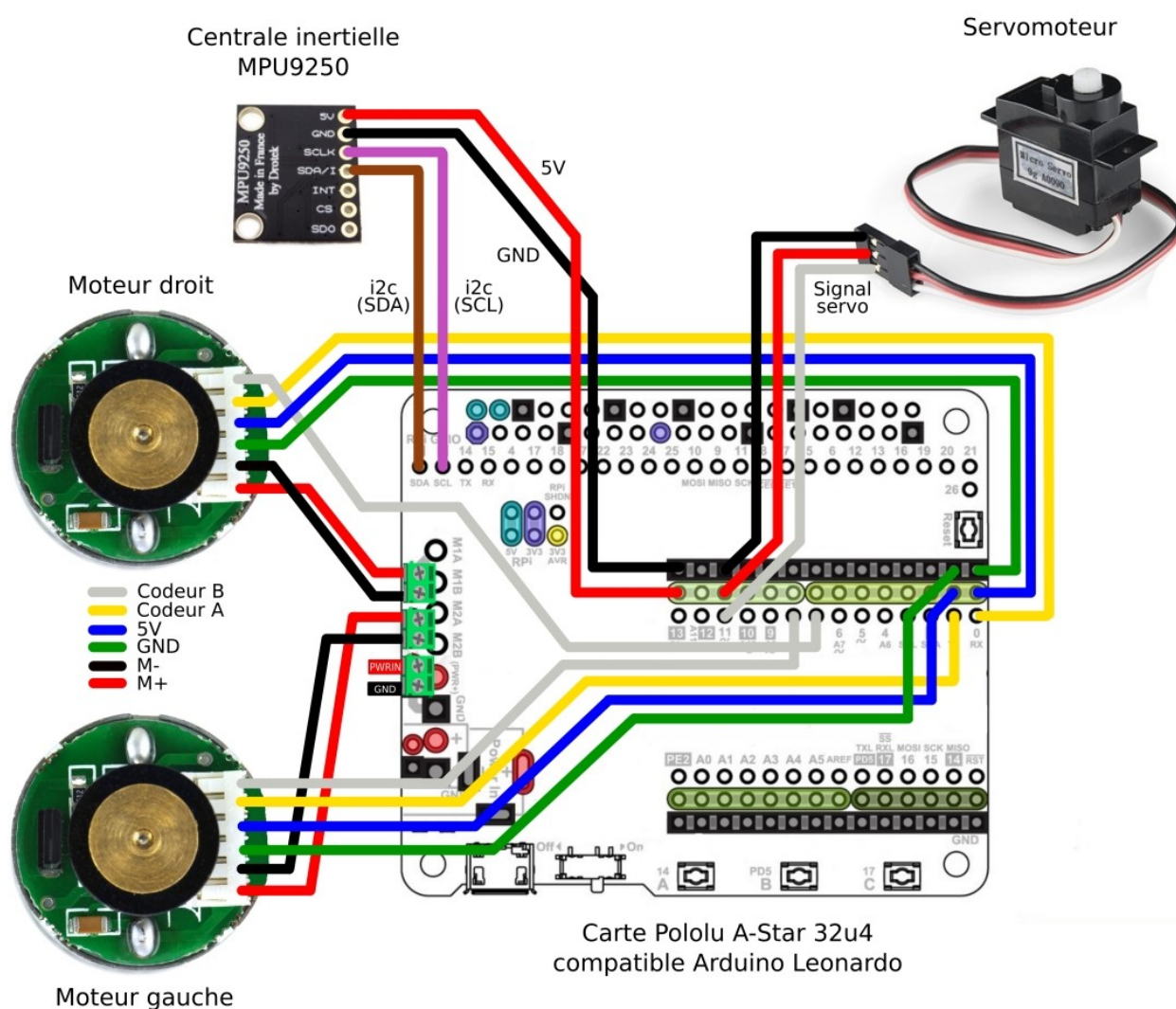


Diagramme des échanges de signaux :





Les connexions entre la carte A-Star et les différents éléments du robots sont représentées ci-dessous :



Les références des principaux composants sont les suivantes :

- carte Raspberry Pi :  
<http://boutique.3sigma.fr/88-raspberry-pi-3-mod%C3%A8le-b.html>
- carte Pololu A-Star (voir plus loin) :  
<http://boutique.3sigma.fr/96-contr%C3%B4leur-de-robot-a-star-32u4.html>
- centrale inertielle 9 axes basée sur le composant MPU9250  
<https://drotek.com/shop/fr/421-mpu9250-gyro-accelerometre-magnetometre.html>
- moteurs électriques :  
<http://boutique.3sigma.fr/12-moteur-%C3%A0-courant-continu-r%C3%A9ducteur-341-codeur-incremental-48-cpr.html>
- supports moteurs :  
<http://boutique.3sigma.fr/16-paire-de-support-de-moteur-pololu-25d.html>
- roues :  
<http://boutique.3sigma.fr/18-paire-de-roues-blanches-pololu-90x10mm.html>
- moyeux :  
<http://boutique.3sigma.fr/17-paire-de-moyeux-aluminium-universels-pololu-m3-pour-arbre-de-4mm.html>
- support pan-tilt :  
<http://boutique.3sigma.fr/48-support-pantilt.html>
- servomoteur :  
<http://boutique.3sigma.fr/47-petit-servomoteur.html>
- webcam :  
<http://boutique.3sigma.fr/85-webcam-logitech-c170.html>

Certains éléments ou points techniques sont présentés plus en détails ci-dessous.

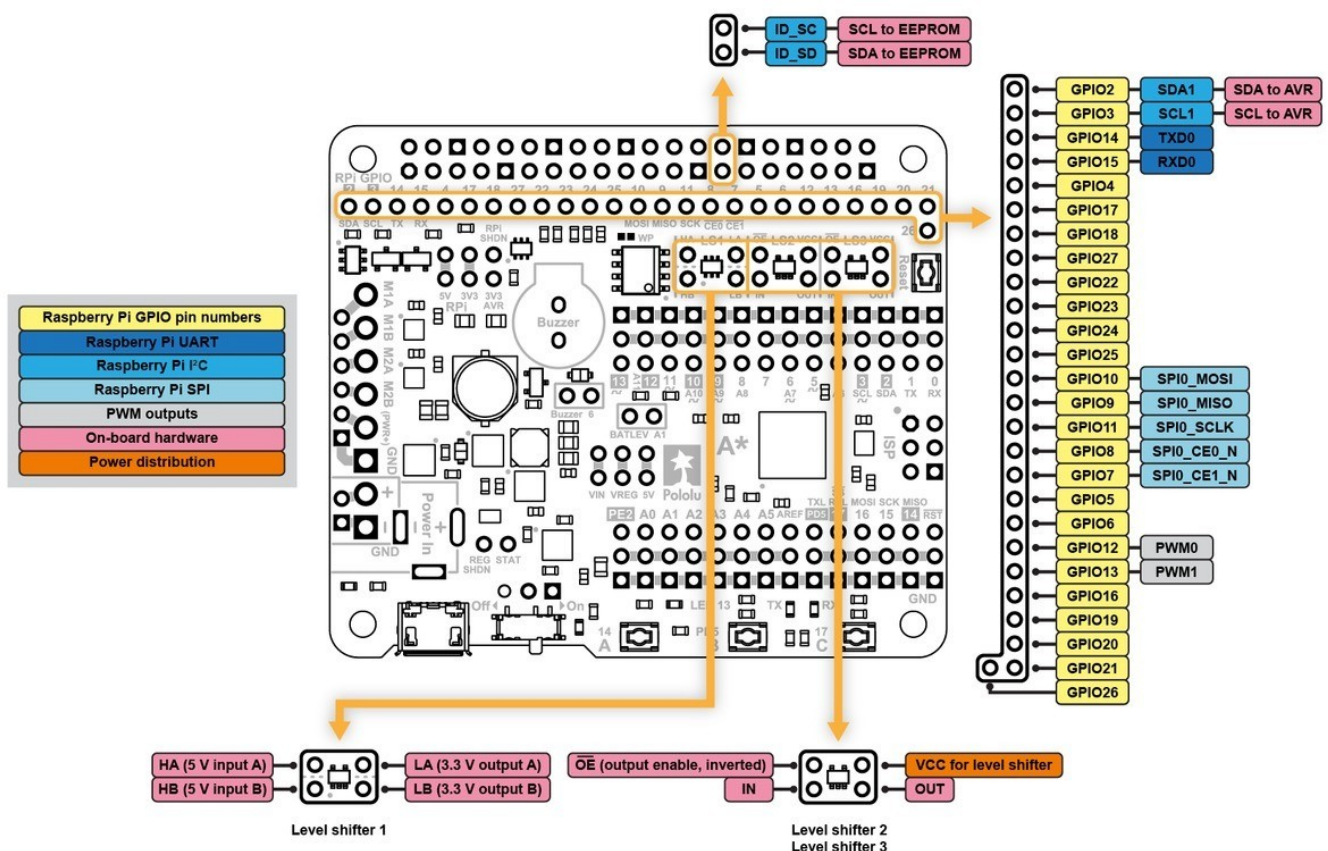
## 4.1 Carte contrôleur de robot Pololu A-Star 32u4

Le contrôleur de robot A-Star 32U4 avec connecteur Raspberry Pi est une carte programmable idéale pour les applications robotiques. Il peut fonctionner ou bien comme contrôleur auxiliaire monté sur une carte Raspberry Pi ou bien de façon autonome.

L'A-Star est basée sur le micro-contrôleur Atmel AVR ATmega32U4. Elle intègre également deux drivers de moteur électrique délivrant jusqu'à 1.8 A en continu par voie. Une gestion d'alimentation intelligente incluant un régulateur de tension très efficace lui permet d'être alimentée à partir de tensions d'entrée comprises entre 2.7 et 11V.

Cette carte est conforme à la spécification Raspberry Pi HAT, ce qui lui permet d'être utilisée en complément d'une Raspberry Pi, montée sur le connecteur 40 broches. Des convertisseurs de niveau intégrés facilitent la communication I<sup>2</sup>C entre les deux cartes. L'A-Star est par ailleurs conçue pour alimenter la Raspberry Pi en 5V. Dans cette configuration, la Raspberry Pi peut gérer les tâches de haut niveau, laissant le soin à l'A-Star d'exécuter les tâches bas-niveau nécessitant les qualités d'un micro-contrôleur (gestion rapide des E/S, notamment, comme la mesure de vitesse à partir de codeurs incrémentaux).

Le brochage « Raspberry Pi » est présenté ci-dessous :





De nombreuses ressources (en anglais) sont par ailleurs disponibles sur la page de présentation du produit chez son fabricant (Pololu) : <https://www.pololu.com/product/3117>.

## 4.2 Moteurs électriques

### 4.2.1 Équations et fonctions de transfert

Ce robot embarque deux moteurs à courant continu 6V, de rapport de réduction 34:1, avec codeur incrémental 48 CPR (Counts Per Revolution).

Les équations d'un moteur sont les suivantes:

$$J_m n^2 \frac{d}{dt} \omega_m(t) + d \omega_m(t) - n K i_m(t) = Tr(t) \quad (4.1)$$

$$L \frac{d}{dt} i_m(t) + R_m i_m(t) = V(t) - K n \omega_m(t) \quad (4.2)$$

Avec :

- $R_m$  : résistance électrique interne : 2.5  $\Omega$
- $L$  : inductance des enroulements : 3 mH
- $J_m$  : moment d'inertie du rotor : 3. 10<sup>-6</sup> kg.m<sup>2</sup>
- $K$  : constante de couple = constante de fem : 0.01 N.m/a
- $d$  : coefficient de frottement visqueux : 0.005 N.m.s/rad
- $\omega_m$  : vitesse de rotation de l'arbre de sortie du réducteur
- $i_m$  : courant dans le moteur
- $V$  : tension d'alimentation
- $Tr$  : couple résistant
- $n$  : rapport de réduction (34)

Ces paramètres ont été identifiés via notre système de commande de moteur électrique (<http://boutique.3sigma.fr/23-commande-de-moteur-%C3%A9lectrique.html>).



La fonction de transfert entre l'entrée  $V(t)$  et la sortie  $\omega_m(t)$  est la suivante :

$$\frac{Kn}{JLn^2s^2 + (JRn^2 + Ld)s + K^2n^2 + Rd} \quad (4.3)$$

La fonction de transfert entre l'entrée  $V(t)$  et la sortie  $i_m(t)$  est la suivante :

$$\frac{Jn^2s + d}{JLn^2s^2 + (JRn^2 + Ld)s + K^2n^2 + Rd} \quad (4.4)$$

#### 4.2.2 Calcul de la vitesse de rotation

Les moteurs à courant continu du robot sont dotés de codeurs incrémentaux afin de mesurer la vitesse de rotation des moteurs. Le codeur incrémental fournit deux signaux carrés en quadrature, comme sur la capture ci-dessous:



Ces deux signaux permettent de mesurer à la fois la vitesse et le sens de rotation. La mesure de la vitesse se fait simplement en comptant le nombre d'impulsions pendant un temps fixe. Les données du problème sont les suivantes :

- Le codeur est fixé à l'arbre moteur et non pas à l'arbre de sortie du réducteur (celui utilisé pour l'entraînement). Le rapport de réduction étant 34:1, l'arbre moteur fait 34 tours lorsque l'arbre « principal » en fait 1
- Le codeur génère 48 impulsions à chaque fois qu'il fait un tour. Cependant, la carte A-Star qui gère le comptage des impulsions ne dispose que de deux lignes d'interruptions matérielles (une pour chaque moteur). Par conséquent, on ne prend en compte qu'une interruption sur 2
- La cadence d'échantillonnage utilisée pour l'asservissement est de 0.01 s

Par conséquent, lorsque l'arbre principal fait un tour, le codeur génère :  $34 \times 24 = 816$  impulsions.

Si N est le nombre d'impulsions comptées en 0.01 s, la vitesse est (en rad/s, l'unité standard, sachant qu'un tour fait  $2 \times \pi$  radians) :

$$\frac{2\pi N}{0.01 * 816} \quad (4.5)$$

Attention : bien que le codeur soit placé sur l'arbre moteur, le calcul ci-dessus donne la vitesse en sortie du réducteur.

Un point très important concerne la résolution de la mesure, c'est-à-dire la plus petite valeur qu'il est possible de calculer. La formule est la suivante (en rad/s) :

$$\frac{2\pi}{Ts \text{ CPR } n} \quad (4.6)$$

avec :

- Ts : cadence d'échantillonnage
- CPR : nombre d'impulsions par tour du codeur
- n : rapport de réduction du moteur

Dans notre cas de figure, la résolution est la suivante :

$$\frac{2\pi}{0.01 * 816} = 0.77 \text{ rad/s} \quad (4.7)$$

Enfin, pour l'améliorer, nous faisons une moyenne sur les 10 derniers échantillons. Ceci introduit un retard (à prendre en compte dans les asservissements) mais permet d'avoir une résolution de l'ordre de **0.08 rad/s**.

### 4.2.3 Remarque sur le comptage des impulsions

Le comptage des impulsions revient à compter le nombre de fronts montants et descendants du signal jaune représenté sur l'image ci-dessus. Pour ce faire, la seule méthode viable consiste à brancher ce signal (le fil jaune du codeur utilisé) sur une des deux entrées « interruption matérielle » de la carte A-Star. Le fil blanc est branché sur une entrée digitale classique et les deux derniers fils (bleu et vert) seront respectivement branchés sur le 5 V et sur la masse du système.

L'intérêt d'une ligne d'interruption est qu'elle permet, comme son nom l'indique, d'interrompre le déroulement des calculs sur le processeur pour effectuer un traitement spécifique, en l'occurrence la mise à jour du compteur d'impulsions, avant de rendre la main à la boucle principale.

La seule « difficulté » est de savoir s'il faut incrémenter ou décrémenter le compteur dans le traitement de l'interruption. Il suffit pour cela d'observer les courbes ci-dessus, obtenues alors que le moteur tourne dans le sens positif. On constate que:

- Lorsque la voie A (en jaune) passe au niveau haut, la voie B (en bleu) est au niveau bas
- Lorsque la voie A passe au niveau bas, la voie B est au niveau haut

Quand le moteur tourne dans le sens positif, lors d'une interruption sur la voie A, les niveaux de A et B sont donc inversés. Pour connaître le sens de rotation du moteur, il faut donc mesurer le niveau de B lorsque l'interruption survient.

## 4.3 Batterie

En fonction des versions, la batterie intégrée au robot est soit une batterie LiPo (couleur orange), soit une batterie NiMH. Dans la cas d'une batterie LiPo, contrairement à ce que l'on trouve couramment en modélisme, cette batterie est « protégée » pour qu'elle ne puisse pas délivrer des courants trop forts.

**Attention : il est impératif de recharger le robot avec le chargeur fourni car il est adapté au type de batterie utilisé.**



## 4.4 Communication Wifi

Geeros embarque une carte Raspberry Pi avec Wifi intégré lui permettant d'être piloté par certains appareils possédant une connectivité Wifi.

Lorsqu'il est démarré et que la led verte de la caméra est allumée, la connexion est disponible. Geeros se comporte alors comme point d'accès Wifi autonome: un nouveau réseau Wifi devient alors visible sur vos appareils, avec le SSID « Geeros ». Ce réseau est sécurisé, le mot de passe est « geerosraspberrry ».

Notez les points suivants:

- Un appareil donné ne peut pas en général se connecter en même temps à deux réseaux Wifi différents: la connexion au réseau Geeros entraînera la déconnexion du réseau auquel vous étiez éventuellement connecté au préalable. Vous n'aurez alors plus accès à Internet sur votre appareil, sauf si celui-ci est
  - un ordinateur connecté en parallèle en filaire (liaison Ethernet). Attention cependant: si ce réseau Ethernet utilise la même plage d'adresse (192.168.0.xxx), vous devrez probablement le désactiver pour vous connecter à l'adresse 192.168.0.199 de Geeros en Wifi
  - un smartphone ou une tablette connecté à un réseau 3G en parallèle
- Geeros fonctionne comme un point d'accès « maître ». Il ne peut pas, dans sa configuration de base, se connecter comme client du réseau Wifi de votre organisation
- L'adresse de Geeros est fixée à 192.168.0.199. Pour que votre appareil puisse accéder au robot, il doit accepter l'adressage automatique (adresse fournie par serveur DHCP). Si votre appareil n'accepte pas l'adressage automatique, cela signifie que vous lui avez donné une adresse IP fixe. Dans ce cas, celle-ci doit commencer par 192.168.0. (elle doit donc être de la forme 192.168.0.xxx). Si ce n'est pas le cas, vous ne pourrez pas communiquer avec Geeros.

Enfin, le système Raspberry Pi est totalement accessible via une connexion SSH (vous pouvez utiliser sur Windows PuTTY pour un accès en ligne de commande ou WinSCP pour un accès via un explorateur graphique). Les informations de connexion sont les suivantes :

- login : root
- mot de passe : raspberrypi

Pour plus de détails concernant l'accès et la modification des fichiers sur le robot, veuillez nous contacter à l'adresse [support@3sigma.fr](mailto:support@3sigma.fr).

## 5 Installation des logiciels

Les expériences réalisables avec ce robot utilisent la carte A-Star (compatible Arduino) seule ou en association avec la carte Raspberry Pi. Elles sont par ailleurs chacune basées sur une interface graphique sous la forme d'un tableau de bord exécuté par le logiciel MyViz.

### 5.1 Installation de l'environnement de développement Arduino

Cette section présente les étapes à mettre en œuvre (une fois pour toute) pour mettre en place sur votre ordinateur l'environnement de programmation de la carte Pololu A-Star.

**Attention** : si les possibilités de programmation de la carte A-Star ne vous intéressent pas, c'est-à-dire si seules les expériences programmées en Python sur la carte Raspberry Pi vous intéressent, vous pouvez sauter ce qui suit et aller directement au paragraphe 5.2. En effet, la carte A-Star est déjà programmée lorsque vous recevez le robot.

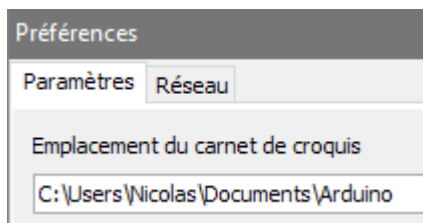
#### 5.1.1 Installation principale

L'IDE Arduino en version supérieure ou égale à 1.8 doit tout d'abord être téléchargé (<http://arduino.cc/en/Main/Software>) et installé (<http://arduino.cc/en/Guide/HomePage>).

### 5.1.2 Installation du support de la carte A-Star

Il faut effectuer les opérations suivantes pour ajouter le support de la carte A-Star dans l'IDE Arduino :

- Télécharger et décompresser l'archive suivante dans le répertoire de votre choix :  
[https://www.pololu.com/file/download/a-star-2.0.0.zip?file\\_id=0J743](https://www.pololu.com/file/download/a-star-2.0.0.zip?file_id=0J743)
- Ouvrir le sous-répertoire « drivers ». Faire un clic bouton droit sur le fichier a-star.inf et sélectionner « Installer »
- Repérer l'emplacement de votre « Répertoire de croquis » (typiquement, Documents\Arduino\libraries sur Windows). Pour identifier ce répertoire sur votre ordinateur, lancez l'IDE Arduino et sélectionnez **Fichier → Préférences**, l'information recherchée se trouve en haut de la fenêtre qui s'ouvre alors (« Emplacement du carnet de croquis »). Par exemple :



- Copier dans ce répertoire de croquis le répertoire nommé « pololu », que vous trouverez dans l'archive précédemment téléchargée
- Redémarrer l'IDE Arduino
- Dans le menu Outils → Type de carte, choisir « Pololu A-Star 32u4 »
- Dans le menu Croquis → Inclure ou bibliothèque, sélectionner « Gérer les bibliothèques »
- Chercher AStar32U4
- Sélectionner l'entrée AStar32U4 trouvée et cliquer sur « Installer »
- Redémarrer l'IDE Arduino

### 5.1.3 Installation de la bibliothèque complémentaire FlexiTimer2

Cette bibliothèque permet d'exécuter à cadence fixe une partie du programme Arduino.

Vous pouvez la télécharger à l'adresse suivante: <http://www.3sigma.fr/telechargements/FlexiTimer2.zip>.

Une fois téléchargée, décompressez-là dans le répertoire des librairies de votre installation Arduino (voir les explications détaillées ci-dessus)

#### 5.1.4 Installation de la bibliothèque complémentaire EnableInterrupt

Cette bibliothèque permet de gérer les interruptions.

Vous pouvez la télécharger à l'adresse suivante: <http://www.3sigma.fr/telechargements/EnableInterrupt.zip>.

Une fois téléchargée, décompressez-là dans le répertoire des librairies de votre installation Arduino (voir les explications détaillées ci-dessus)

#### 5.1.5 Installation de la bibliothèque complémentaire MPU9250

Cette bibliothèque intègre de nombreuses fonctions permettant d'accéder aux informations mesurées par le MPU9250, qui est le capteur accélérométrique, gyroscopique et magnétométrique 3 axes intégré au robot.

Vous pouvez la télécharger à l'adresse suivante:

[http://www.3sigma.fr/telechargements/Sparkfun\\_MPU-9250.zip](http://www.3sigma.fr/telechargements/Sparkfun_MPU-9250.zip).

Une fois téléchargée, décompressez-là dans le répertoire des librairies de votre installation Arduino (voir les explications détaillées ci-dessus)

#### 5.1.6 Installation de la bibliothèque complémentaire SoftwareServo

Geeros est muni d'une caméra orientable dans le plan vertical grâce à un servomoteur. Il existe bien une bibliothèque « Servo » en standard dans l'environnement Arduino, mais celle-ci n'est pas compatible avec le fonctionnement sur « interruption timer » d'une partie du programme du robot, conduisant à des perturbations importantes sur le positionnement du servomoteur.

La bibliothèque « SoftwareServo » permet de remédier à ce problème en pilotant le servomoteur de façon synchronisée avec l'« interruption timer » réalisant les asservissements du robot. Vous pouvez la télécharger à l'adresse suivante: <http://www.3sigma.fr/telechargements/SoftwareServo.zip>.

Une fois téléchargée, décompressez-là dans le répertoire des librairies de votre installation Arduino (voir les explications détaillées ci-dessus)

**ATTENTION !**

**L'environnement Arduino doit être redémarré après l'installation d'une bibliothèque complémentaire.**

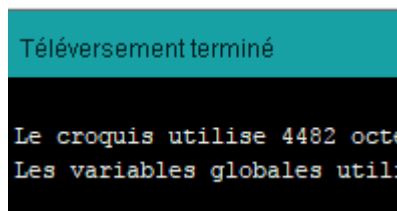
### 5.1.7 Choix de la carte Arduino dans l'IDE

L'IDE Arduino permet de programmer un très grand nombre de cartes. Il faut par conséquent le configurer pour fonctionner avec la carte compatible A-Star. Dans le menu **Outils → Type de carte**, choisir Pololu A-Star 32U4.

Lorsque la carte A-Star est reliée à l'ordinateur via le câble micro USB fourni, il est également possible de choisir le port de cet ordinateur dans l'IDE Arduino en sélectionnant, dans la liste **Outils → Ports**, celui correspondant à l'A-Star.

### 5.1.8 Programmation de la carte Arduino

Enfin, la programmation de la carte se fait en ouvrant (depuis l'IDE Arduino) le programme que vous souhaitez utiliser et en sélectionnant **Croquis → Téléverser**. La programmation est faite quand il s'affiche « Téléversement terminé » dans la zone bleue en bas de l'écran :



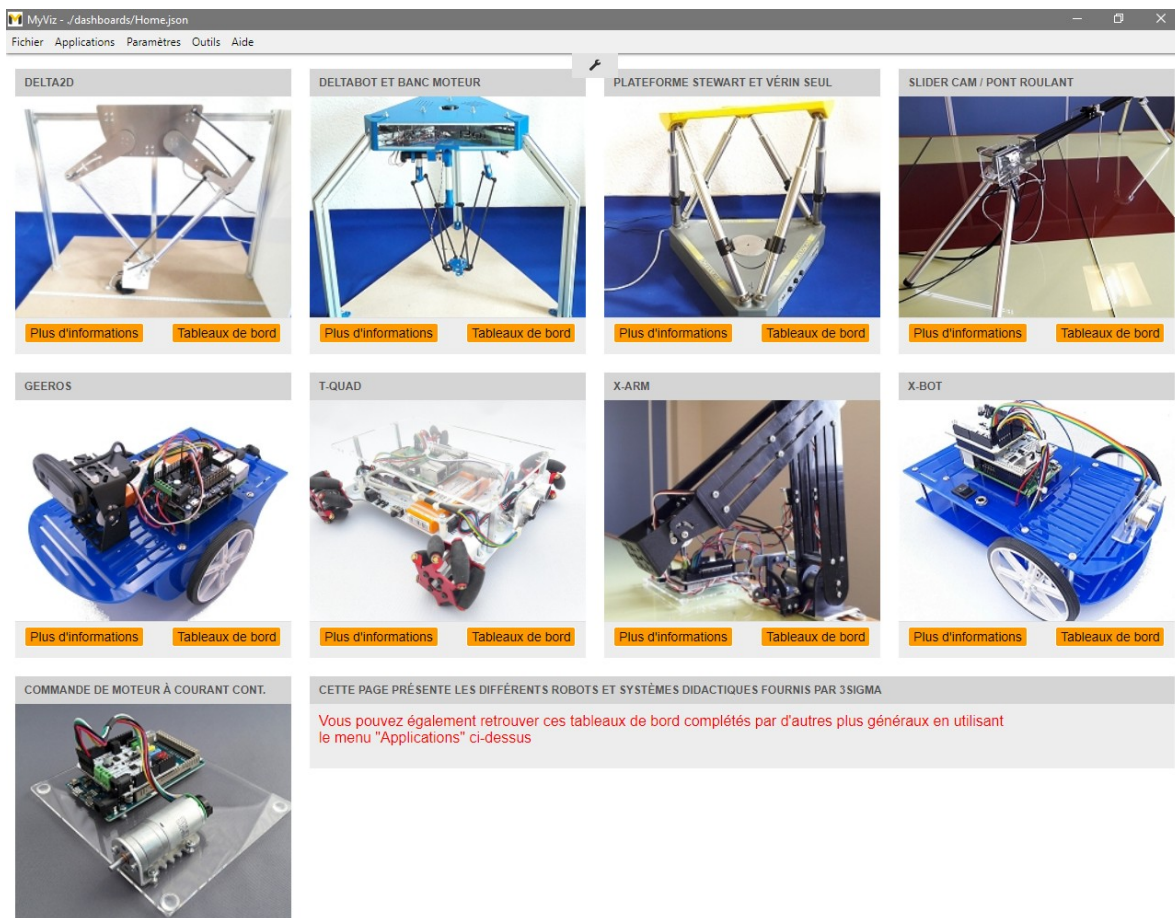
## 5.2 Installation et première utilisation du logiciel MyViz

MyViz se télécharge depuis la page suivante :

<https://www.3sigma.fr/myviz/MyViz.zip>

Il n'y a pas à proprement parler d'installateur. Il suffit de décompresser l'archive dans le répertoire de votre choix (mais si possible dans C: et jamais dans une arborescence profonde à cause de la limitation sur la longueur des chemins de fichiers dans Windows) et de lancer le programme **MyViz.exe**.

Le tableau de bord initialement affiché sera similaire à la capture d'écran ci-dessous :



Notez que si une ancienne version de MyViz (antérieure à la 0.8.5) a été utilisée précédemment, l'écran de démarrage ou le passage d'une application à l'autre pourraient ne pas fonctionner correctement. Dans ce cas, il faut supprimer le répertoire C:\Utilisateurs\nom\_d\_utilisateur\AppData\local\MyViz, ou équivalent.

Cette opération doit être faite une seule et unique fois pour un compte donné.

## 6 Mise en œuvre de l'ensemble

### 6.1 Précautions d'emploi

Nous insistons sur le fait que ce robot est un matériel de développement qui nécessite un certain nombre de précautions d'emploi.

#### 6.1.1 Connexions d'alimentation sur la carte A-Star

Il est impératif de faire très attention aux connexions de l'alimentation de la carte A-Star car celle-ci n'est pas protégée contre les inversions de polarité. Une erreur de connexion sur les bornes d'alimentation risque d'entraîner la destruction de certains éléments. Le robot étant livré connecté et fonctionnel, il est préférable de ne pas modifier les branchements sur les connecteurs d'alimentation.

#### 6.1.2 Recharge de la batterie

Le chargeur fourni est destiné à recharger la batterie du robot en le connectant sur le connecteur jack situé à côté du bouton de marche-arrêt.

Il faut compter environ 2 heures pour une demi-recharge de la batterie et 8 heures pour une recharge complète. Lorsque la batterie est en charge, la LED rouge du chargeur est allumée. Lorsque la batterie est chargée, la LED verte du chargeur s'allume.

Les robots Geeros possèdent par ailleurs un afficheur LCD permettant de visualiser la tension en sortie de la batterie. Lorsqu'une batterie est totalement chargée, sa tension de sortie est d'environ 8.4 V. Lorsque la tension descend au dessous de 7V, il faut envisager une recharge.

### 6.1.3 Précautions d'utilisation

Le robot doit être utilisé dans les conditions suivantes:

- Au sol. Ne jamais l'utiliser en hauteur (sur une table, par exemple) à cause des risques de chute
- A l'intérieur. Les sols extérieurs présentent souvent des aspérités et des obstacles sur-dimensionnés par rapport à la taille du robot. Par ailleurs, celui-ci craint l'eau et l'humidité

Le robot peut être reprogrammé à votre guise. Vous pouvez par exemple faire des programmes permettant d'étudier la commande des moteurs électriques. Mais attention: il est fortement déconseillé de faire des expériences de fonctionnement « rotor bloqué » avec une tension d'alimentation du moteur trop élevée. Ce type d'expérience peut générer des courants trop forts qui réduisent la durée des vie des éléments.



## 6.2 Première utilisation

Le programme pré-chargé dans le robot à la livraison lui permet de tenir en équilibre sur ses deux roues et de recevoir des consignes de mouvement (transmises par liaison Wifi) provenant d'un ordinateur, d'un smartphone ou d'une tablette.

La procédure à suivre est simple:

- Poser le robot au sol sans l'allumer, penché vers l'arrière
- Mettre le robot sous tension en basculant l'interrupteur sur « I »
- Attendre environ une minute que la led verte s'allume sur la Webcam
- Redresser doucement le robot : il se maintient alors en équilibre même en cas de perturbations
- Pour le remettre en position d'attente sur l'arrière, soulevez le robot et penchez-le vers l'arrière pour désactiver l'asservissement de verticalité et arrêter les moteurs


### 6.2.1 Pilotage par ordinateur

Pour piloter le robot, se connecter en Wifi sur le réseau de SSID « Geeros ». Le mot de passe Wifi est « geerosraspberry », puis :

- Lancer le logiciel MyViz
- Si le tableau de bord initialement affiché dans MyViz est celui présenté au paragraphe 5.2, cliquer sur le bouton « Tableaux de bord » dans la zone « Geeros ».  
Sinon, si le tableau de bord affiché ne concerne pas le banc moteur, aller dans le menu Applications → Robots didactiques → Geeros → Tableaux de bord

Le tableau de bord permettant d'accéder aux différentes applications du banc moteur s'ouvre alors :

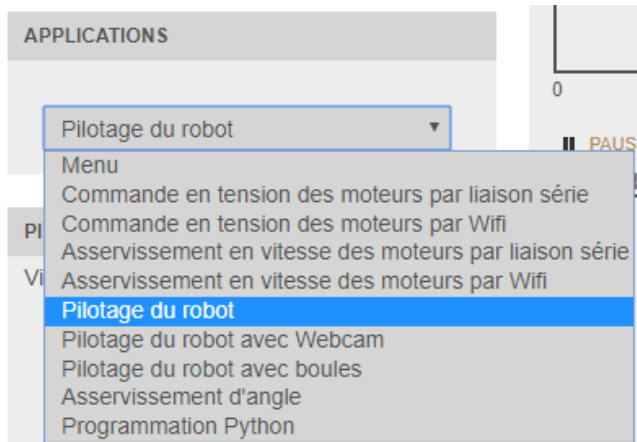
**GEEROS**



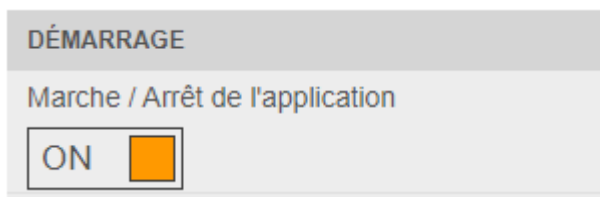
APPLICATIONS	
Commande en tension des moteurs par liaison série	<a href="#">Ouvrir</a>
Commande en tension des moteurs par Wifi	<a href="#">Ouvrir</a>
Asservissement en vitesse des moteurs par liaison série	<a href="#">Ouvrir</a>
Asservissement en vitesse des moteurs par Wifi	<a href="#">Ouvrir</a>
Pilotage du robot	<a href="#">Ouvrir</a>
Pilotage du robot avec Webcam	<a href="#">Ouvrir</a>
Pilotage du robot avec boules	<a href="#">Ouvrir</a>
Asservissement d'angle	<a href="#">Ouvrir</a>
Programmation Python	<a href="#">Ouvrir</a>

- Choisir une des applications présentées à l'exception (dans un premier temps) des deux nécessitant une liaison série avec le robot et de celle de pilotage du robot avec boules.  
Utiliser par exemple l'application de « pilotage du robot » après avoir posé le robot au sol, incliné vers l'arrière

- Noter que par la suite, il est possible de passer directement d'une application à une autre en utilisant la zone « Applications » intégrée dans chaque tableau de bord :



- Démarrer l'interaction en cliquant sur le bouton "ON / OFF" en haut à gauche du tableau de bord pour le passer sur ON:



- Lorsque des courbes commencent à défiler, redresser le robot jusqu'à ce qu'il s'équilibre tout seul
- Vous pouvez alors interagir avec le système grâce au joystick de la zone « Pilotage » du tableau de bord:



Ce joypad permet de donner:

- une consigne de vitesse de rotation uniquement, lorsque le centre de la boule jaune se trouve dans la moitié supérieure du disque blanc
- une consigne de vitesse de rotation et une consigne de vitesse de translation lorsque le centre de la boule jaune se trouve dans la moitié supérieure de l'anneau gris
- une consigne de vitesse de translation (négative) uniquement, lorsque le centre de la boule jaune se trouve dans la moitié inférieure de l'anneau gris
- aucune consigne lorsque le centre de la boule jaune se trouve dans la moitié inférieure du disque blanc

Noter par ailleurs que ce contrôle est doté d'un « ressort de rappel virtuel » : quand vous « lâchez » la boule avec la souris, celle-ci revient automatiquement au centre et ne donne plus de consigne.

Enfin, si vous souhaitez arrêter le programme de contrôle de verticalité et de mouvement, positionnez le bouton sur OFF en haut à gauche du tableau de bord.

## 6.2.2 Pilotage par smartphone ou tablette

Pour pouvoir piloter Geeros en Wifi grâce à votre smartphone ou votre tablette (nous utiliserons par la suite le nom générique « appareil »), les conditions suivantes doivent être respectées:

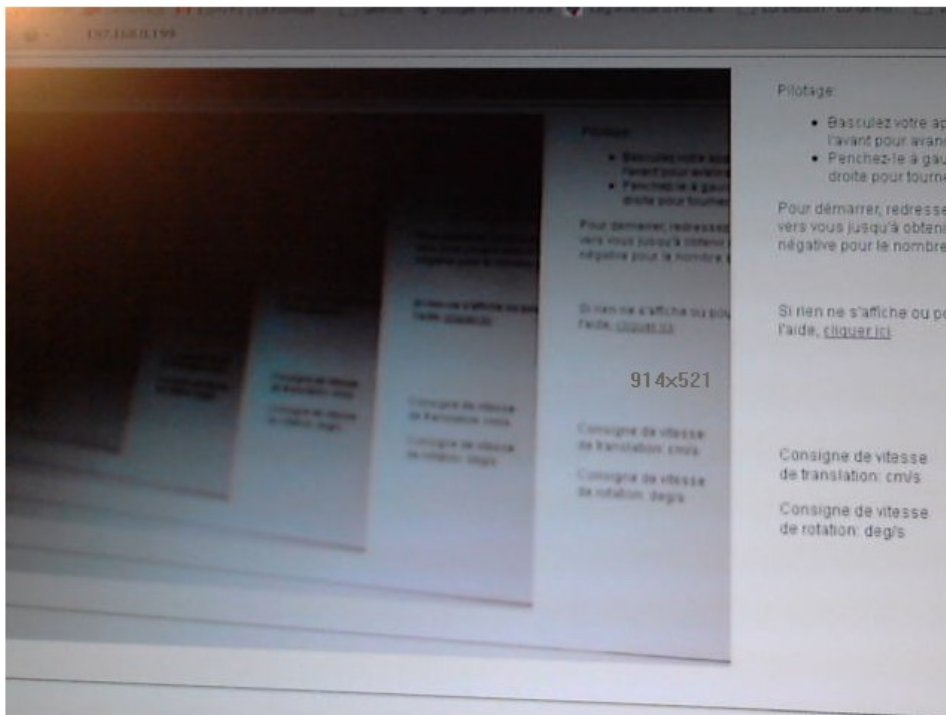
- Votre appareil doit posséder une interface Wifi
- Il doit posséder un accéléromètre
- Il doit être orienté en mode « paysage », comme ceci:



- Vous devez avoir rejoint le réseau Wifi « Geeros » (voir plus haut)

Normalement, tout appareil récent respectant les conditions ci-dessus devrait fonctionner.

Ouvrez ensuite le navigateur Web de votre appareil à l'adresse 192.168.0.199. Vous obtenez tout d'abord la page suivante:



Pilotage:

- Basculez votre appareil vers l'avant pour avancer
- Penchez-le à gauche ou à droite pour tourner

Pour démarrer, redressez l'appareil vers vous jusqu'à obtenir une valeur négative pour le nombre suivant:

Si rien ne s'affiche ou pour avoir de l'aide, [cliquer ici](#).

Consigne de vitesse de translation: cm/s

Consigne de vitesse de rotation: deg/s

Cette page permet de tester si votre appareil est compatible. Si tout se passe bien, vous devez voir, à droite de la caméra, un nombre changer en fonction de l'inclinaison de votre appareil. Ce nombre correspond à l'accélération de la pesanteur mesurée par votre appareil, normalisée entre 0 et 1. Si vous ne voyez pas ce nombre, cela signifie:

- que votre appareil ne possède pas d'accéléromètre
- ou bien qu'il n'est pas compatible avec cette application Web de pilotage (iPhone, iPad ou iPod avec iOS < 4.2, par exemple)
- ou bien que le navigateur utilisé n'est pas compatible

Dans ce cas, merci de nous contacter à l'adresse [support@3sigma.fr](mailto:support@3sigma.fr) en nous indiquant les caractéristiques précises du smartphone ou de la tablette que vous utilisez. Une solution personnalisée est peut-être possible.

Si vous voyez l'accélération évoluer en fonction de l'inclinaison de votre appareil, tout va bien. Avant d'accéder au pilotage à proprement parler, vous devez activer l'application. C'est très simple, il suffit de redresser votre appareil jusqu'à obtenir une accélération nulle. Une fois ceci fait, l'interface change légèrement : l'accélération n'est plus visible, seules restent les consignes de vitesse de translation et de rotation (en plus de la transmission vidéo temps-réel de la Webcam, à gauche de l'écran). Piloter Geeros est alors très facile:

- Lorsque vous basculez votre appareil vers l'avant, vous modifiez la consigne de vitesse de translation (entre 0 et 50 cm/s)
- Lorsque vous inclinez votre appareil sur la gauche ou sur la droite, vous modifiez la consigne de vitesse de rotation (entre -180 et 180 deg/s), comme avec un volant. Si l'inclinaison est trop forte à gauche ou à droite, l'application détecte que votre appareil est en mode portrait et annule les consignes

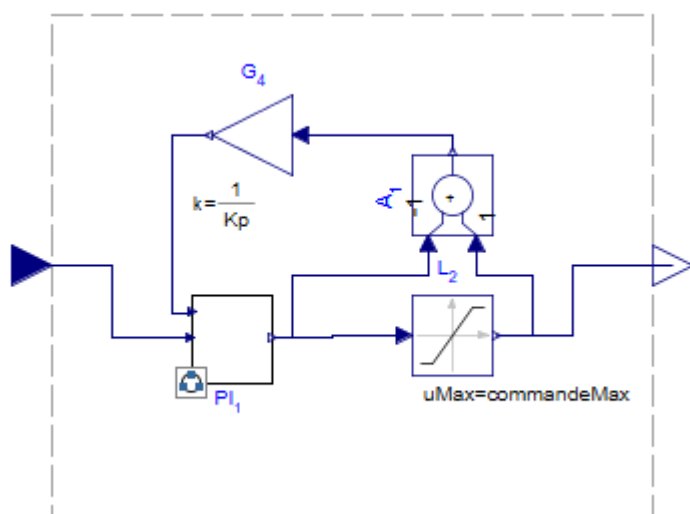
**Remarque :** Dans ce mode de pilotage par smartphone ou tablette, il est impossible de faire reculer le robot car il est impossible de lui donner une consigne de vitesse négative.

## 7 Détails techniques

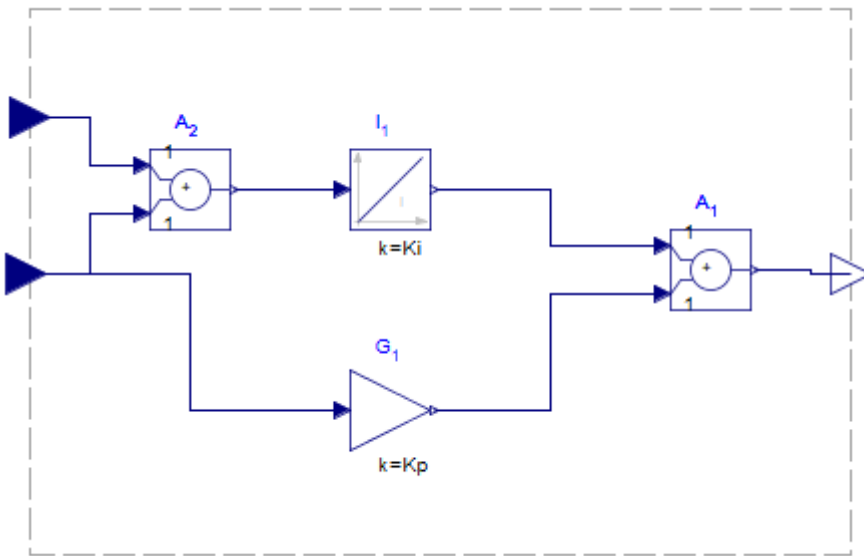
### 7.1 Asservissements de type PI

Une partie des asservissements est réalisée par des régulateurs de type proportionnel-intégral avec anti-saturation de l'intégrateur (dispositif anti-windup).

Le régulateur a la forme générale suivante :



On voit le dispositif d'anti-windup et le correcteur proportionnel-intégral dans le sous-système PI, implémenté comme ceci :



La fonction de transfert du régulateur PI si la sortie n'est pas saturée est la suivante :

$$Kp + \frac{Ki}{s} \quad (7.1)$$

Si la sortie est saturée, la relation entre l'entrée ( $u$ ), la sortie non saturée ( $y$ ) et la sortie saturée ( $ysat$ ) est la suivante :

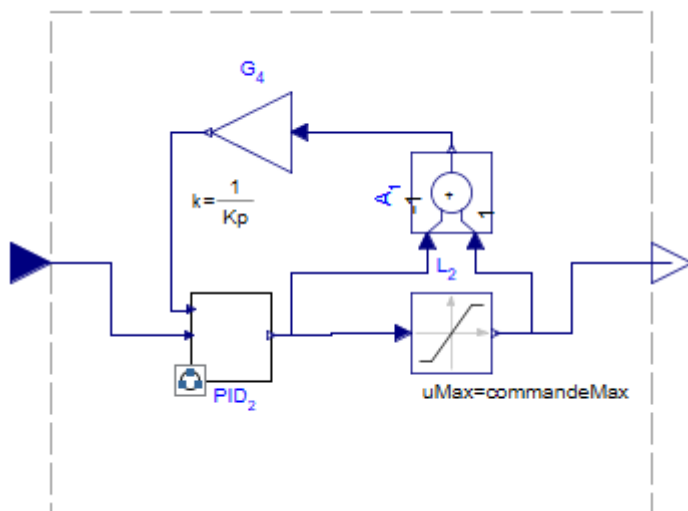
$$ysat = \left( Kp + \frac{Ki}{s} \right) u + \frac{Ki (ysat - y)}{sKp} \quad (7.2)$$

## 7.2 Asservissements de type PID

Une autre partie des asservissements est réalisée par des régulateurs de type proportionnel-intégral-dérivé avec anti-saturation de l'intégrateur (dispositif anti-windup).

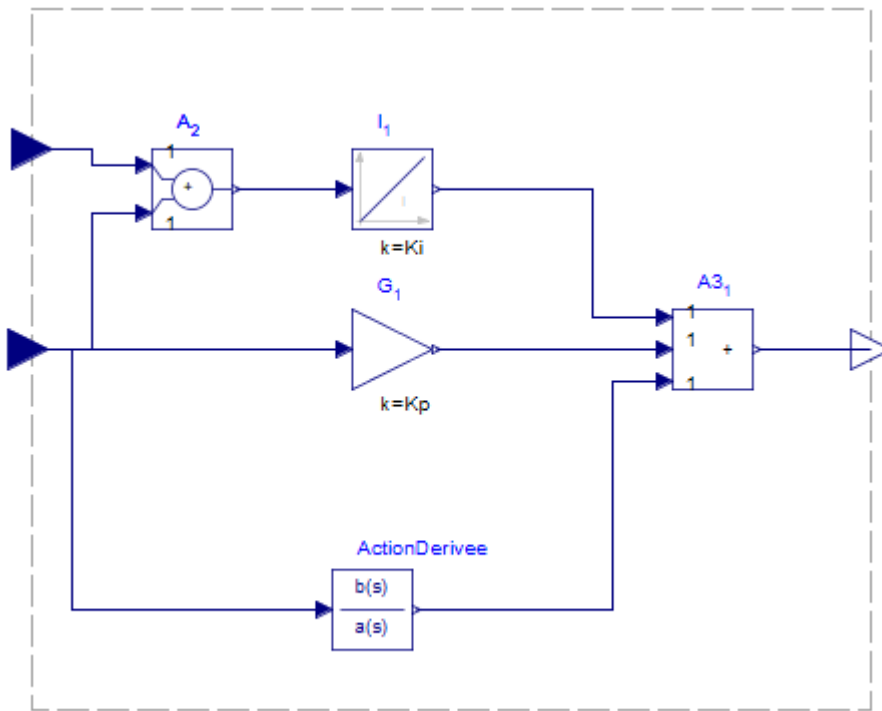
Le régulateur a la forme générale suivante :





On voit le dispositif d'anti-windup et le correcteur proportionnel-intégral-dérivé dans le sous-système PID.

Celui-ci est implémenté comme ceci :



Par rapport au PI, présenté précédemment, on a ajouté la fonction de transfert « ActionDerivee » suivante :

$$\frac{Kd s}{Tf s + 1} \quad (7.3)$$

D'un point de vue mathématique, le dénominateur permet d'éviter que la fonction de transfert du PID soit « non-propre » (ordre du numérateur supérieur à l'ordre du dénominateur). D'un point de vue plus concret, il permet aussi de filtrer la dérivée. La valeur de  $Tf$  est 0.02.

Si la sortie est saturée, la relation entre l'entrée ( $u$ ), la sortie non saturée ( $y$ ) et la sortie saturée ( $ysat$ ) est la suivante :

$$ysat = \left( Kp + \frac{Ki}{s} + \frac{Kd s}{Tf s + 1} \right) u + \frac{Ki (ysat - y)}{s Kp} \quad (7.4)$$

## 7.3 Fonctionnement d'un gyropode

Ce chapitre décrit le principe de fonctionnement d'un gyropode, qu'il est utile de connaître pour comprendre son comportement.

### 7.3.1 Maintien en équilibre

Deux capteurs sont essentiels pour le maintien en équilibre: un accéléromètre et un gyroscope. Geeros est équipé d'un capteur qui communique avec la carte Raspberry Pi et la carte A-Star via une interface I2C. Ce capteur combine un accéléromètre 3 axes, un gyroscope 3 axes et un magnétomètre 3 axes. Seul un axe accélérométrique et un axe gyroscopique sont utilisés dans l'asservissement de verticalité.

Un accéléromètre 3 axes permet de mesurer l'accélération du gyropode plus l'accélération de la pesanteur dans 3 axes orthogonaux liés au robot. L'algorithme de maintien en équilibre utilise la mesure de l'accélération longitudinale: en effet, quand le gyropode est incliné sans mouvement, l'accéléromètre mesure la pesanteur multipliée par l'angle d'inclinaison:

$$acc = g \sin(\theta) \quad (7.5)$$

La valeur de la pesanteur ( $g$ ) étant connue ( $9.81 \text{ m/s}^2$ ) et l'angle d'inclinaison étant en général faible ( $\sin(\theta) \sim \theta$ ), cela nous permet d'obtenir:

$$\theta = \frac{acc}{g} \quad (7.6)$$

Mais la connaissance de l'angle d'inclinaison n'est pas suffisante. En effet, deux cas de figure peuvent se présenter:

- le gyropode est incliné et est en train de tomber (l'inclinaison augmente)
- le gyropode est incliné mais est en train de se redresser (l'inclinaison diminue)

Dans le premier cas, il faudra agir plus fortement que dans le second cas. La distinction entre les deux cas se fait grâce à la mesure de la vitesse de chute (ou de redressement) fournie par un des axes (celui qui est parallèle à l'axe des roues) de mesure du gyroscope.

Avec ces deux mesures (1 axe accélérométrique et 1 axe gyroscopique), nous possédons en théorie assez d'informations pour maintenir correctement le gyropode à la verticale. Cependant, nous avons fait précédemment l'hypothèse d'un gyropode incliné sans mouvement pour la mesure de l'accélération. Or, dans le cas général le gyropode est en mouvement et peut avoir une accélération longitudinale qui va se superposer à l'accélération de la pesanteur:

$$acc\_mesurée = g \sin(\theta) + acc\_longitudinale \quad (7.7)$$

Cette accélération longitudinale étant inconnue, on ne peut pas en théorie remonter à l'angle d'inclinaison. Pour cette raison, l'asservissement de verticalité de Geeros intègre un filtre permettant d'estimer l'angle d'inclinaison à partir des deux mesures de l'accéléromètre et du gyroscope.

Remarque : Geeros est capable de se maintenir en équilibre même s'il n'est pas parfaitement équilibré en « statique » (lorsqu'il est éteint). Dans ce cas, il penchera vers l'avant (même à l'arrêt) s'il y a trop de poids sur l'arrière et vice-versa.

### 7.3.2 Mouvement

Avant tout, il est important de bien comprendre comment fonctionne un gyropode: lorsqu'il est en mouvement, il est penché en avant dans le sens de la marche pour maintenir son équilibre. En effet, la force de motorisation ne s'applique pas sur le centre de gravité du robot mais au point de contact avec le sol. La force de motorisation engendre donc un couple de rotation du gyropode autour de son centre d'inertie et a tendance à le faire tomber. Pour contrer ce couple de chute, le gyropode asservi se penche en avant pour utiliser la pesanteur.

Le gyropode peut donc rester à l'équilibre (mais pas à la verticale) en mouvement lorsque le couple de rotation généré par les moteurs est compensé par le couple de rotation généré par la pesanteur. Il se déplace alors penché dans le sens de la marche.

### 7.3.3 Démarrage

Comme indiqué précédemment, Geeros est penché vers l'avant lorsqu'il se déplace. Or, il est vertical à l'arrêt. La phase de démarrage doit permettre de passer d'un état à l'autre (tout se fait automatiquement). Cette phase permettant de passer de l'état « vertical à l'arrêt » à l'état « penché vers l'avant en mouvement » est la suivante:

Les moteurs vont tout d'abord tourner dans le sens inverse de la marche. Cela a pour effet de faire pencher le gyropode vers l'avant

Les moteurs tournent ensuite dans le sens de la marche, leur vitesse de rotation et l'angle d'inclinaison se compensant mutuellement pour éviter la chute. Nous nous trouvons donc en présence du paradoxe suivant: pour avancer, un gyropode doit d'abord (légèrement) reculer.

#### **IMPORTANT !**

Si vous démarrez Geeros collé à un mur à l'arrière, il ne pourra par conséquent pas démarrer puisqu'il ne pourra pas initialement reculer !

### **7.3.4 Arrêt**

Nous avons vu précédemment que lorsque Geeros est en mouvement, il est penché en avant dans le sens de la marche pour maintenir son équilibre. Quand il doit s'arrêter et retrouver la verticale, il doit transitoirement se pencher vers l'arrière, dans le sens contraire au mouvement, pour freiner (sinon, il chute vers l'avant). La seule façon de réaliser ça est de faire en sorte que la base du gyropode aille transitoirement plus vite que son centre de gravité. Par conséquent, pour s'arrêter, le gyropode doit tout d'abord accélérer !

S'il roule déjà à la vitesse maximale permise par les moteurs, il ne pourra plus accélérer et sera donc incapable de s'arrêter. La vitesse du robot est limitée à 50 cm/s dans les différentes applications de pilotage, ce qui permet de conserver aux moteurs une réserve d'accélération suffisante pour que ceux-ci puissent réaliser un arrêt du robot.

Une dernière conséquence est que l'arrêt immédiat d'un robot gyropode n'est pas possible. Si on coupe brutalement l'alimentation des moteurs, le robot va chuter, entraîné par sa vitesse.

### **7.3.5 Équations**

L'archive des fichiers du robot (voir chapitre 1) contient un répertoire « ModelesEtEquations » contenant la mise en équations du robot ainsi qu'un modèle de simulation réalisé sous le logiciel MapleSim.

### 7.3.6 Asservissements

Le suivi de vitesse longitudinale avec maintien de verticalité est obtenu grâce à un asservissement par retour d'état. Ces états sont les suivants :

- $int\_errV$  : intégrale de l'écart entre la consigne de vitesse et la vitesse longitudinale
- $V$  : vitesse longitudinale
- $\Theta$  : angle du robot par rapport à la verticale
- $\omega$  : vitesse de rotation par rapport à la verticale

Dans le programme Python gérant cet asservissement (Geeros.py), la commande générée par le retour d'état est écrite sous une forme se rapprochant le plus possible de celle de deux régulateurs PI :

$$\frac{K_{iv} \cdot (V_{cons}(t) - V(t))}{s} + \left( K_{pv1} + \frac{K_{pv2}}{R} \right) \cdot (0 \cdot V_{cons}(t) - V(t)) + K_{p\omega} \cdot (\omega_{cons}(t) - \omega(t)) + K_{i\omega} \cdot (\theta_{cons}(t) - \theta(t))$$

Avec:

$$\frac{(V_{cons}(t) - V(t))}{s} = int\_errV(t)$$

$$\omega_{cons}(t) = 0$$

$$\theta_{cons}(t) = 0$$

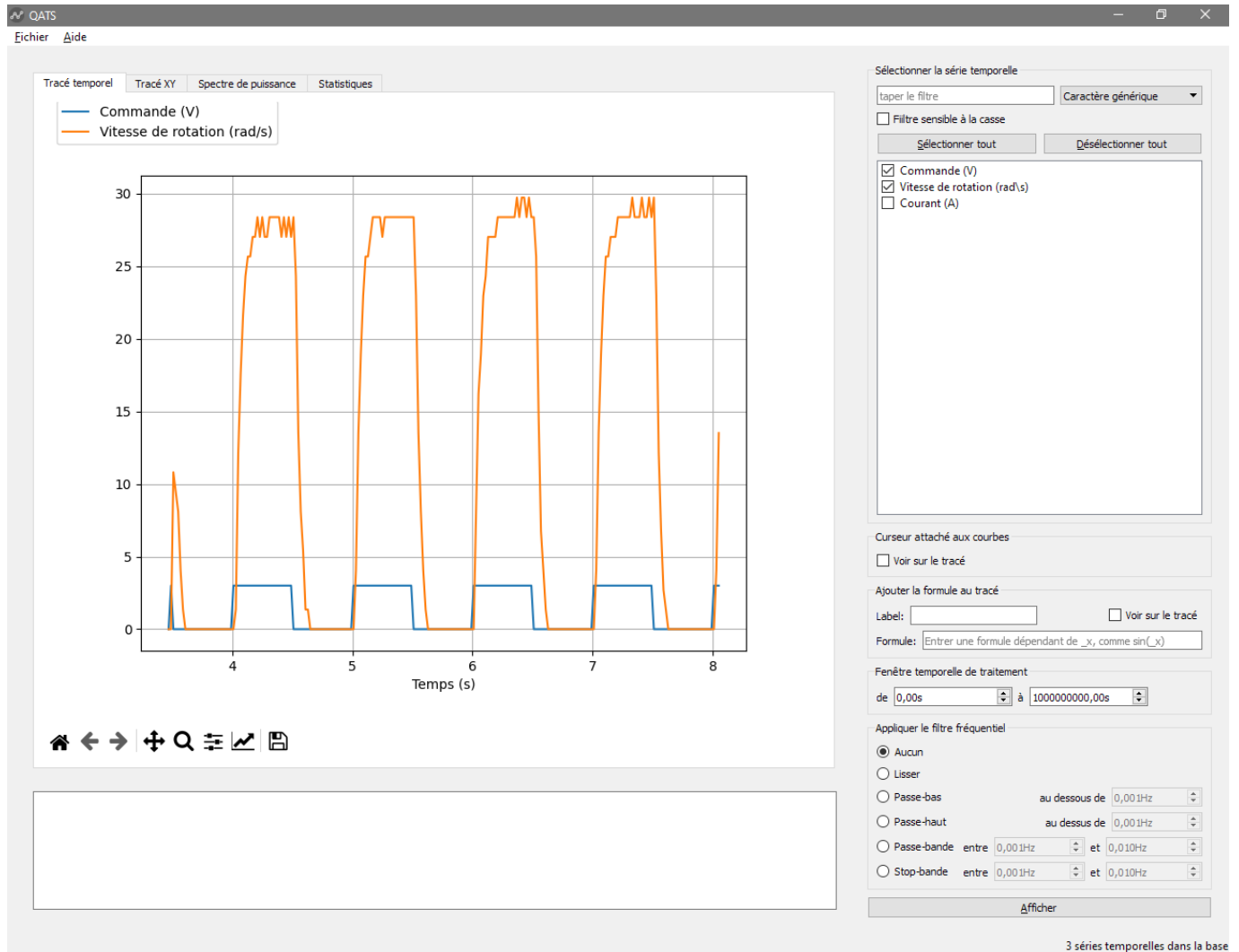
Le robot étant piloté en vitesse, la seule consigne est celle spécifiée pour la vitesse longitudinale. Les consignes d'angle et de vitesse angulaire par rapport à la verticale sont bien sûr nulle pour éviter la chute du robot.

Si vous souhaitez de plus amples détails sur la conception de cet asservissement par retour d'état, vous pouvez nous contacter à l'adresse [support@3sigma.fr](mailto:support@3sigma.fr).

Enfin, l'asservissement en vitesse de rotation autour de la verticale est un simple régulateur PI.

## 7.4 Tracés graphiques

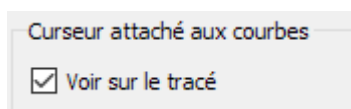
Chaque tableau de bord dispose d'un bouton « Courbes » permettant d'ouvrir un outil de tracé spécifique, dont voici une capture d'écran :



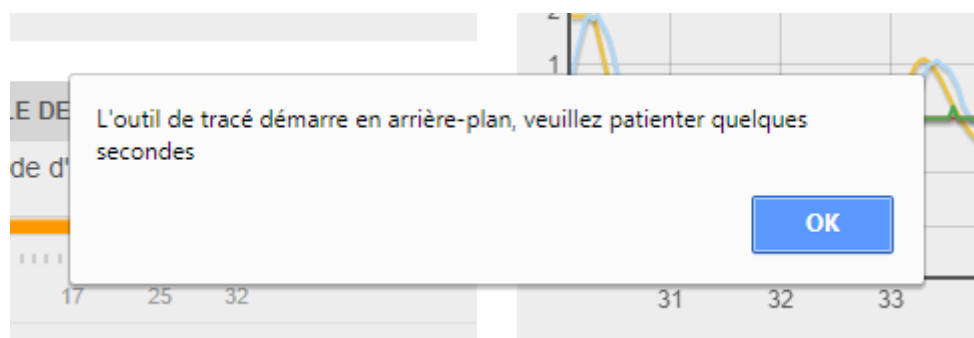
Cet outil possède les fonctionnalités suivantes :

- Possibilité de charger de multiples fichiers
- Sélection / désélection facile des signaux à tracer
- Tracé temporel, fréquentiel et affichage des données statistiques
- Possibilité de lissage et de filtrage (cliquer sur le bouton « Afficher » après la sélection du traitement à apporter sur les courbes)

- Curseur temporel multi-courbes. Cocher la case suivante pour l'afficher :



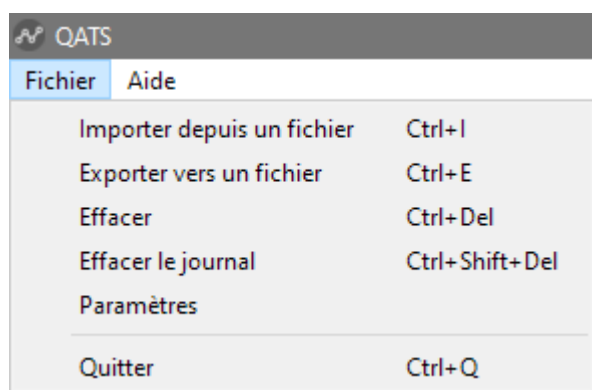
Noter que lors du clic sur le bouton « Courbes » d'un tableau de bord, la fenêtre d'information suivante s'affiche au milieu :



Elle indique que la demande a bien été prise en compte et que l'outil va démarrer, ce qui peut prendre quelques secondes. Une fois lancée, cette application peut ne pas apparaître directement au premier plan ; dans ce cas, son icône clignote dans la barre des tâches :



Il est également possible de lancer cet outil de tracé sans signaux chargés a priori, depuis le menu principal de MyViz (Outils → Tracés). Dans ce cas, le menu Fichier permet d'importer un fichier .csv externe, pas nécessairement issu de MyViz mais devant contenir impérativement le temps en première colonne :



Noter que la zone de tracé peut être agrandie avec la souris en « tirant » les zones grises sous et à droite de cette zone vers le bas et vers la droite.



## 8 Activités réalisables avec le système en mode Raspberry Pi / Python

Dans ce mode, les différentes applications sont programmées en Python et sont exécutées par la carte Raspberry Pi. Les programmes sont pré-chargés sur la carte, celle-ci n'a donc pas besoin d'être « programmée » pour chaque nouvelle application.

La carte A-Star est également sollicitée pour le comptage des impulsions générées par les codeurs incrémentaux. Pour que l'ensemble fonctionne, il faut téléverser un programme spécifique sur cette carte A-Star :

**FirmwareAStarGeeros.ino**

**Ce programme est pré-chargé sur la carte A-Star à la livraison du robot. Si vous commencez l'exploitation de ce dernier par des programmes Python, vous n'avez pas besoin de le téléverser de nouveau.**

Les différentes applications du système en mode Raspberry Pi / Python sont présentées ci-dessous. Nous indiquons pour chacune le programme Python utilisé sur la carte Raspberry Pi : n'hésitez pas à le modifier en fonction de vos besoins.

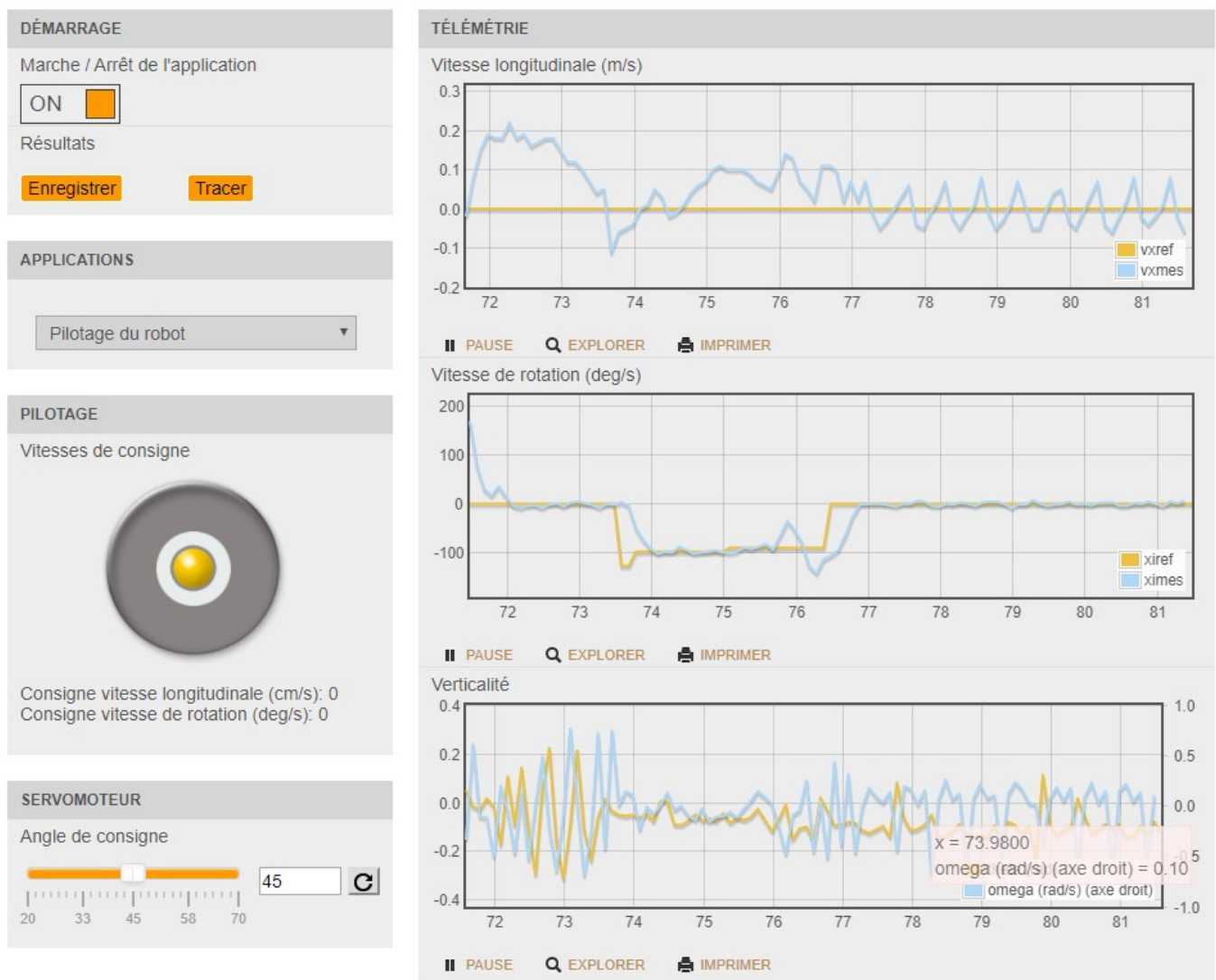
## 8.1 Contrôle de vitesse avec asservissement de verticalité et pilotage en Wifi

### 8.1.1 Présentation

Cette expérience correspond au programme exécuté par le robot à sa mise sous tension dans sa configuration initiale (en sortie du carton de livraison). Elle permet de piloter en Wifi les mouvements du robot et d'avoir une télémétrie des données mesurées.

Le programme Python utilisé (déjà présent sur la carte Raspberry Pi) est :  
`/root/programmes_python/Geeros.py`

Voici une capture d'écran du tableau de bord MyViz :



Cette interface possède les fonctionnalités suivantes :

- Visualisation de la vitesse longitudinale (consigne et mesure) mesurée à partir de la vitesse de rotation des roues (odométrie)
- Visualisation de la vitesse de rotation (consigne et mesure) mesurée à partir de la vitesse de rotation des roues (odométrie)
- Visualisation de l'angle et de la vitesse de rotation du robot par rapport à la verticale
- Pilotage du robot via un joystick virtuel (voir paragraphe 6.2 pour l'utilisation de ce dernier)
- Pilotage du servomoteur permettant d'orienter la Webcam
- Tracé différé des données brutes avec sauvegarde possible dans un fichier texte
- Passage facile d'une application à une autre via une liste déroulante

Attention : les commandes de pilotages ne sont pas actives pendant les 5 premières secondes.

### 8.1.2 Expérimentations possibles

Au-delà de son aspect ludique, cette expérience permet de visualiser les caractéristiques des asservissements de mouvement et de verticalité du robot :

- les courbes montrent la réponse de ce système asservi par rapport à une consigne de vitesse (longitudinale et / ou de rotation) manuelle
- l'angle et la vitesse de rotation par rapport à la verticale doivent idéalement rester le plus proche possible de 0, mais comme ce n'est bien sûr pas le cas lors d'un mouvement, il est intéressant de comprendre l'évolution de ces courbes lors de perturbations appliquées sur le robot (une perturbation peut être une tentative de déstabilisation manuelle, une consigne de vitesse non nulle, un changement d'orientation de la Webcam, un stylo posé à l'avant ou à l'arrière, ...)
- les frottements secs sur les moteurs provoquent un mouvement de balancier permanent du robot si celui-ci est posé sur une surface dure. Sur un tapis, l'effet de frein de ce dernier peut empêcher ce mouvement

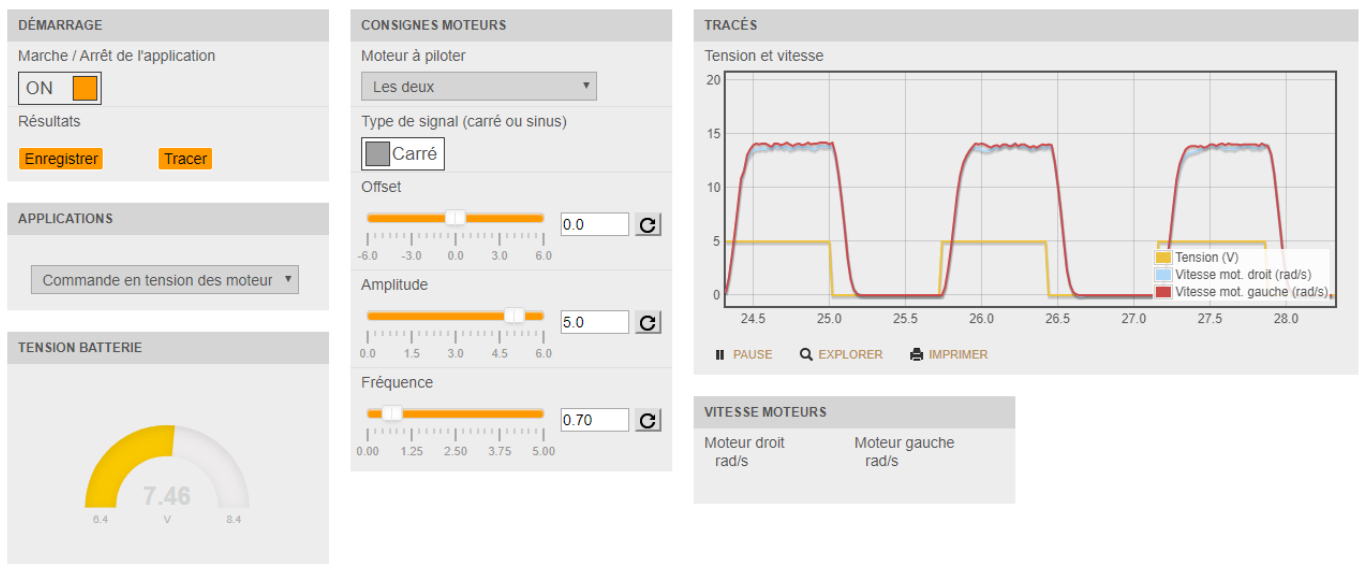
## 8.2 Commande des moteurs en tension

### 8.2.1 Présentation

Cette expérience permet de donner une consigne de tension aux moteurs du robot et d'avoir une télémétrie des données mesurées.

Le programme Python utilisé (déjà présent sur la carte Raspberry Pi) est :  
`/root/programmes_python/CommandeEnTension2.py`

Voici une capture d'écran du tableau de bord MyViz :



Cette interface possède les fonctionnalités suivantes :

- Visualisation de la tension de commande et des vitesses mesurées
- Génération d'une consigne de tension sous la forme d'un signal carré ou sinusoïdal d'amplitude et de fréquence variable, avec ou sans offset (remarque : la somme offset + amplitude est saturée en interne à  $\pm 6$  V)
- Visualisation de la tension de la batterie
- Tracé différé des données brutes avec sauvegarde possible dans un fichier texte
- Passage facile d'une application à une autre via une liste déroulante

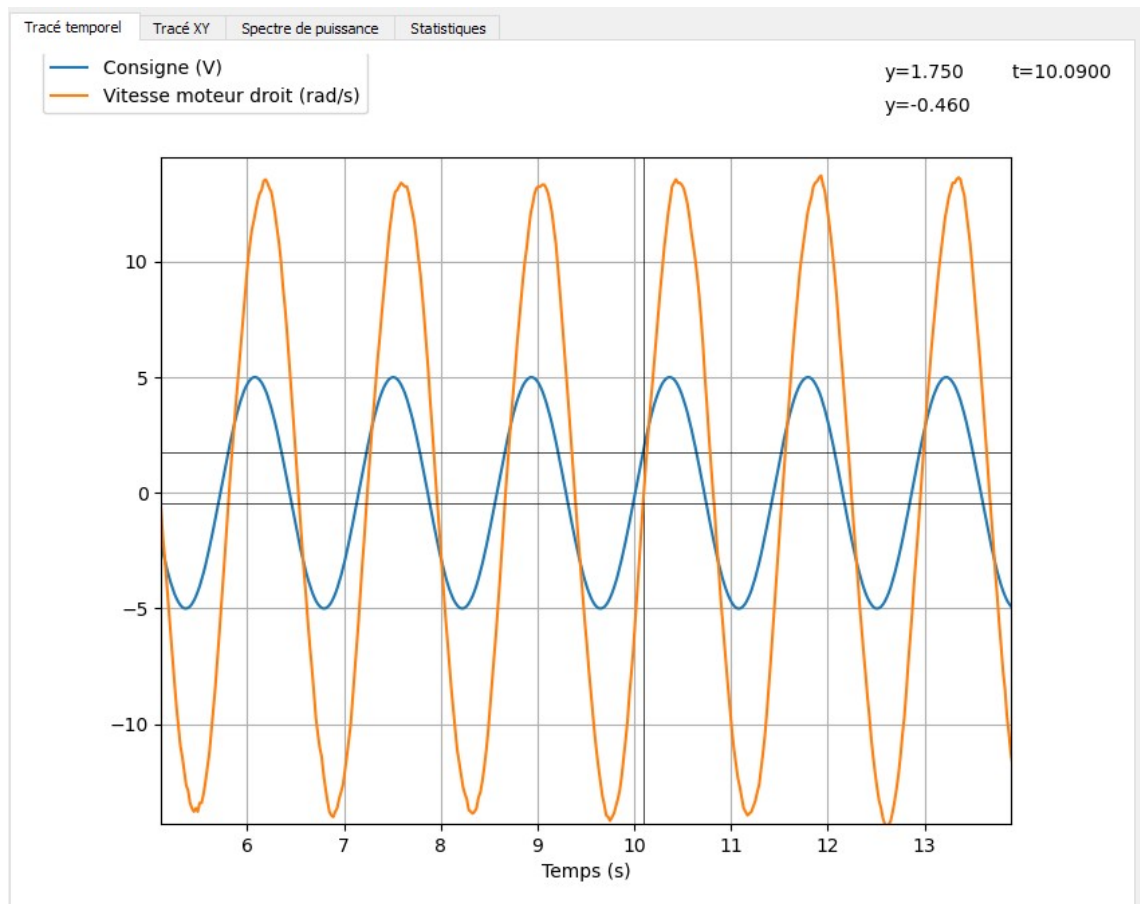
Attention : les commandes de pilotages ne sont pas actives pendant les 5 premières secondes.

## 8.2.2 Expérimentations possibles

Cette expérience permet d'estimer les caractéristiques des moteurs en boucle ouverte :

- Le temps de réponse peut être mesuré grâce à un essai de réponse à un échelon :
  - en changeant avec la souris l'offset de commande
  - ou en générant un signal carré
- Il est possible de construire un diagramme de Bode point par point de la façon suivante :
  - définir une amplitude de consigne de tension sinusoïdale constante pour tous les points de mesure
  - définir la fréquence du point de mesure
  - cliquer sur le bouton « Tracer » dans la zone des résultats du tableau de bord
  - mesurer le gain et le déphasage entre la commande et la mesure
  - (à l'extérieur de MyViz) stocker les différents points dans un tableau. Quand celui-ci est rempli, le diagramme de Bode peut être tracé

La capture d'écran ci-dessous montre un exemple d'exploitation de la fenêtre de tracé graphique :

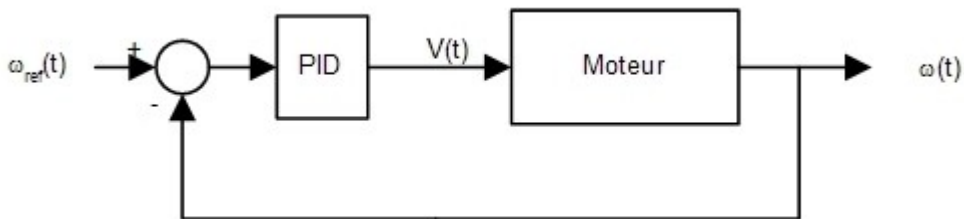


### 8.3 Asservissement des moteurs en vitesse

Cette expérience permet de piloter les moteurs du robot en boucle fermée et d'avoir une télémétrie des données mesurées. L'asservissement de vitesse est par ailleurs réglable pendant le fonctionnement du système.

Le programme Python utilisé (déjà présent sur la carte Raspberry Pi) est :  
`/root/programmes_python/AsservissementVitesse2.py`

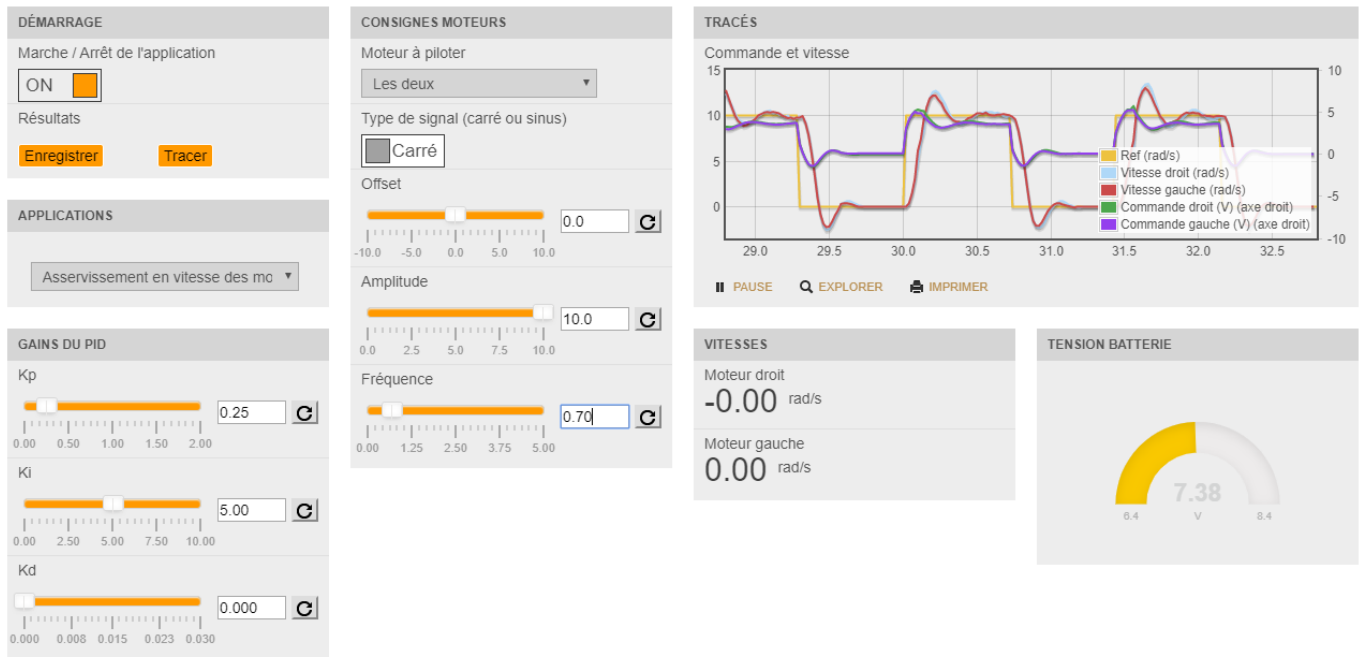
Le schéma de l'asservissement est le suivant :



La fonction de transfert du PID est la suivante :

$$K_p + \frac{K_i}{p} + \frac{K_d p}{T_f p + 1} \quad (8.1)$$

Voici une capture d'écran du tableau de bord MyViz :



Cette interface possède les fonctionnalités suivantes :

- Visualisation de la consigne de vitesse, des tensions de commande issues de l'asservissement et des vitesses mesurées
- Génération d'une consigne de vitesse sous la forme d'un signal carré ou sinusoïdal d'amplitude et de fréquence variable, avec ou sans offset
- Modification des gains du PID pendant le fonctionnement
- Visualisation de la tension de la batterie
- Tracé différé des données brutes avec sauvegarde possible dans un fichier texte
- Passage facile d'une application à une autre via une liste déroulante

Attention : les commandes de pilotages ne sont pas actives pendant les 5 premières secondes.

### 8.3.1 Expérimentations possibles

Cette expérience permet de comprendre l'intérêt et le fonctionnement d'un asservissement et de tester l'effet des 3 gains d'un régulateur PID :

- Si une des deux roues est freinée, son asservissement va tout faire pour suivre malgré tout la consigne de vitesse
- On constate alors que la tension de commande de la roue freinée est plus grande que celle de la roue « libre » : en effet, le travail de l'asservissement est de calculer la tension de commande permettant de suivre la consigne de vitesse
- La modification des gains du PID permet de constater leurs effets respectifs



## 8.4 Pilotage du robot sans asservissement de verticalité

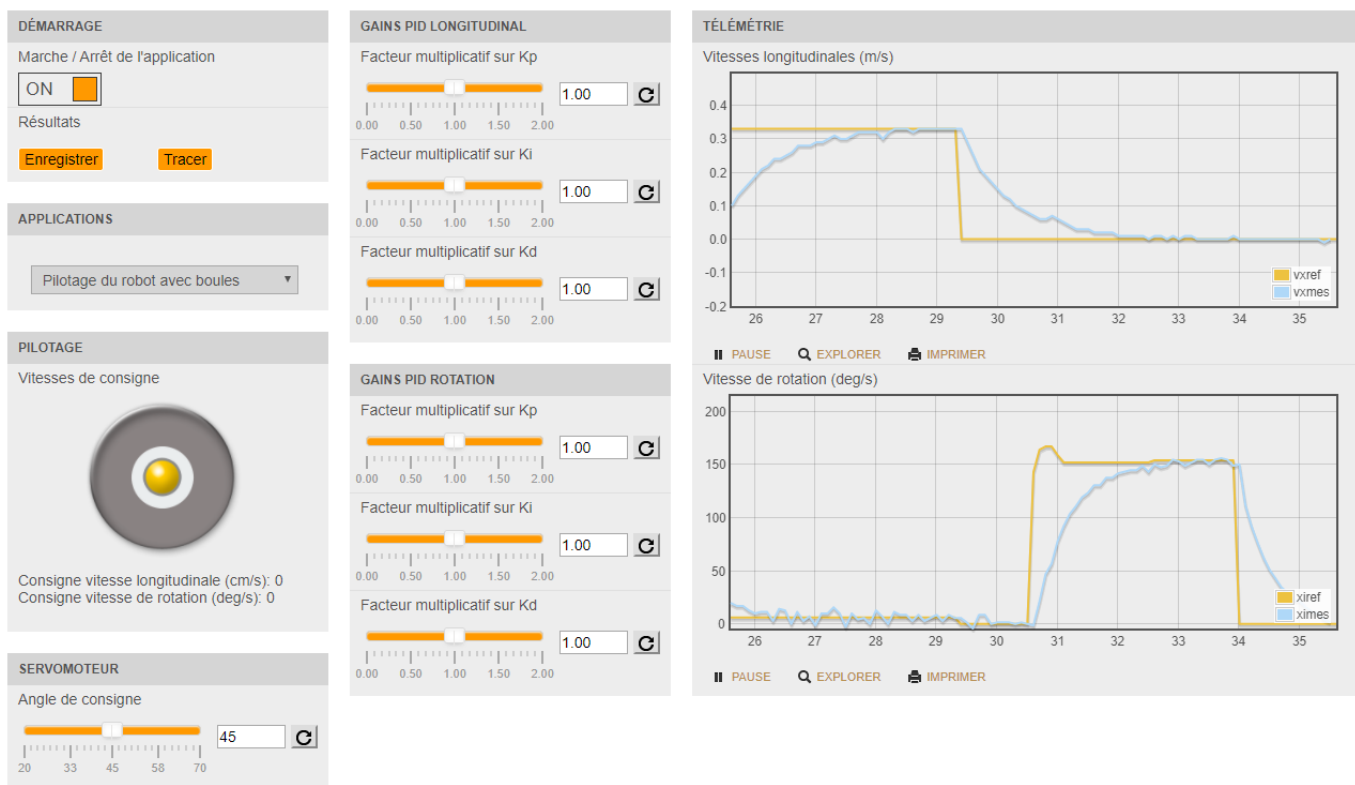
Vous pouvez convertir Geeros en un robot classique (non gyropode) en montant les deux boules omnidirectionnelles (livrées dans le colis) à l'avant et à l'arrière.

Le programme Python utilisé (déjà présent sur la carte Raspberry Pi) est :  
`/root/programmes_python/GeerosAvecBoules.py`

Cette activité permet de modifier les asservissements de vitesse du robot sans se préoccuper de l'asservissement de verticalité.

Vous pouvez également utiliser les programmes de commande en tension ou d'asservissement en vitesse des moteurs.

Voici une capture d'écran du tableau de bord MyViz :



Cette interface possède les fonctionnalités suivantes :

- Visualisation de la vitesse longitudinale (consigne et mesure) mesurée à partir de la vitesse de rotation des roues (odométrie)
- Visualisation de la vitesse de rotation (consigne et mesure) mesurée à partir de la vitesse de rotation des roues (odométrie)
- Pilotage du robot via un joystick virtuel (voir paragraphe 6.2 pour l'utilisation de ce dernier)
- Pilotage du servomoteur permettant d'orienter la Webcam
- Modification des gains du PID d'asservissement de mouvement longitudinal (via un facteur de multiplication) pendant le fonctionnement
- Modification des gains du PID d'asservissement en rotation (via un facteur de multiplication) pendant le fonctionnement
- Visualisation textuelle des données mesurées avec possibilité de sauvegarde sur l'ordinateur hôte

Attention : les commandes de pilotages ne sont pas actives pendant les 5 premières secondes.

#### 8.4.1 Expérimentations possibles

Cette expérience permet de d'appréhender l'influence de la charge sur un moteur. En effet, dans les expérimentations avec asservissement de vitesse et roues libres (robot non posé sur le sol), les moteurs doivent juste mettre les roues (de faible inertie) en mouvement. Mais dans le cadre de cette expérience, si le robot est posé au sol, les moteurs doivent mettre en mouvement l'inertie totale du robot et, de plus, compenser la résistance au roulement.

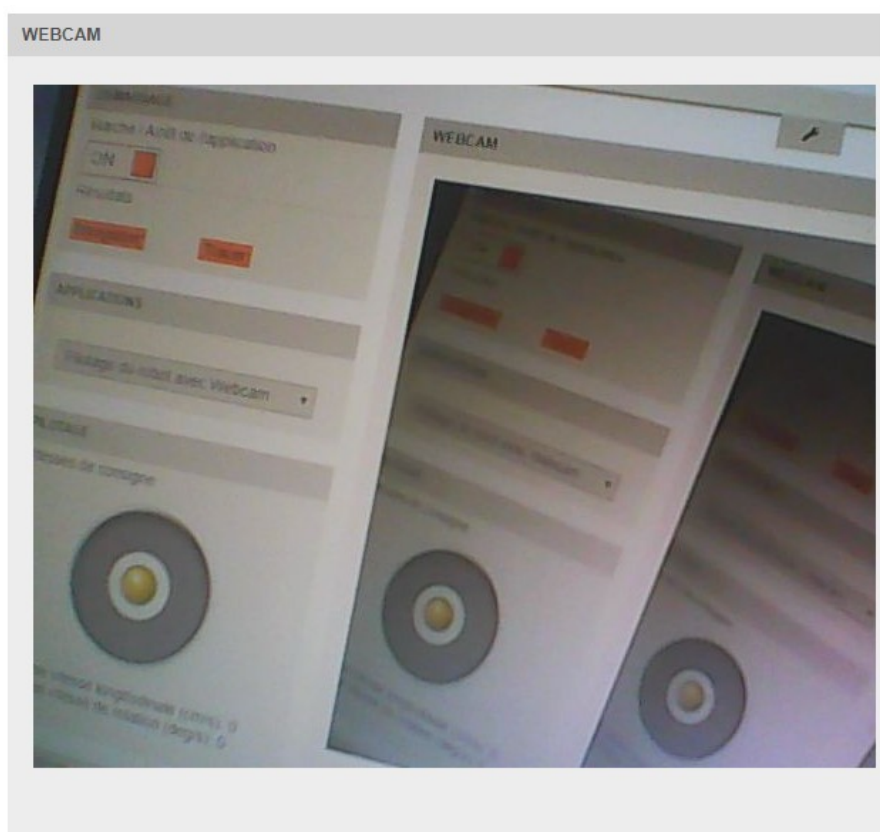
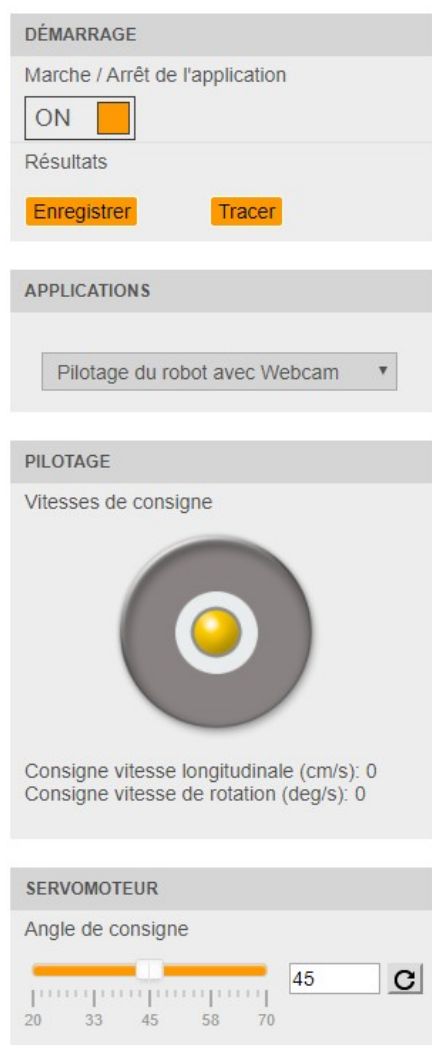
Il est intéressant de constater la différence de temps de réponse sur la rotation des moteurs si on donne une consigne de vitesse avec le robot « en l'air » ou posé au sol. On peut ainsi déterminer par expérience le facteur de multiplication à utiliser sur les gains pour que les deux temps de réponses soient similaires.

## 8.5 Pilotage du robot et visualisation des images de la Webcam

Cette activité permet de piloter le robot à distance, hors de la vue du pilote, en utilisant uniquement le retour des images de la Webcam.

Le programme Python utilisé (déjà présent sur la carte Raspberry Pi) est :  
`/root/programmes_python/Geeros.py`

Voici une capture d'écran du tableau de bord MyViz :



Cette interface possède les fonctionnalités suivantes :

- Pilotage du robot via un joystick virtuel (voir paragraphe 6.2 pour l'utilisation de ce dernier)
- Pilotage du servomoteur permettant d'orienter la Webcam
- Visualisation en temps-réel des images de la Webcam
- Tracé différé des données brutes avec sauvegarde possible dans un fichier texte
- Passage facile d'une application à une autre via une liste déroulante

Attention : les commandes de pilotages ne sont pas actives pendant les 5 premières secondes.

### 8.5.1 Expérimentations possibles

Cette expérience permet d'utiliser le robot en mode « surveillance à distance » (l'opérateur doit cependant rester à portée de signal Wifi).

Elle fait sentir la difficulté de mise en œuvre de ce genre de tâche et, pour la conception d'un tel système dans la réalité, la nécessité d'utiliser des éléments très performants. En effet, le pilotage est perturbé par :

- le champ de vision de la Webcam : il est difficile de se déplacer dans un milieu encombré d'obstacles si le champ de vision n'est pas très large
- les différentes latences (envoi de la consigne, réponse des asservissements et retour de l'image) : si ces retards ne sont pas réduits à des valeurs très petites, le pilote a (au début) des difficultés à suivre la trajectoire voulue. Il comprend ensuite qu'il doit anticiper ces retards. L'être humain, qui est dans la boucle de pilotage, agit alors comme un asservissement de type « avance de phase » car il anticipe pour compenser les retards du système.

## 8.6 Asservissement d'angle

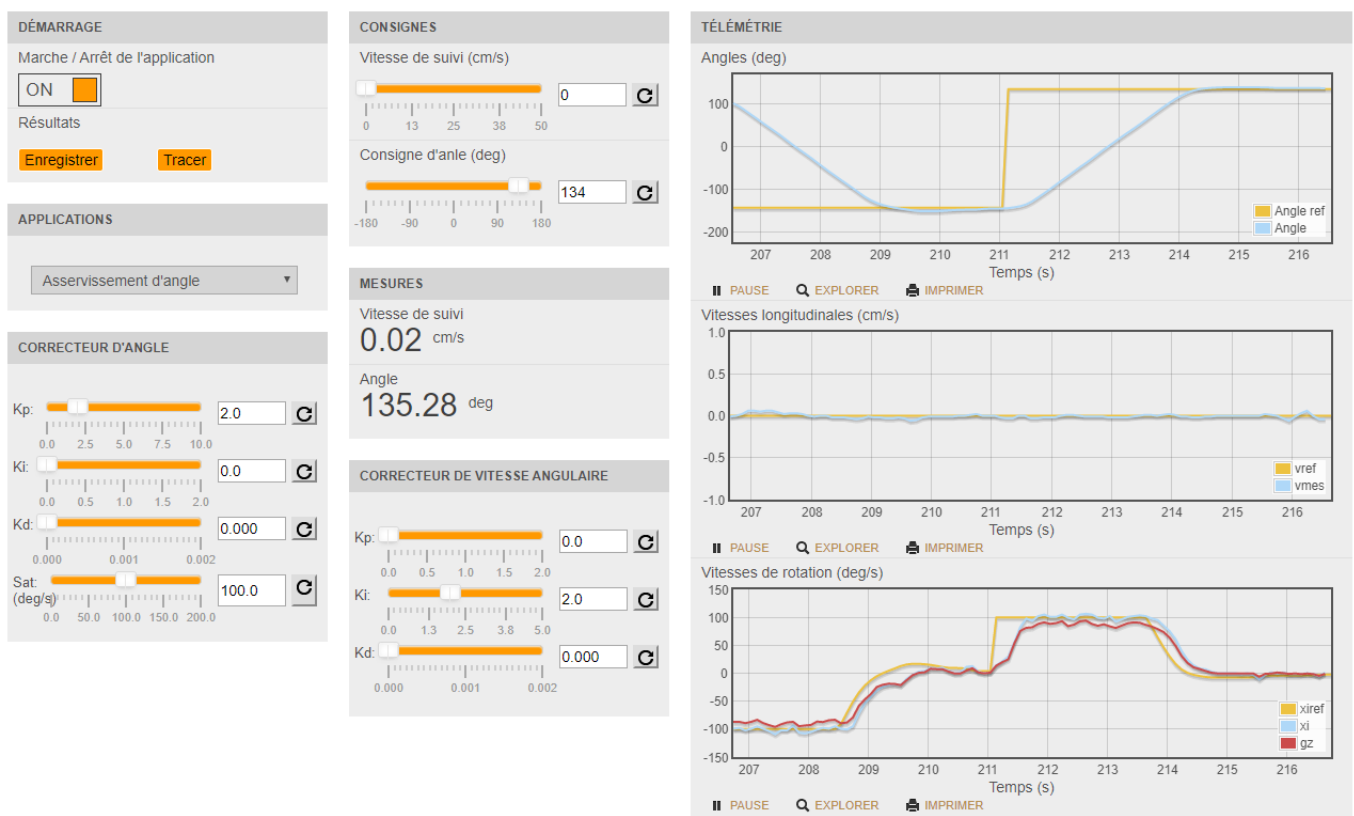
### 8.6.1 Présentation

Cette expérience permet d'asservir l'angle de rotation du robot autour de l'axe vertical, la référence (valeur nulle de l'angle) étant correspondant à la position de départ lors du démarrage du programme.

Ceci permet de travailler avec ce système sur un asservissement de type « position », alors que tous ceux vus précédemment sont des asservissements de vitesse.

Le programme Python utilisé (déjà présent sur la carte Raspberry Pi) est :  
`/root/programmes_python/AsservissementAngle.py`

Voici une capture d'écran du tableau de bord MyViz :



Cette interface possède les fonctionnalités suivantes :

- Visualisation de l'angle (consigne et mesure) mesuré à partir de la vitesse de rotation des roues (odométrie)
- Visualisation de la vitesse (consigne et mesure) mesurée à partir de la vitesse de rotation des roues (odométrie)
- Visualisation de la vitesse de rotation (consigne issue de la boucle de position, mesure odométrique et mesure issue du gyroscope)
- Définition de consigne d'angle et de vitesse longitudinale par des curseurs
- Modification des gains des PID des boucles de position et de vitesse
- Tracé différé des données brutes avec sauvegarde possible dans un fichier texte
- Passage facile d'une application à une autre via une liste déroulante

Attention : les commandes de pilotages ne sont pas actives pendant les 5 premières secondes.

### 8.6.2 Expérimentations possibles

L'asservissement est composé de deux boucles imbriquées, l'une en position et l'autre en vitesse. Les gains des deux PID sont réglables pendant le fonctionnement du système. Par ailleurs, il est possible de définir la consigne de vitesse maximale issue de la boucle de position.

Les expériences les plus intéressantes consistent à modifier les gains des deux boucles afin de voir leur influence sur les performances des asservissements.

## 8.7 Pilotage par programme Python

Cette activité permet de piloter le robot non plus via une application interactive, mais par programme, en langage Python, en utilisant une API (Application Programming Interface), c'est-à-dire une bibliothèque de fonctions utilisant dans un script Python.

### 8.7.1 Principe de fonctionnement

L'utilisation de l'API Python met en œuvre deux composantes :

- Un serveur de pilotage du robot, qui doit au préalable être lancé via une application MyViz
- Des commandes exécutées dans l'interpréteur Python intégré à MyViz

Plus précisément :

- Les commandes de l'API permettant de modifier l'état du robot (par exemple pour lui donner une consigne de vitesse longitudinale) envoient des données au serveur de pilotage, qui actionne les moteurs en conséquence
- Les commandes de l'API permettant de lire les valeurs de variables du robot réceptionnent ces valeurs lorsqu'elles sont envoyées par le serveur de pilotage

### 8.7.2 Fonctions de l'API

L'API de pilotage de Geeros en Python se trouve dans le fichier  
MyVizFiles\MyViz.nw\python\scripts\Geeros\API.py de votre installation de MyViz.

Elle contient les fonctions suivantes :

#### **geeros\_api()**

- Description : fonction d'initialisation à exécuter impérativement au début de chaque programme

#### **Avancer(vitesseLongitudinale, duree=-1)**

- Paramètres :
  - vitesseLongitudinale : réel ou chaîne de caractères (expression Python valide du type ' $3 * \text{math.sin}(t)$ ',  $t$  étant reconnu comme le temps courant). Vitesse longitudinale (cm/s), saturée en interne entre -50 et 50 cm/s
  - duree : réel (optionnel). Durée de la manœuvre en secondes. Valeur par défaut : -1 (la valeur de la tension reste constante jusqu'à ce qu'elle soit changée par une nouvelle commande)
- Description : cette fonction donne une consigne de vitesse longitudinale au robot pendant une certaine durée.

#### **Tourner(vitesseRotation, duree=-1)**

- Paramètres :
  - vitesseRotation : réel ou chaîne de caractères (expression Python valide du type ' $3 * \text{math.sin}(t)$ ',  $t$  étant reconnu comme le temps courant). Vitesse de rotation (deg/s), saturée en interne entre -360 et 360 deg/s
  - duree : réel (optionnel). Durée de la manœuvre en secondes. Valeur par défaut : -1 (la valeur de la tension reste constante jusqu'à ce qu'elle soit changée par une nouvelle commande)
- Description : cette fonction donne une consigne de vitesse de rotation au robot pendant une certaine durée.



### **Mouvement(vitesseLongitudinale , vitesseRotation, duree=-1)**

- Paramètres :
  - vitesseLongitudinale : réel ou chaîne de caractères (expression Python valide du type ' $3 * \text{math.sin}(t)$ ',  $t$  étant reconnu comme le temps courant). Vitesse longitudinale (cm/s), saturée en interne entre -50 et 50 cm/s
  - vitesseRotation : réel ou chaîne de caractères (expression Python valide du type ' $3 * \text{math.sin}(t)$ ',  $t$  étant reconnu comme le temps courant). Vitesse de rotation (deg/s), saturée en interne entre -360 et 360 deg/s
  - duree : réel (optionnel). Durée de la manœuvre en secondes. Valeur par défaut : -1 (la valeur de la tension reste constante jusqu'à ce qu'elle soit changée par une nouvelle commande)
- Description : cette fonction donne une consigne de mouvement au robot (c'est-à-dire une combinaison de consigne de vitesse longitudinale et de vitesse de rotation) pendant une certaine durée.

### **AngleServo(angle, duree=-1)**

- Paramètres :
  - angle : réel ou chaîne de caractères (expression Python valide du type ' $3 * \text{math.sin}(t)$ ',  $t$  étant reconnu comme le temps courant). Angle d'orientation du servomoteur (deg), saturé en interne entre 20 et 70 deg
  - duree : réel (optionnel). Durée de la manœuvre en secondes. Valeur par défaut : -1 (la valeur de la tension reste constante jusqu'à ce qu'elle soit changée par une nouvelle commande)
- Description : cette fonction donne une consigne d'angle d'orientation du servomoteur portant la Webcam pendant une certaine durée.

### **LireVariable(variable)**

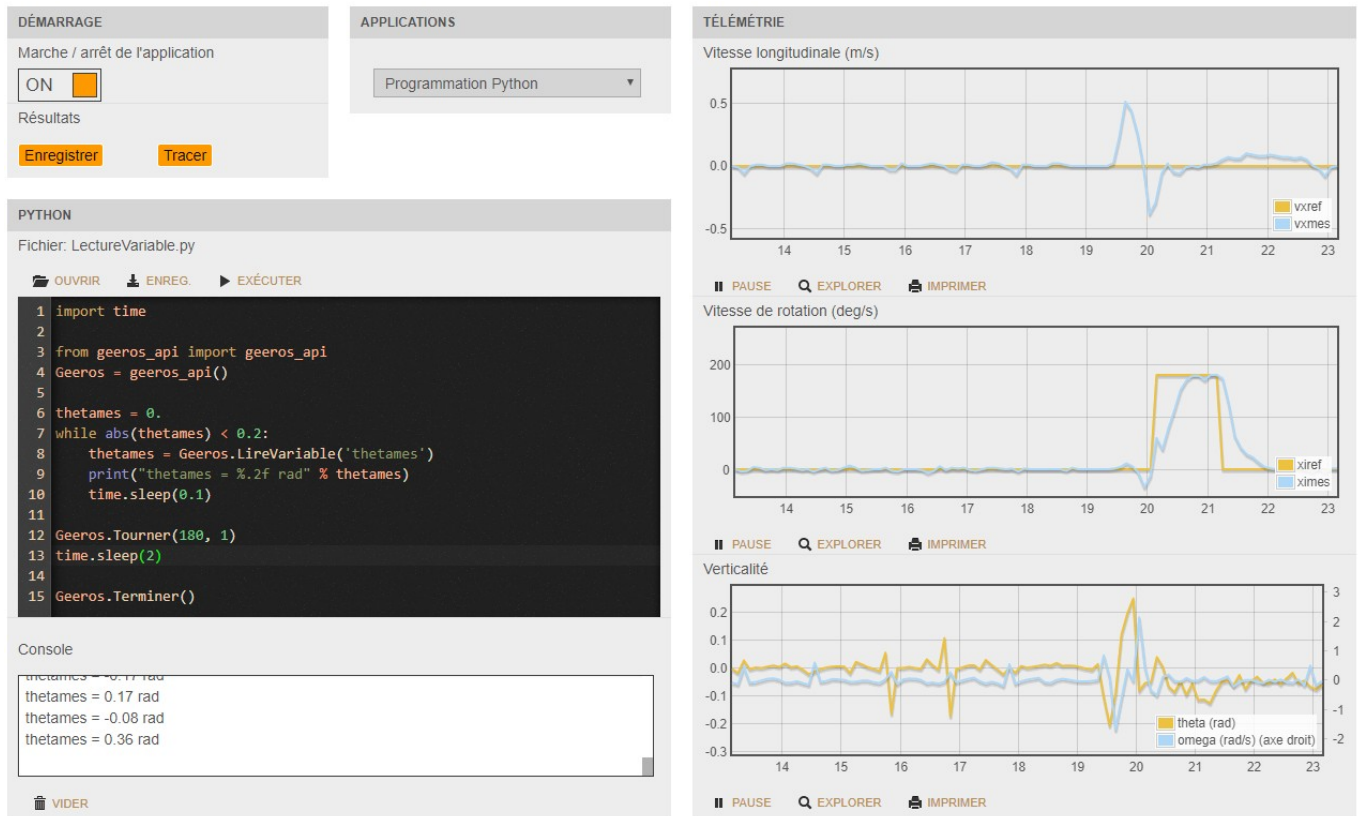
- Paramètres :
  - variable : chaîne de caractères. Variable à lire, parmi la liste suivante :
    - 'Temps' : temps courant (s)
    - 'commandeDroit' : tension de commande (V) envoyée au moteur droit
    - 'commandeGauche' : tension de commande (V) envoyée au moteur gauche
    - 'omega' : vitesse de rotation du robot par rapport à la verticale (rad/s)
    - 'omegaDroit' : vitesse de rotation (rad/s) du moteur droit
    - 'omegaGauche' : vitesse de rotation (rad/s) du moteur gauche
    - 'thetames' : tension de commande du moteur (V)
    - 'Consigne vitesse longitudinale' : tension de commande du moteur (V)
    - 'Consigne vitesse de rotation' : tension de commande du moteur (V)
    - 'Vitesse longitudinale' : tension de commande du moteur (V)
    - 'Vitesse de rotation' : tension de commande du moteur (V)
- Description : cette fonction permet de lire la valeur d'une variable du système

### **Terminer()**

- Description : cette fonction doit être appelée à la fin de chaque programme

Le programme Python utilisé (déjà présent sur la carte Raspberry Pi) est :  
/root/programmes\_python/Geeros.py

Voici une capture d'écran du tableau de bord MyViz :



Cette interface possède les fonctionnalités suivantes :

- Visualisation de la vitesse longitudinale (consigne et mesure) mesurée à partir de la vitesse de rotation des roues (odométrie)
- Visualisation de la vitesse de rotation (consigne et mesure) mesurée à partir de la vitesse de rotation des roues (odométrie)
- Visualisation de l'angle et de la vitesse de rotation du robot par rapport à la verticale
- Pilotage du robot via des commandes Python entrées dans un éditeur intégré
- Tracé différé des données brutes avec sauvegarde possible dans un fichier texte
- Passage facile d'une application à une autre via une liste déroulante

### 8.7.3 Exemples d'utilisation de l'API

Les exemples présentés ci-dessous doivent être exécutés dans la zone de programmation Python du tableau de bord. Les fichiers Python correspondant se trouvent dans le répertoire MyVizFiles\MyViz.nw\dashboards\Geeros de votre installation de MyViz.

#### 8.7.3.1 Aller-retour

Le robot avance à la vitesse de 30 cm/s pendant 3 s, fait demi-tour, revient à la même vitesse et pivote de nouveau :

```
from geeros_api import geeros_api

Geeros = geeros_api()

Geeros.Avancer(30,3)

Geeros.Tourner(180,1)

Geeros.Avancer(30,3)

Geeros.Tourner(180,1)

Geeros.Terminer()
```

#### 8.7.3.2 Rotation en suivant une formule

Ce programme illustre, sur une rotation autour de l'axe vertical, la possibilité de donner des consignes sous la forme de formules :

```
from geeros_api import geeros_api

Geeros = geeros_api()

Geeros.Tourner("180 * math.sin(2*t)", 10)

Geeros.Terminer()
```

### ***8.7.3.3 Combinaison de mouvements***

La combinaison d'une vitesse longitudinale et d'une vitesse de rotation conduit le robot à suivre une trajectoire circulaire :

```
from geeros_api import geeros_api

Geeros = geeros_api()

Geeros.Mouvement(30, 180, 10)

Geeros.Terminer()
```

### ***8.7.3.4 Pilotage du servomoteur portant la Webcam***

Ce programme donne des consignes d'orientation successives du servomoteur :

```
from geeros_api import geeros_api

Geeros = geeros_api()

Geeros.AngleServo(20, 1)

Geeros.AngleServo(70, 1)

Geeros.AngleServo(45, 1)

Geeros.Terminer()
```

#### 8.7.3.5 Rotation conditionnelle

Ce programme permet de déclencher une rotation du robot quand son angle par rapport à la verticale dépasse 0.2 rad (en valeur absolue). Ce déclenchement se fait par exemple en appuyant sur l'avant du robot alors qu'il est dans son état d'équilibre asservi en maintien de verticalité :

```
import time

from geeros_api import geeros_api

Geeros = geeros_api()

thetames = 0.

while abs(thetames) < 0.2:

    thetames = Geeros.LireVariable('thetames')

    print("thetames = %.2f rad" % thetames)

    time.sleep(0.1)

Geeros.Tourner(180, 1)

time.sleep(2)

Geeros.Terminer()
```

## 9 Activités réalisables avec le système en mode Arduino / C

Dans ce mode, les différentes applications sont programmées en C (Arduino) et sont exécutées par la carte A-Star. Il est nécessaire de reprogrammer cette dernière pour chaque application.

Les différentes applications du système dans ce mode sont très similaires à celles présentées dans le mode Raspberry Pi / Python. Le choix d'utiliser un mode plutôt que l'autre est essentiellement une question de préférence de langage de programmation ou d'utilisation de plate-forme (micro-contrôleur ou mini-ordinateur).

Par conséquent, nous indiquons par la suite uniquement les différences par rapport aux applications présentées au chapitre précédent et en particulier le programme Arduino à charger sur la carte A-Star pour chacune d'elle. N'hésitez pas à le modifier en fonction de vos besoins.

**Important :** l'utilisation de MyViz dans ce mode nécessite de lui indiquer le numéro de port série sur lequel est connecté le robot, en suivant la procédure ci-dessous :

- Allumer le robot, posé sur l'arrière
- Brancher le câble mini USB fourni sur le connecteur micro USB de la carte A-Star
- Lancer le logiciel MyViz
- Ouvrir un des tableaux de bord du robot Geeros nécessitant une liaison série (commande en tension ou asservissement en vitesse)
- Aller dans Paramètres → Port série par défaut et sélectionner le port série de l'ordinateur sur lequel est connecté le robot
- Démarrer l'interaction en cliquant sur le bouton "ON / OFF" en haut à gauche du tableau de bord:



- Les acquisitions doivent démarrer très rapidement. Vous pouvez alors interagir avec le système grâce aux contrôles prévus à cet effet.

## 9.1 Commande des moteurs en tension

Cette expérience (analogue à celle du paragraphe 8.2) permet de donner une consigne de tension aux moteurs du robot et d'avoir une télémétrie (via liaison série) des données mesurées.

Le programme Arduino qui doit être téléversé sur la carte A-Star est :  
CommandeMoteursEnTension\_1.ino

## 9.2 Asservissement des moteurs en vitesse

Cette expérience (analogue à celle du paragraphe 8.3) permet de piloter les moteurs du robot en boucle fermée et d'avoir une télémétrie (via liaison série) des données mesurées. L'asservissement de vitesse est par ailleurs réglable pendant le fonctionnement du système.

Le programme Arduino qui doit être téléversé sur la carte A-Star est :  
AsservissementMoteursEnVitesse\_1.ino



## 9.3 Trajectoire pré-programmée avec asservissement de verticalité

### 9.3.1 Présentation

Contrairement au mode Raspberry Pi / Python (voir 8.1), on ne fait pas ici de pilotage à distance et on n'utilise pas de tableau de bord car le robot n'est pas libre de ses mouvements s'il est connecté à l'ordinateur. L'objectif dans cette version « Arduino » est de programmer cette carte micro-contrôleur avec des manœuvres pré-déterminées.

Le programme Arduino est :

GeerosDefinitionMouvement\_1.ino

La programmation de manœuvres se fait en modifiant la partie « Consignes » du programme. Par exemple, le code suivant permet de réaliser une suite de virages :

```
if (temps > Tmax) {
    Tmax = temps + 6.; // Le 8 dure 6 s
}
else if (temps < (Tmax-5.5)) { // La ligne droite dure 0.5 s
    vxref = 0.3;
    xiref = 0.;
}
else if (temps < (Tmax-3.)) { // Le virage dure 2.5 s
    vxref = 0.3;
    xiref = 2.;
}
else if (temps < (Tmax-2.5)) { // La ligne droite dure 0.5 s
    vxref = 0.3;
    xiref = 0.;
}
else { // // Le virage dure 2.5 s
    vxref = 0.3;
    xiref = -2.;
}
```

**Important** : dans la partie « Setup » du programme, il est nécessaire d'attendre au moins 30 secondes pour que la Raspberry Pi ait démarré et ne perturbe pas l'A-Star.

## 9.4 Trajectoire pré-programmée sans asservissement de verticalité

Le principe est le même qu'avec l'asservissement de verticalité (voir 9.3), sauf qu'on utilise ici le robot après lui avoir ajouté ses deux boules omnidirectionnelles optionnelles.

Le programme Arduino est :

GeerosAvecBoulesDefinitionMouvement\_1.ino

**Important** : dans la partie « Setup » du programme, il est nécessaire d'attendre au moins 30 secondes pour que la Raspberry Pi ait démarré et ne perturbe pas l'A-Star.

## 10 Support technique

Pour toute question ou problème, veuillez nous contacter à l'adresse [support@3sigma.fr](mailto:support@3sigma.fr).

Par ailleurs, ce système est un produit « vivant ». N'hésitez pas à vérifier régulièrement la présence de nouvelles versions de la documentation ou des programmes et tableaux de bord.