



# ENSEMBLE « COMMANDE DE MOTEUR À COURANT CONTINU » PCDUINO NANO

DOCUMENTATION COMPLÈTE



Date de dernière mise à jour : 10/10/2017

## Table des matières

---

<b>1 - Introduction.....</b>	<b><u>3</u></b>
<b>2 - Matériel inclus.....</b>	<b><u>3</u></b>
<b>3 - Conformité.....</b>	<b><u>4</u></b>
<b>4 - Mise en œuvre de l'ensemble.....</b>	<b><u>5</u></b>
4.1 - Opérations préalables.....	<u>5</u>
4.2 - Utilisation standard.....	<u>5</u>
4.3 - Précautions d'emploi.....	<u>5</u>
<b>4.3.1 - Connexions d'alimentation sur la carte de commande moteur.....</b>	<b><u>5</u></b>
<b>4.3.2 - Tension d'alimentation.....</b>	<b><u>6</u></b>
<b>4.3.3 - Utilisation.....</b>	<b><u>6</u></b>
<b>5 - Expériences de commande de moteur à courant continu.....</b>	<b><u>7</u></b>
5.1 - Caractéristiques du moteur à courant continu avec codeur incrémental associé.....	<u>7</u>
<b>5.1.1 - Calcul de la vitesse de rotation du moteur.....</b>	<b><u>9</u></b>
<b>5.1.2 - Comptage du nombre d'impulsions.....</b>	<b><u>10</u></b>
5.2 - Applications compatibles uniquement avec le matériel en version 2.....	<u>11</u>
<b>5.2.1 - Application MyViz de commande en tension.....</b>	<b><u>11</u></b>
<b>5.2.2 - Application MyViz d'asservissement de vitesse.....</b>	<b><u>15</u></b>
<b>5.2.3 - Application MyViz d'asservissement de position.....</b>	<b><u>19</u></b>
5.3 - Applications compatibles avec toutes les versions de matériel.....	<u>23</u>
<b>5.3.1 - Application MyViz de commande en tension.....</b>	<b><u>23</u></b>
<b>5.3.2 - Application MyViz d'asservissement de vitesse.....</b>	<b><u>27</u></b>
<b>5.3.3 - Ancienne interface graphique : commande en tension.....</b>	<b><u>31</u></b>
<b>5.3.4 - Ancienne interface graphique : asservissement de vitesse.....</b>	<b><u>34</u></b>
<b>6 - API Python.....</b>	<b><u>38</u></b>
6.1 - Fonctions de l'API.....	<u>39</u>
6.2 - Lancement du serveur d'exécution.....	<u>41</u>
<b>6.2.1 - Lancement du serveur d'exécution en ligne de commande.....</b>	<b><u>41</u></b>
<b>6.2.2 - Lancement du serveur d'exécution via MyViz.....</b>	<b><u>44</u></b>
6.3 - Exemples d'utilisation de l'API.....	<u>47</u>
<b>6.3.1 - Consigne de position sinusoïdale.....</b>	<b><u>47</u></b>
<b>6.3.2 - Consigne de vitesse suivant un signal carré.....</b>	<b><u>48</u></b>
<b>7 - Accès aux programmes Python.....</b>	<b><u>49</u></b>
<b>8 - Important.....</b>	<b><u>50</u></b>

## 1 - Introduction

Cet ensemble permet de réaliser différentes expériences sur la base d'un moteur à courant continu avec codeur incrémental associé, commandé par une carte pcDuino Nano.

Cette documentation, ainsi que les programmes associés sont téléchargeables sur notre site Web à l'adresse suivante:

<http://www.3sigma.fr/telechargements>

## 2 - Matériel inclus

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

- un boîtier en plexiglas blanc, avec connecteur d'alimentation 5.5mm x 2.1mm, bouton marche-arrêt et douilles 2mm pour la mesure à l'oscilloscope de différents signaux.

Repérage des douilles 2mm:

Rouge	Bleu	Jaune	Blanc	Vert	Noir
+ moteur	- moteur	Voie A codeur	Voie B codeur	Signal PWM	Masse

- une carte pcDuino Nano
- un micro-contrôleur dédié à la mesure de vitesse (uniquement dans les matériels livrés après le 15 juin 2017)
- un moteur à courant continu 6V, rapport de réduction 9.7:1, avec codeur incrémental 48 CPR. Le moteur est assemblé sur le boîtier par l'intermédiaire d'un support de montage en équerre
- 1 adaptateur USB / Ethernet permettant de communiquer directement entre un ordinateur (connexion USB) et le pcDuino (connexion Ethernet)
- 1 câble Ethernet
- 1 alimentation 9V, 1A avec connecteur d'alimentation 5.5mm x 2.1mm

Remarque : comme indiqué ci-dessus, le système intègre depuis le 15 juin 2017 un micro-contrôleur dédié à la mesure de vitesse du moteur. Sur les systèmes antérieurs, la vitesse est mesurée directement sur la carte pcDuino.

L'ajout du micro-contrôleur dédié permet d'améliorer les performances de comptage des interruptions générées par le codeur incrémental du moteur (voir paragraphe 5.1.1)

Noter que les nouveaux systèmes (parfois dénommés « Version 2 » dans la suite de ce document) sont compatibles avec les anciens : ils peuvent également mesurer la vitesse du moteur uniquement grâce à la carte pcDuino.

## 3 - Conformité

L'ensemble « Commande de moteur à courant continu », **dans sa configuration livrée aux clients**, est conforme à la directive 1999/EC.

## 4 - Mise en œuvre de l'ensemble

### 4.1 - Opérations préalables

Attention : avant la première utilisation du système, vous devez installer sur votre ordinateur le pilote de l'adaptateur USB / Ethernet fourni avec le système. Ce pilote se trouve sur le CDROM livré avec l'adaptateur.

### 4.2 - Utilisation standard

La mise en œuvre de l'ensemble « Commande de moteur à courant continu » est très simple:

- brancher l'alimentation 9V fournie sur le connecteur jack du système
- connecter l'ordinateur au système pcDuino en utilisant le câble Ethernet et l'adaptateur USB – Ethernet (USB côté ordinateur)
- commuter l'interrupteur sur la position « I »
- attendre environ une minute le démarrage de la carte pcDuino
- exécuter l'un des logiciels de pilotage (voir plus loin)

### 4.3 - Précautions d'emploi

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

#### 4.3.1 - Connexions d'alimentation sur la carte de commande moteur

Il est impératif de faire très attention aux connexions de l'alimentation du shield de commande moteur car celui-ci n'est pas protégé contre les inversions de polarité. Une erreur de connexion sur les bornes d'alimentation risque d'entraîner la destruction du sous-ensemble de gestion d'alimentation de la carte et de rendre celle-ci inutilisable. L'ensemble « Commande de moteur à courant continu » étant livré connecté et fonctionnel, il est préférable de ne pas modifier les branchements sur les connecteurs d'alimentation.

### 4.3.2 - Tension d'alimentation

---

Cet ensemble est prévu pour fonctionner avec l'alimentation 9V, 1A fournie. Le moteur a une tension nominale de 6V et bien qu'il supporte sans problème des tensions jusqu'à 9V, les programmes disponibles sur la carte pcDuino (ainsi que tous les programmes téléchargeables sur notre site) empêchent que la tension à ses bornes dépasse la valeur de 6V.

Il est possible d'utiliser un autre bloc d'alimentation à condition de bien respecter les points suivants:

- la tension ne doit pas dépasser 9V. Il est cependant recommandé de mettre éventuellement les programmes de la carte pcDuino à jour si ceux-ci sont basés sur une valeur de tension différente
- la polarité doit être « positif au centre du connecteur »
- l'alimentation doit pouvoir fournir un courant suffisant, de préférence supérieur ou égal à 1A

Notez que le système intègre un régulateur de tension 5V pour alimenter la carte pcDuino via la broche 5V du shield de commande moteur.

### 4.3.3 - Utilisation

---

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 vies des éléments.

## 5 - Expériences de commande de moteur à courant continu

Cet ensemble permet de réaliser différentes expériences de commande de moteur à courant continu.

### 5.1 - Caractéristiques du moteur à courant continu avec codeur incrémental associé

L'expérience de commande de moteur électrique embarque un moteur à courant continu 6V, de rapport de réduction 9.7:1, avec codeur incrémental 48 CPR (Counts Per Revolution).

Ses équations sont les suivantes:

$$\frac{d}{dt} \omega_m(t) = \frac{\text{ratio} K i_m(t) - d \omega_m(t)}{J \cdot \text{ratio}^2}$$

$$\frac{d}{dt} i_m(t) = \frac{V(t) - R i_m(t) - K \cdot \text{ratio} \omega_m(t)}{L}$$

Avec :

- R : résistance électrique interne: 3.0 Ohms
- L : inductance des enroulements: 3.0 mH
- J : moment d'inertie du rotor:  $3 \cdot 10^{-6}$  kg.m<sup>2</sup>
- K : constante de couple = constante de fem: 0.01 N.m/A
- d : coefficient de frottement visqueux: 0.00025 N.m.s/rad
- $\omega_m$  : vitesse de rotation de l'arbre de sortie du réducteur (rad/s)
- $i_m$  : courant dans le moteur (A)
- V : tension d'alimentation (V)

Ces paramètres ont été identifiés à partir d'un essai de réponse du moteur à un échelon de tension.

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

$$\frac{K \text{ratio}}{JL \text{ratio}^2 s^2 + (JR \text{ratio}^2 + Ld) s + K^2 \text{ratio}^2 + Rd}$$

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

$$\frac{J \text{ratio}^2 s + d}{JL \text{ratio}^2 s^2 + (JR \text{ratio}^2 + Ld) s + K^2 \text{ratio}^2 + Rd}$$

Noter que les valeurs numériques données ci-dessus concernent l'identification du moteur avec son driver, l'influence de ce dernier étant importante et devant être pris en compte dans l'asservissement global du système.

Le brochage du moteur (couleur des fils) est donné dans le tableau ci-dessous:

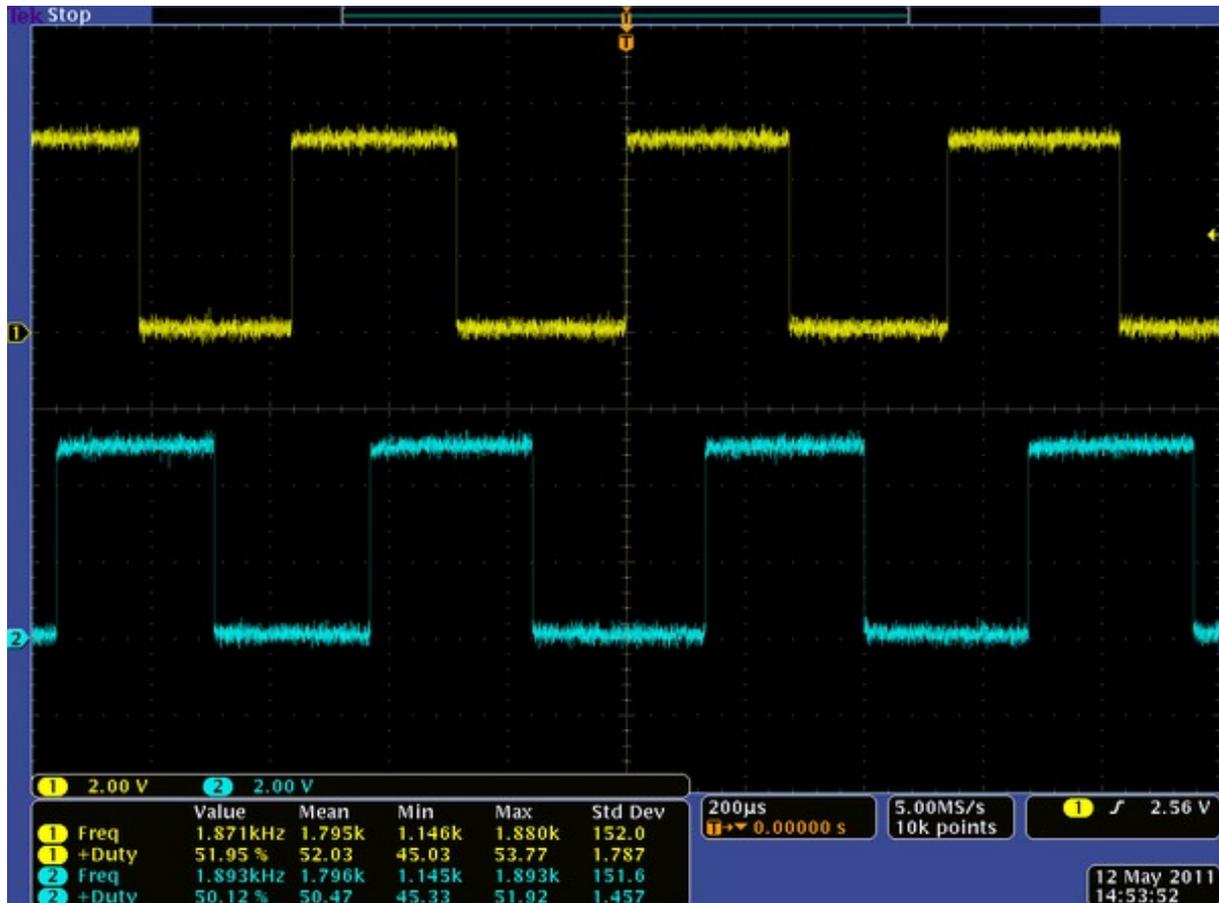
Rouge	Noir	Bleu	Vert	Jaune	Blanc
+ moteur	- moteur	5V codeur	Masse codeur	Voie A codeur	Voie B codeur

**ATTENTION !**

Ne pas confondre la couleur des fils du moteur et la couleur des douilles 2mm.

## 5.1.1 - Calcul de la vitesse de rotation du moteur

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 9.7:1, l'arbre moteur fait 9.7 tours lorsque l'arbre « principal » en fait 1
- Le codeur génère 48 impulsions à chaque fois qu'il fait un tour
- La cadence d'échantillonnage utilisée pour l'asservissement sera de 0.01 s

Par conséquent, lorsque l'arbre principal fait un tour, le codeur génère :  
 $9.7 * 48 \approx 465$  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  $2\pi$  radians) :

$$2\pi * N / (0.01 * 465)$$

**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) :

$$2*\pi/(Ts*CPR*ratio)$$

avec :

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

Dans notre cas de figure, la résolution est la suivante

$$2*\pi/(0.01*465) = 1.35 \text{ rad/s}$$

## 5.1.2 - Comptage du nombre d'impulsions

---

Compter le nombre d'impulsions du codeur revient à compter le nombre de fronts montants et descendants des signaux jaune et bleu représentés sur l'image ci-dessus. Pour ce faire, la seule méthode viable consiste à brancher les deux signaux (les fils jaune et blanc sur le codeur utilisé) sur deux entrées « interruption » de la carte pcDuino. Les deux autres fils (bleu et vert) seront respectivement branchés sur le 3.3 V et sur la masse du pcDuino.

Sur une carte pcDuino, il y a deux lignes d'interruption (numérotées 0 et 1), qui correspondent aux broches digitales 2 et 3. 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.

En ce qui concerne l'interruption liée à la voie B, c'est l'inverse :

- Lorsque la voie B passe au niveau haut, la voie A est au niveau haut
- Lorsque la voie B passe au niveau bas, la voie A est au niveau bas

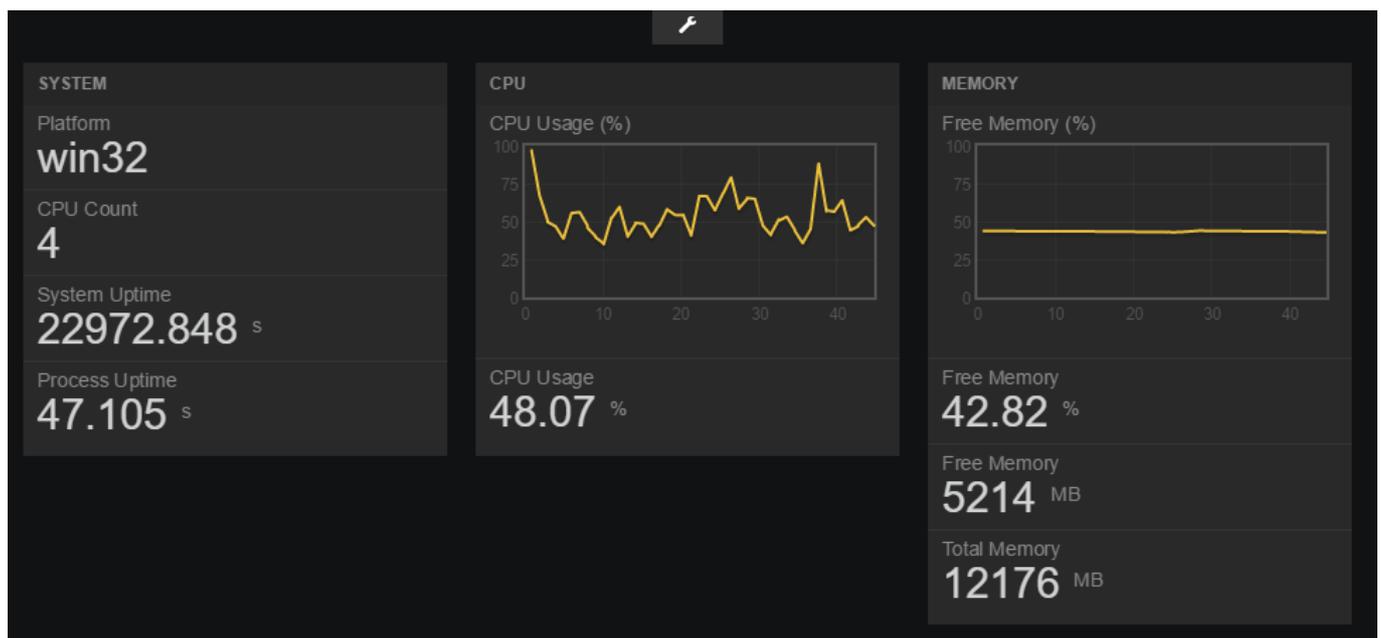
## 5.2 - Applications compatibles uniquement avec le matériel en version 2

### 5.2.1 - Application MyViz de commande en tension

**ATTENTION : Cette application est compatible uniquement avec les systèmes livrés après le 15 juin 2017. Si votre système est plus ancien, vous devez utiliser l'application MyViz correspondante (voir paragraphe 5.3.1) ou bien l'ancienne interface graphique (voir paragraphe 5.3.3)**

Cette activité utilise le logiciel MyViz, très souple pour créer des tableaux de bord de pilotage et de visualisation de données.

Après l'avoir téléchargé (<http://www.3sigma.fr/Telechargements-MyViz.html>) et installé, lancez son exécution. Le tableau de bord initialement affiché sera similaire à la capture d'écran ci-dessous :



Ce tableau de bord n'est qu'un exemple de ce qui peut être réalisé avec MyViz. Nous verrons plus loin comment charger celui correspondant à l'expérience que nous souhaitons réaliser dans ce chapitre.

Pour réaliser cette activité, les conditions suivantes doivent être remplies :

- l'ordinateur hôte doit être connecté à la carte pcDuino (voir chapitre 4)
- le système doit être allumé

Charger ensuite le tableau de bord de l'activité dans MyViz. Pour cela, il faut tout d'abord récupérer ce

dernier sur votre ordinateur, à partir du lien suivant :

[https://raw.githubusercontent.com/3sigma/CommandeMoteurElectrique/master/MyViz/CommandeMoteurEnTension2\\_Reseau.json](https://raw.githubusercontent.com/3sigma/CommandeMoteurElectrique/master/MyViz/CommandeMoteurEnTension2_Reseau.json)

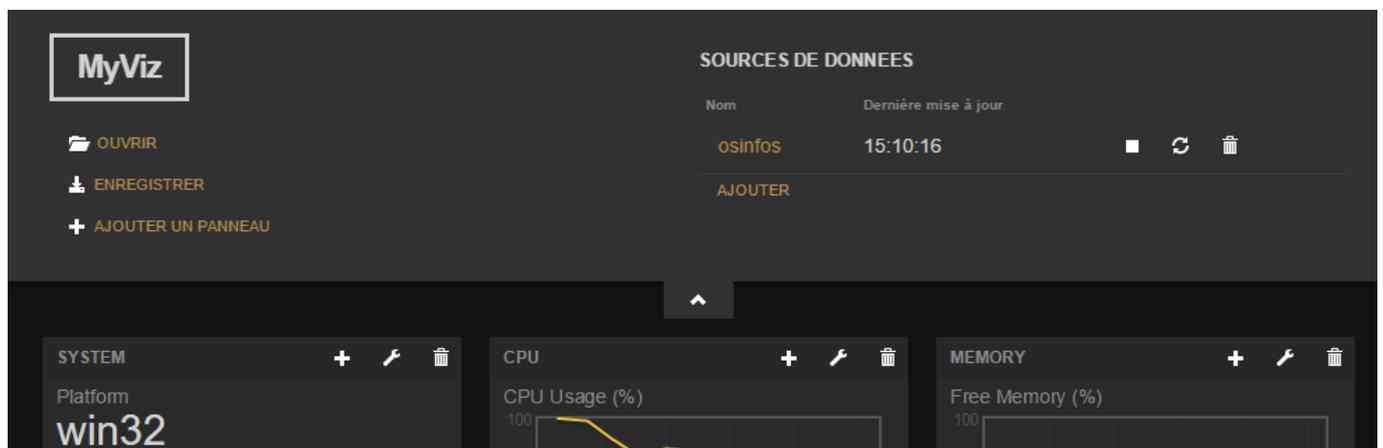
Il se trouve également dans l'archive suivante :

<https://github.com/3sigma/CommandeMoteurElectrique/archive/master.zip>

Pour l'ouvrir dans MyViz, il suffit ensuite de cliquer sur la clé en haut de la fenêtre :

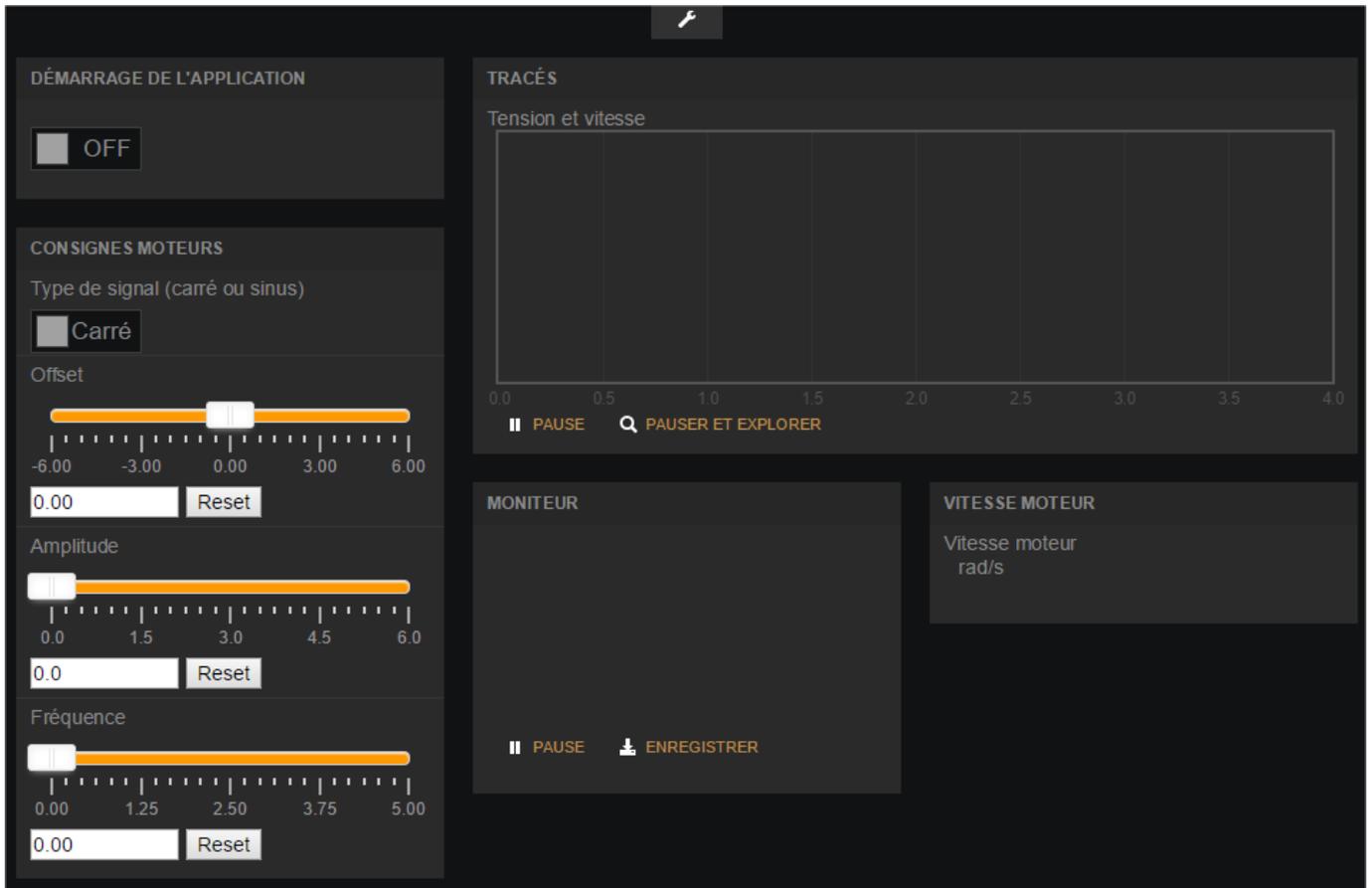


Ceci permet de déplier le panneau supérieur :



Cliquez sur « Ouvrir » et sélectionnez le fichier CommandeMoteurEnTension2\_Reseau.json que vous venez de télécharger.

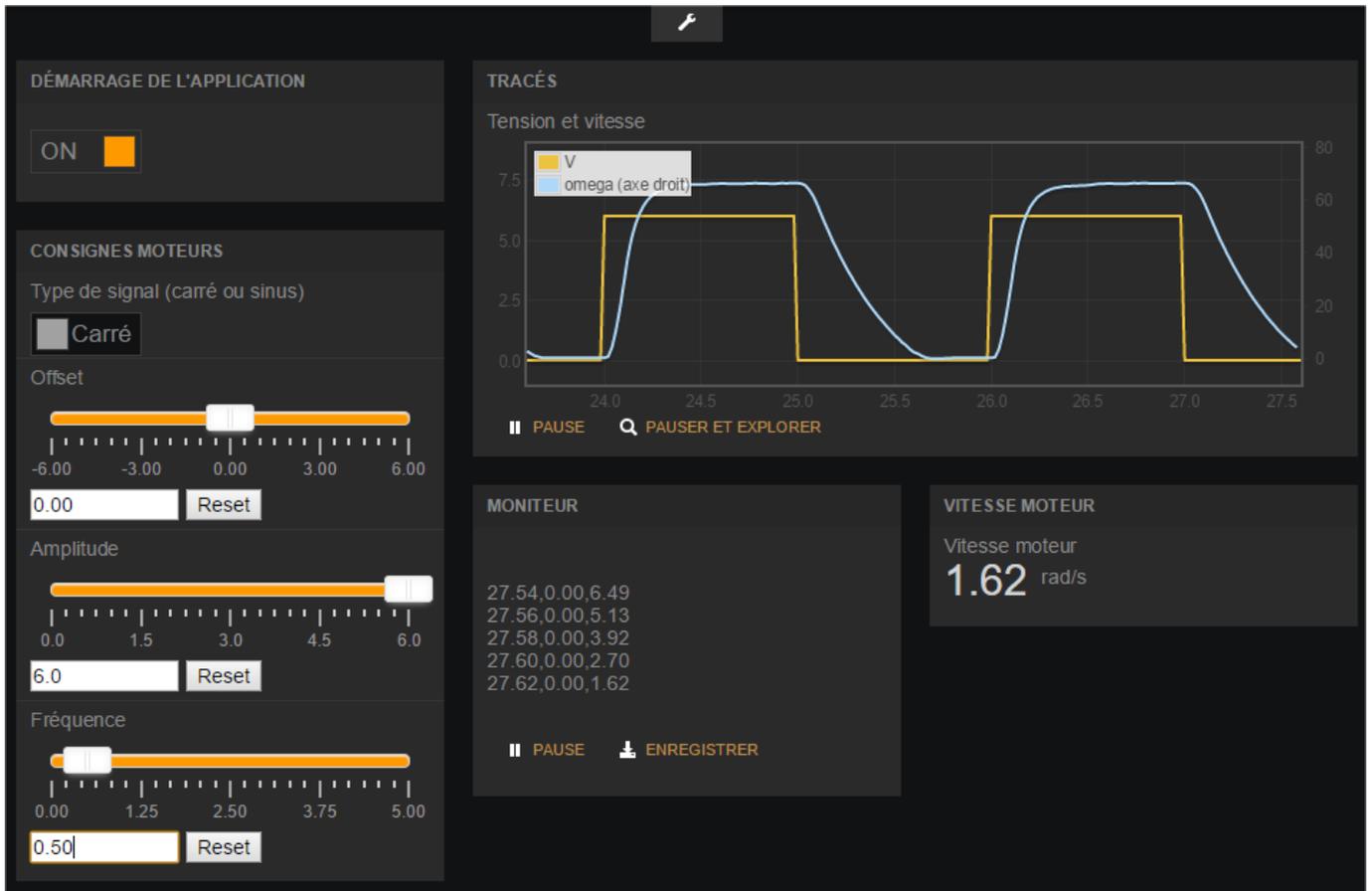
Le tableau de bord s'affiche alors :



Son utilisation est a priori intuitive. Il faut cependant noter les points suivants :

- le démarrage de l'application se fait via la bouton marche-arrêt en haut à gauche.  
**Attention** : il faut attendre quelques secondes avant de voir apparaître les courbes de télémétrie et de pouvoir piloter la tension
- bien que la plage de variation des curseurs soit modifiable (le passage en mode « édition » du tableau de bord se fait en cliquant sur la clé en haut de l'écran), la tension de commande envoyée aux moteurs est saturée par programme à  $\pm 6V$

En fonctionnement, ce tableau de bord peut avoir l'allure suivante :



Il permet de réaliser diverses expériences :

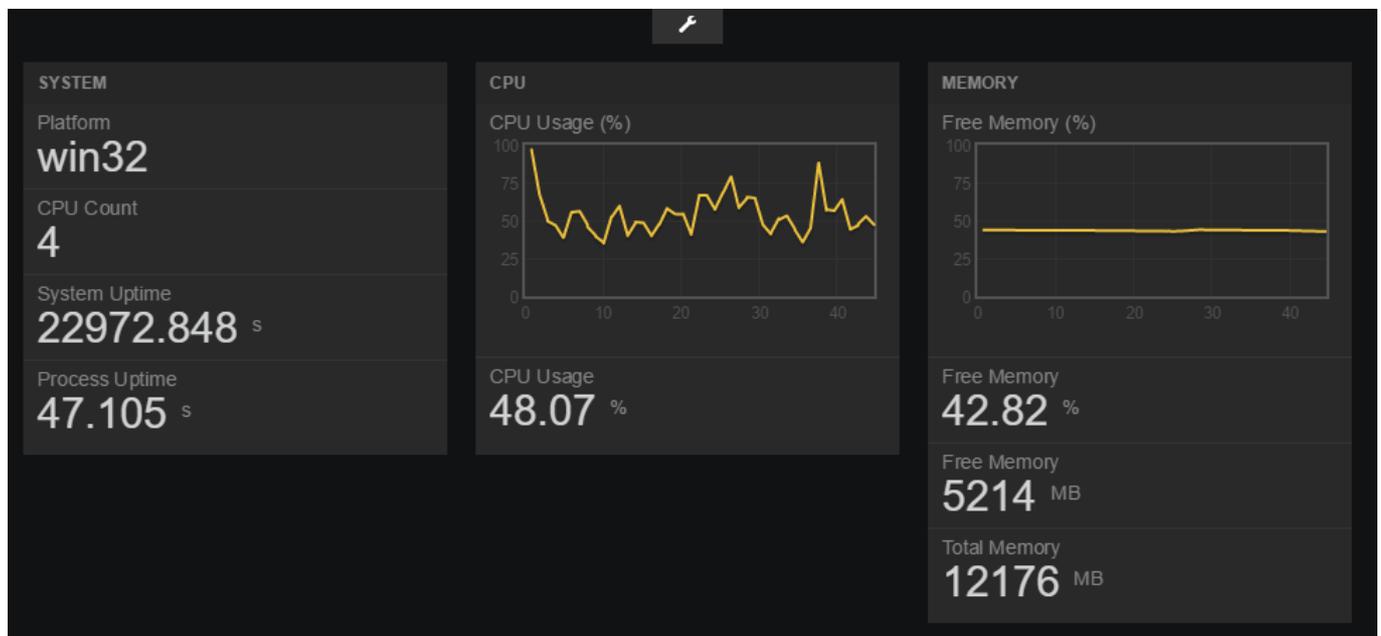
- réponse indicielle
- mesure de l'amplitude et du déphasage de la vitesse du moteur soumis à une tension sinusoïdale de fréquence variable, ce qui permet de tracer un diagramme de Bode du système en boucle ouverte

## 5.2.2 - Application MyViz d'asservissement de vitesse

**ATTENTION : Cette application est compatible uniquement avec les systèmes livrés après le 15 juin 2017. Si votre système est plus ancien, vous devez utiliser l'application MyViz correspondante (voir paragraphe 5.3.2) ou bien l'ancienne interface graphique (voir paragraphe 5.3.4)**

Cette activité utilise le logiciel MyViz, très souple pour créer des tableaux de bord de pilotage et de visualisation de données.

Après l'avoir téléchargé (<http://www.3sigma.fr/Telechargements-MyViz.html>) et installé, lancez son exécution. Le tableau de bord initialement affiché sera similaire à la capture d'écran ci-dessous :



Ce tableau de bord n'est qu'un exemple de ce qui peut être réalisé avec MyViz. Nous verrons plus loin comment charger celui correspondant à l'expérience que nous souhaitons réaliser dans ce chapitre.

Pour réaliser cette activité, les conditions suivantes doivent être remplies :

- l'ordinateur hôte doit être connecté à la carte pcDuino (voir chapitre 4)
- le système doit être allumé

Charger ensuite le tableau de bord de l'activité dans MyViz. Pour cela, il faut tout d'abord récupérer ce dernier sur votre ordinateur, à partir du lien suivant :

[https://raw.githubusercontent.com/3sigma/CommandeMoteurElectrique/master/MyViz/AsservissementMoteurEnVitesse2\\_Reseau.json](https://raw.githubusercontent.com/3sigma/CommandeMoteurElectrique/master/MyViz/AsservissementMoteurEnVitesse2_Reseau.json)

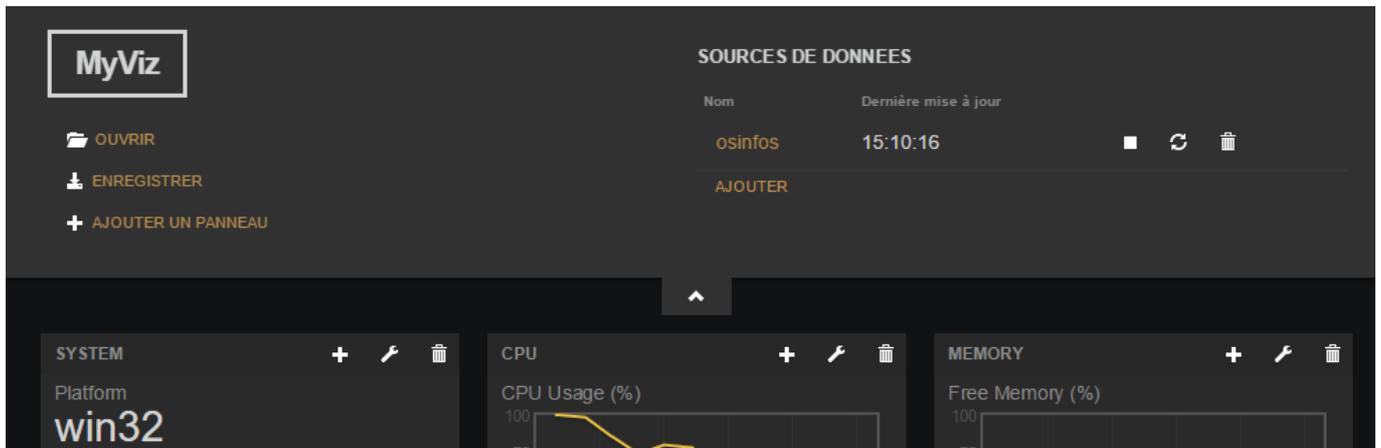
Il se trouve également dans l'archive suivante :

<https://github.com/3sigma/CommandeMoteurElectrique/archive/master.zip>

Pour l'ouvrir dans MyViz, il suffit ensuite de cliquer sur la clé en haut de la fenêtre :

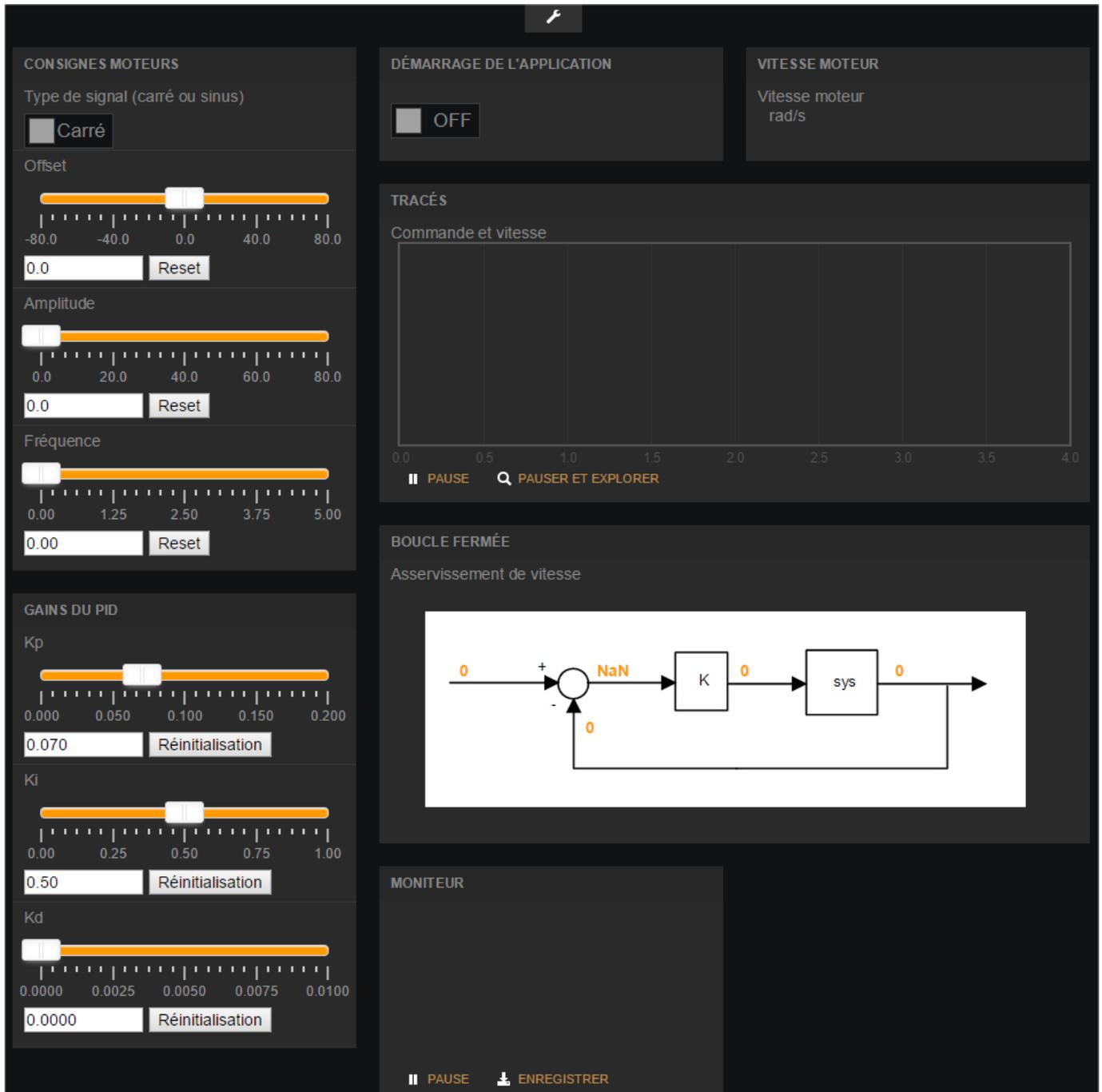


Ceci permet de déplier le panneau supérieur :



Cliquez sur « Ouvrir » et sélectionnez le fichier AsservissementMoteurEnVitesse2\_Reseau.json que vous venez de télécharger.

Le tableau de bord s'affiche alors :



Son utilisation est a priori intuitive. Il faut cependant noter les points suivants :

- le démarrage de l'application se fait via la bouton marche-arrêt en haut au centre.  
**Attention** : il faut attendre quelques secondes avant de voir apparaître les courbes de télémétrie et de pouvoir piloter la vitesse
- la tension de commande envoyée aux moteurs est saturée par programme à  $-/+ 6V$

En fonctionnement, ce tableau de bord peut avoir l'allure suivante :



Il permet de réaliser diverses expériences :

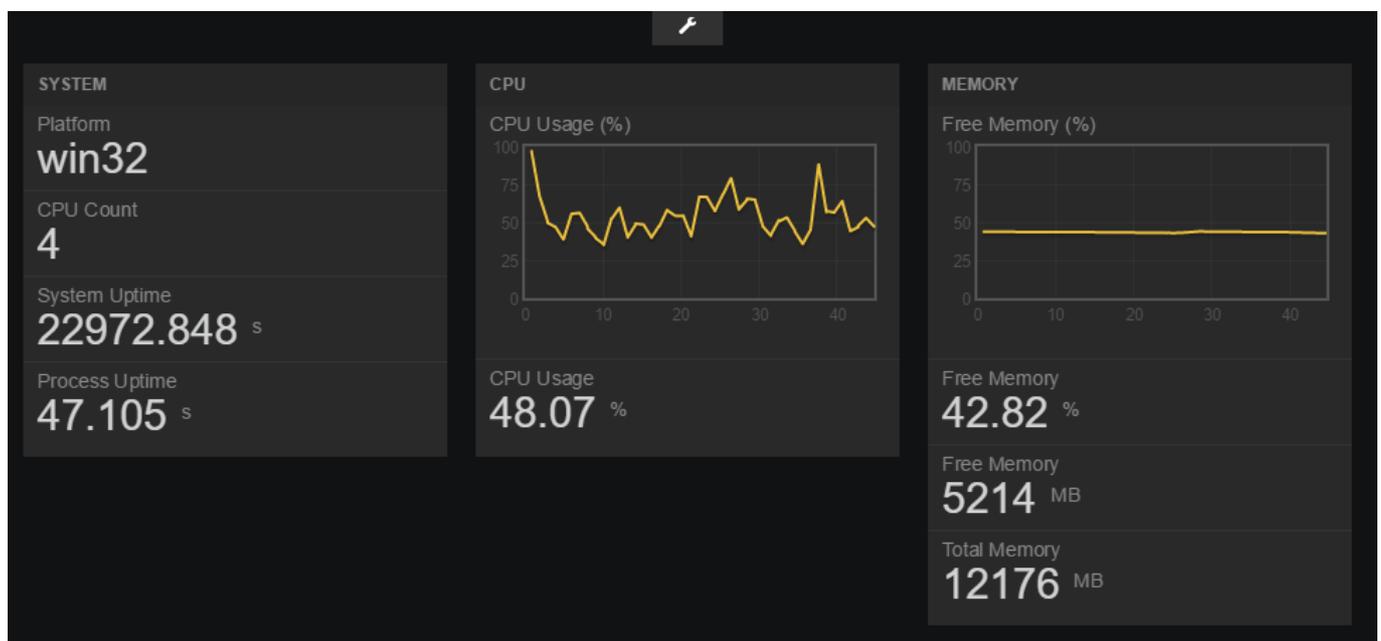
- réponse indicielle
- mesure de l'amplitude et du déphasage de la vitesse du moteur soumis à une consigne de vitesse sinusoïdale de fréquence variable, ce qui permet de tracer un diagramme de Bode du système en boucle fermée
- réglage de l'asservissement pendant le fonctionnement du système, celui-ci étant par exemple soumis à une consigne de vitesse sous la forme d'un signal carré

## 5.2.3 - Application MyViz d'asservissement de position

**ATTENTION : Cette application est compatible uniquement avec les systèmes livrés après le 15 juin 2017.**

Cette activité utilise le logiciel MyViz, très souple pour créer des tableaux de bord de pilotage et de visualisation de données.

Après l'avoir téléchargé (<http://www.3sigma.fr/Telechargements-MyViz.html>) et installé, lancez son exécution. Le tableau de bord initialement affiché sera similaire à la capture d'écran ci-dessous :



Ce tableau de bord n'est qu'un exemple de ce qui peut être réalisé avec MyViz. Nous verrons plus loin comment charger celui correspondant à l'expérience que nous souhaitons réaliser dans ce chapitre.

Pour réaliser cette activité, les conditions suivantes doivent être remplies :

- l'ordinateur hôte doit être connecté à la carte pcDuino (voir chapitre 4)
- le système doit être allumé

Charger ensuite le tableau de bord de l'activité dans MyViz. Pour cela, il faut tout d'abord récupérer ce dernier sur votre ordinateur, à partir du lien suivant :

[https://raw.githubusercontent.com/3sigma/CommandeMoteurElectrique/master/MyViz/AsservissementMoteurEnPosition2\\_Reseau.json](https://raw.githubusercontent.com/3sigma/CommandeMoteurElectrique/master/MyViz/AsservissementMoteurEnPosition2_Reseau.json)

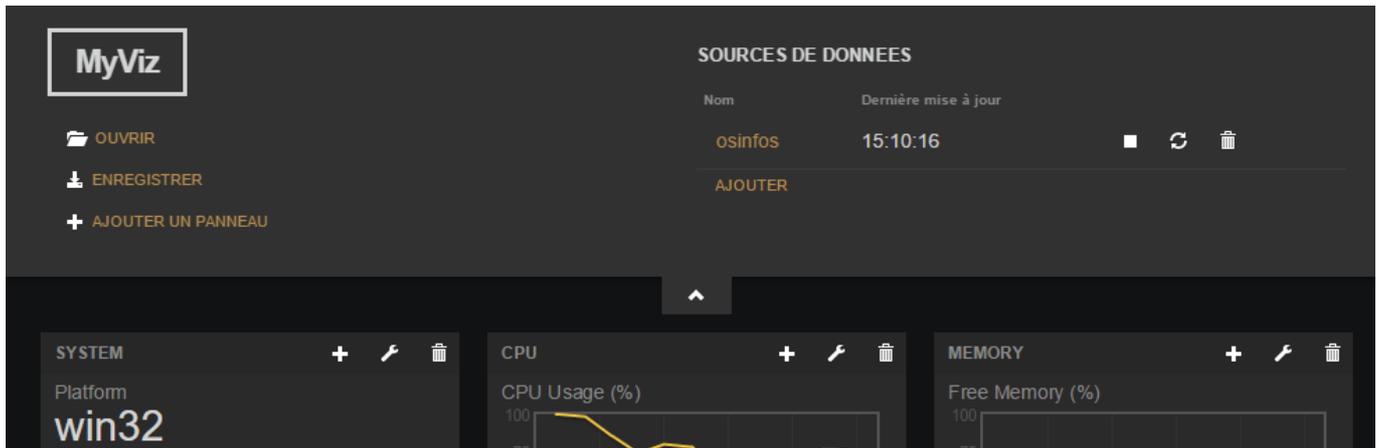
Il se trouve également dans l'archive suivante :

<https://github.com/3sigma/CommandeMoteurElectrique/archive/master.zip>

Pour l'ouvrir dans MyViz, il suffit ensuite de cliquer sur la clé en haut de la fenêtre :



Ceci permet de déplier le panneau supérieur :



Cliquez sur « Ouvrir » et sélectionnez le fichier AsservissementMoteurEnPosition2\_Reseau.json que vous venez de télécharger.

Le tableau de bord s'affiche alors :

The dashboard interface is divided into several functional areas:

- CONSIGNES MOTEURS:** Controls the signal type (Carré), offset (0.00), amplitude (0.00), and frequency (0.00) with sliders and 'Reset' buttons.
- DÉMARRAGE DE L'APPLICATION:** A toggle switch currently set to 'OFF'.
- POSITION MOTEUR:** Displays the current motor position in radians.
- TRACÉS:** A graph area for 'Commande et position' with a 'PAUSE' button and a 'PAUSER ET EXPLORER' button.
- BOUCLE FERMÉE:** A block diagram of a closed-loop control system. The error signal is 'NaN', and the control signal is '0'. The system consists of a gain block 'K' and a system block 'sys'.
- GAINS DU PID:** Controls the PID gains: Kp (1.30), Ki (0.50), and Kd (0.0000) with sliders and 'Réinitialisation' buttons.
- MONITEUR:** A monitoring panel with a 'PAUSE' button and an 'ENREGISTRER' button.

Son utilisation est a priori intuitive. Il faut cependant noter les points suivants :

- le démarrage de l'application se fait via la bouton marche-arrêt en haut au centre.  
**Attention** : il faut attendre quelques secondes avant de voir apparaître les courbes de télémétrie et de pouvoir piloter la position
- la tension de commande envoyée aux moteurs est saturée par programme à  $\pm 6V$

En fonctionnement, ce tableau de bord peut avoir l'allure suivante :



Il permet de réaliser diverses expériences :

- réponse indicielle
- mesure de l'amplitude et du déphasage de la position du moteur soumis à une consigne de position sinusoïdale de fréquence variable, ce qui permet de tracer un diagramme de Bode du système en boucle fermée
- réglage de l'asservissement pendant le fonctionnement du système, celui-ci étant par exemple soumis à une consigne de position sous la forme d'un signal carré

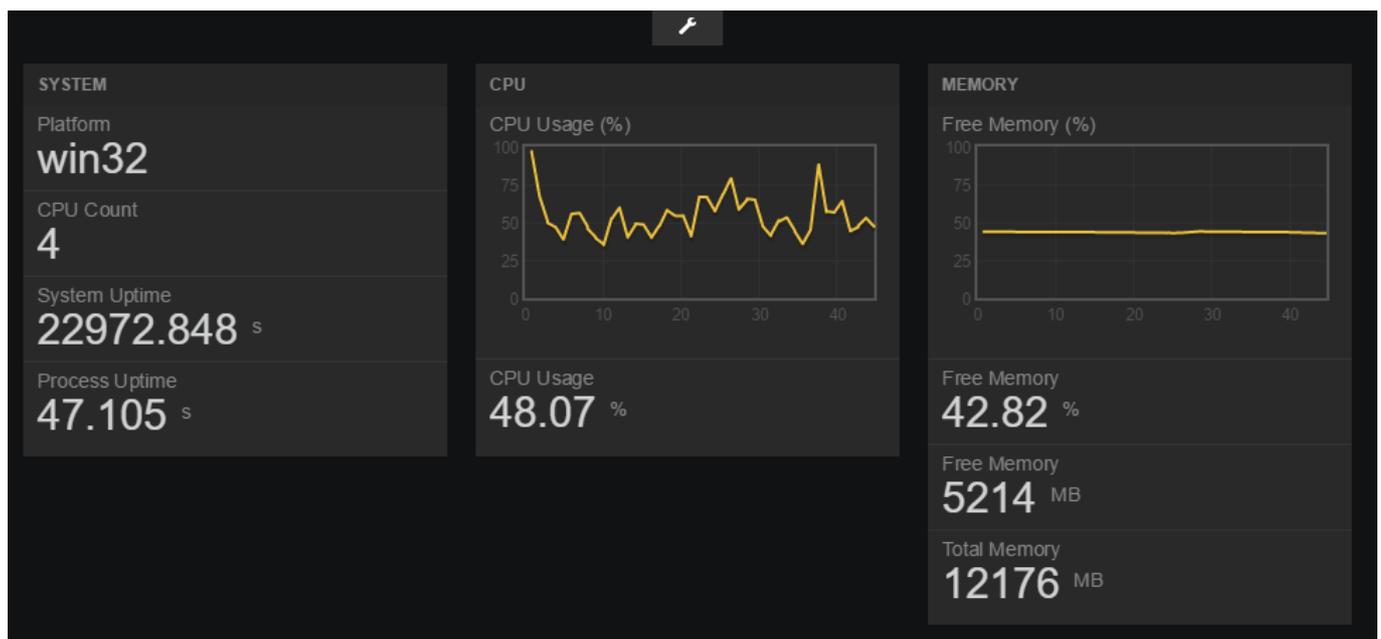
## 5.3 - Applications compatibles avec toutes les versions de matériel

### 5.3.1 - Application MyViz de commande en tension

**ATTENTION : Cette application a été développée pour les systèmes livrés avant le 15 juin 2017. Si votre système est plus récent, vous pouvez avantageusement utiliser l'application MyViz correspondante (voir paragraphe 5.2.1)**

Cette activité utilise le logiciel MyViz, très souple pour créer des tableaux de bord de pilotage et de visualisation de données.

Après l'avoir téléchargé (<http://www.3sigma.fr/Telechargements-MyViz.html>) et installé, lancez son exécution. Le tableau de bord initialement affiché sera similaire à la capture d'écran ci-dessous :



Ce tableau de bord n'est qu'un exemple de ce qui peut être réalisé avec MyViz. Nous verrons plus loin comment charger celui correspondant à l'expérience que nous souhaitons réaliser dans ce chapitre.

Pour réaliser cette activité, les conditions suivantes doivent être remplies :

- l'ordinateur hôte doit être connecté à la carte pcDuino (voir chapitre 4)
- le système doit être allumé

Charger ensuite le tableau de bord de l'activité dans MyViz. Pour cela, il faut tout d'abord récupérer ce dernier sur votre ordinateur, à partir du lien suivant :

[https://raw.githubusercontent.com/3sigma/CommandeMoteurElectrique/master/MyViz/CommandeMoteurEnTension\\_Reseau.json](https://raw.githubusercontent.com/3sigma/CommandeMoteurElectrique/master/MyViz/CommandeMoteurEnTension_Reseau.json)

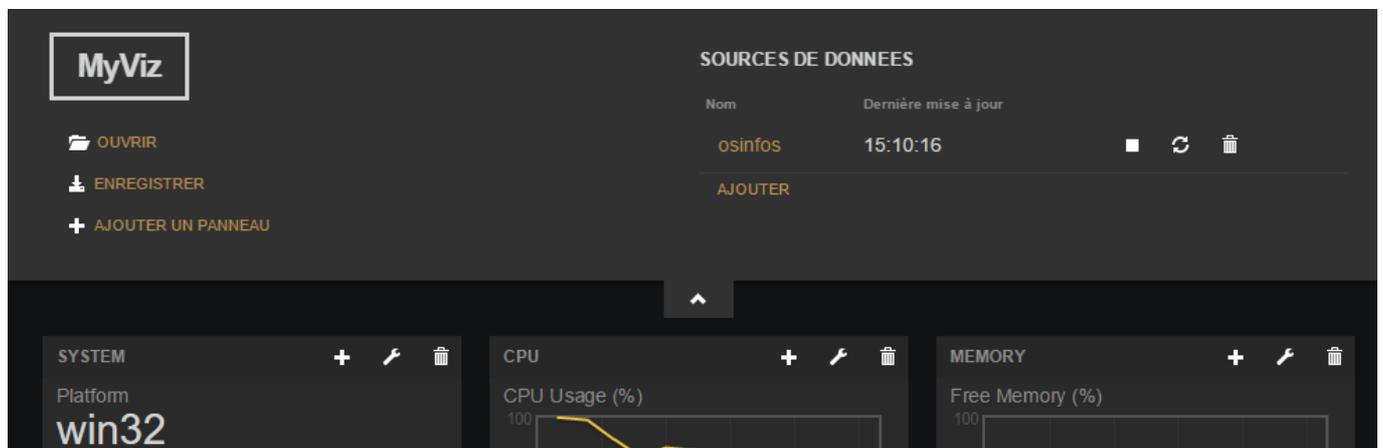
Il se trouve également dans l'archive suivante :

<https://github.com/3sigma/CommandeMoteurElectrique/archive/master.zip>

Pour l'ouvrir dans MyViz, il suffit ensuite de cliquer sur la clé en haut de la fenêtre :

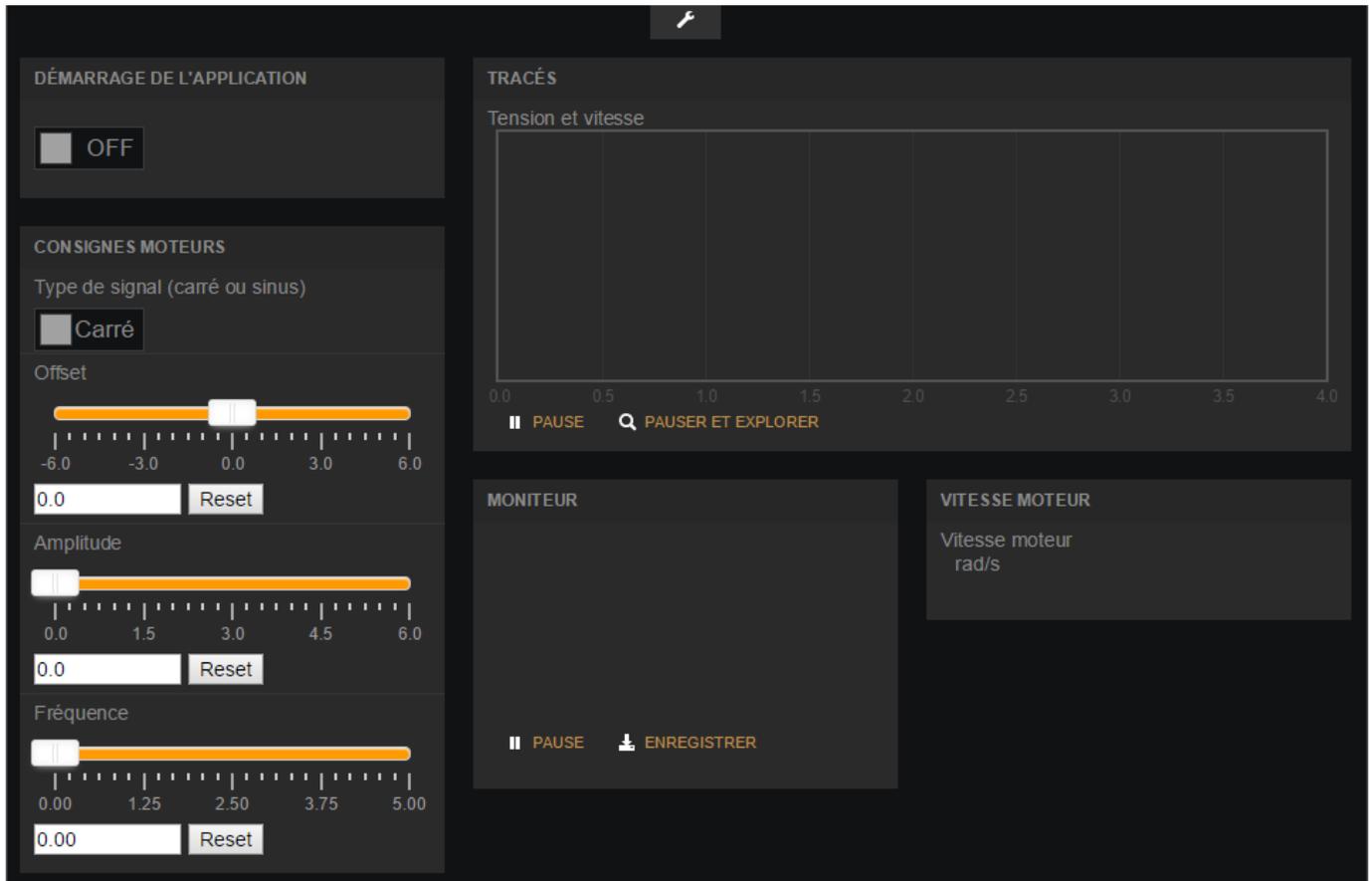


Ceci permet de déplier le panneau supérieur :



Cliquez sur « Ouvrir » et sélectionnez le fichier CommandeMoteurEnTension\_Reseau.json que vous venez de télécharger.

Le tableau de bord s'affiche alors :



Son utilisation est a priori intuitive. Il faut cependant noter les points suivants :

- le démarrage de l'application se fait via la bouton marche-arrêt en haut à gauche.  
**Attention** : il faut attendre quelques secondes avant de voir apparaître les courbes de télémétrie et de pouvoir piloter la tension
- bien que la plage de variation des curseurs soit modifiable (le passage en mode « édition » du tableau de bord se fait en cliquant sur la clé en haut de l'écran), la tension de commande envoyée aux moteurs est saturée par programme à  $-/+ 6V$

En fonctionnement, ce tableau de bord peut avoir l'allure suivante :



Il permet de réaliser diverses expériences :

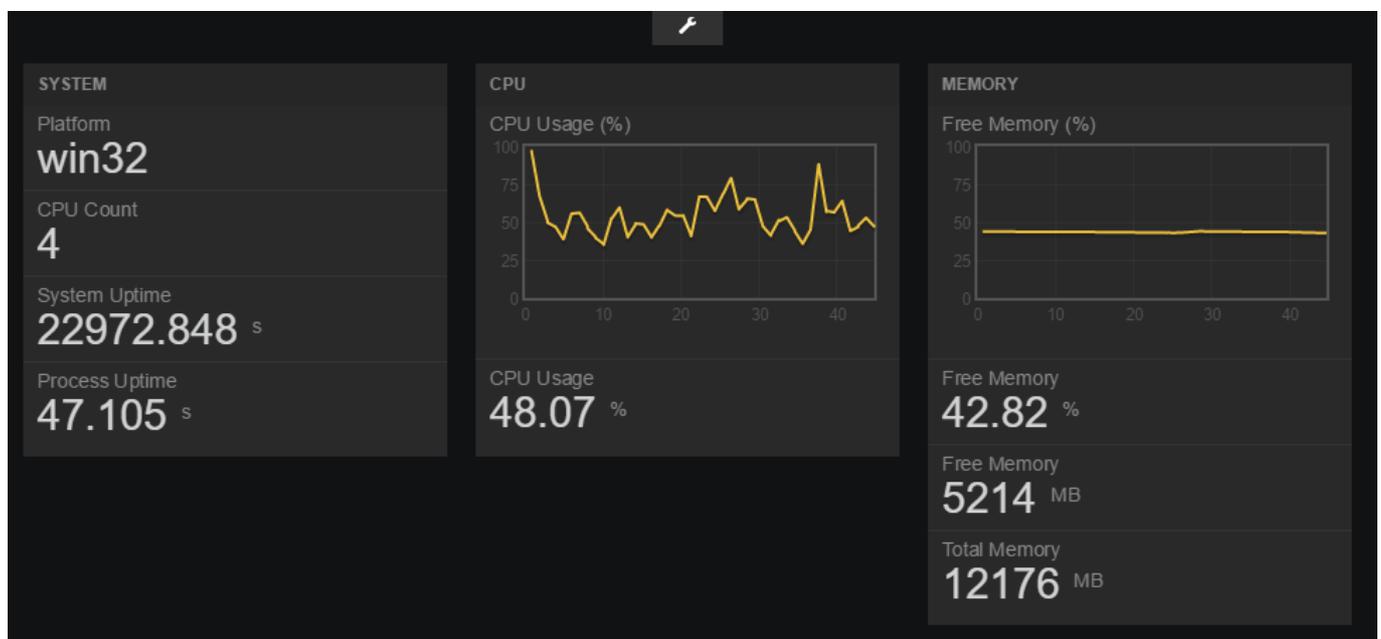
- réponse indicielle
- mesure de l'amplitude et du déphasage de la vitesse du moteur soumis à une tension sinusoïdale de fréquence variable, ce qui permet de tracer un diagramme de Bode du système en boucle ouverte

## 5.3.2 - Application MyViz d'asservissement de vitesse

**ATTENTION : Cette application a été développée pour les systèmes livrés avant le 15 juin 2017. Si votre système est plus récent, vous pouvez avantageusement utiliser l'application MyViz correspondante (voir paragraphe 5.2.2)**

Cette activité utilise le logiciel MyViz, très souple pour créer des tableaux de bord de pilotage et de visualisation de données.

Après l'avoir téléchargé (<http://www.3sigma.fr/Telechargements-MyViz.html>) et installé, lancez son exécution. Le tableau de bord initialement affiché sera similaire à la capture d'écran ci-dessous :



Ce tableau de bord n'est qu'un exemple de ce qui peut être réalisé avec MyViz. Nous verrons plus loin comment charger celui correspondant à l'expérience que nous souhaitons réaliser dans ce chapitre.

Pour réaliser cette activité, les conditions suivantes doivent être remplies :

- l'ordinateur hôte doit être connecté à la carte pcDuino (voir chapitre 4)
- le système doit être allumé

Charger ensuite le tableau de bord de l'activité dans MyViz. Pour cela, il faut tout d'abord récupérer ce dernier sur votre ordinateur, à partir du lien suivant :

[https://raw.githubusercontent.com/3sigma/CommandeMoteurElectrique/master/MyViz/AsservissementMoteurEnVitesse\\_Reseau.json](https://raw.githubusercontent.com/3sigma/CommandeMoteurElectrique/master/MyViz/AsservissementMoteurEnVitesse_Reseau.json)

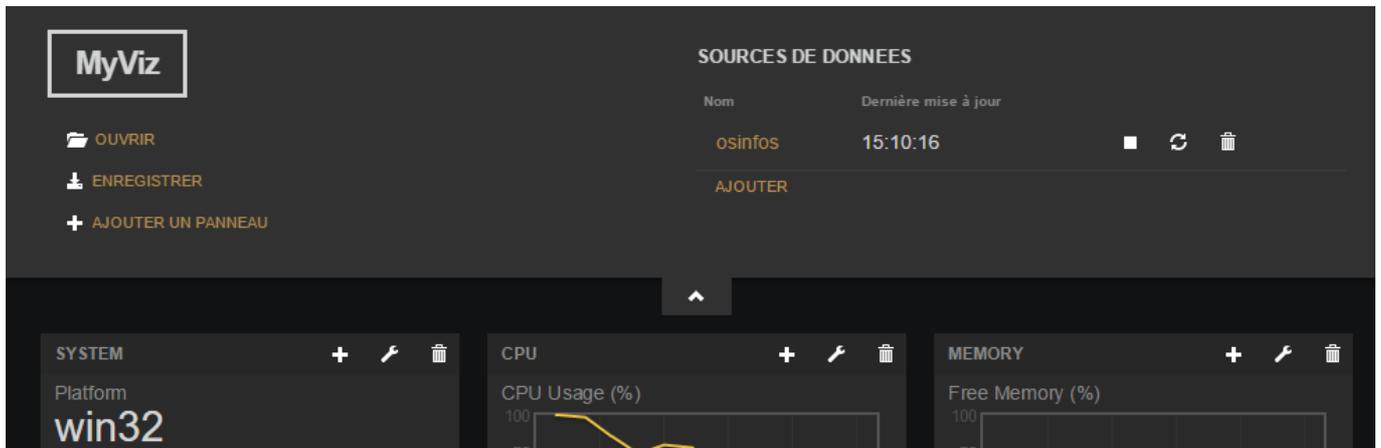
Il se trouve également dans l'archive suivante :

<https://github.com/3sigma/CommandeMoteurElectrique/archive/master.zip>

Pour l'ouvrir dans MyViz, il suffit ensuite de cliquer sur la clé en haut de la fenêtre :

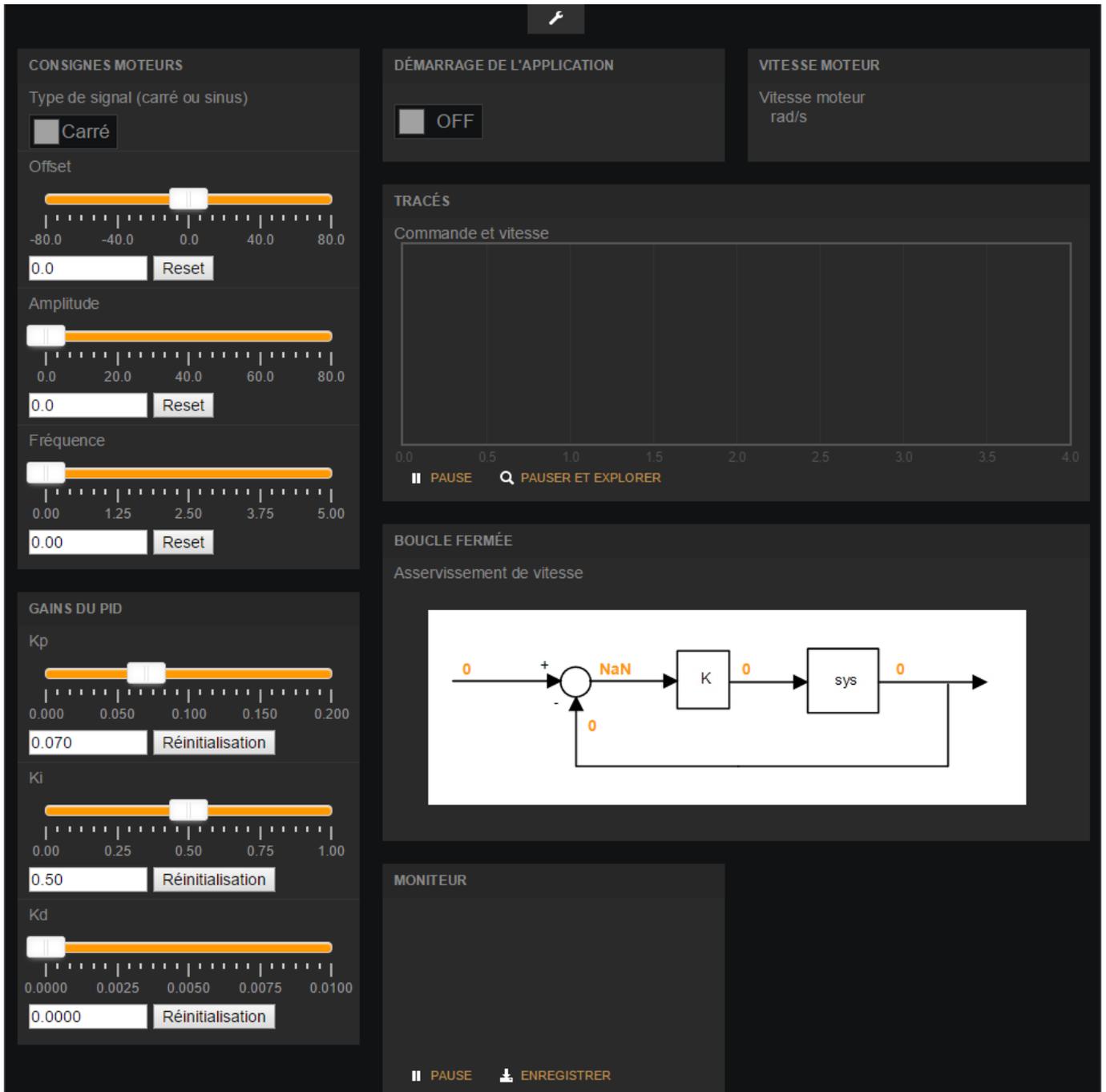


Ceci permet de déplier le panneau supérieur :



Cliquez sur « Ouvrir » et sélectionnez le fichier AsservissementMoteurEnVitesse\_Reseau.json que vous venez de télécharger.

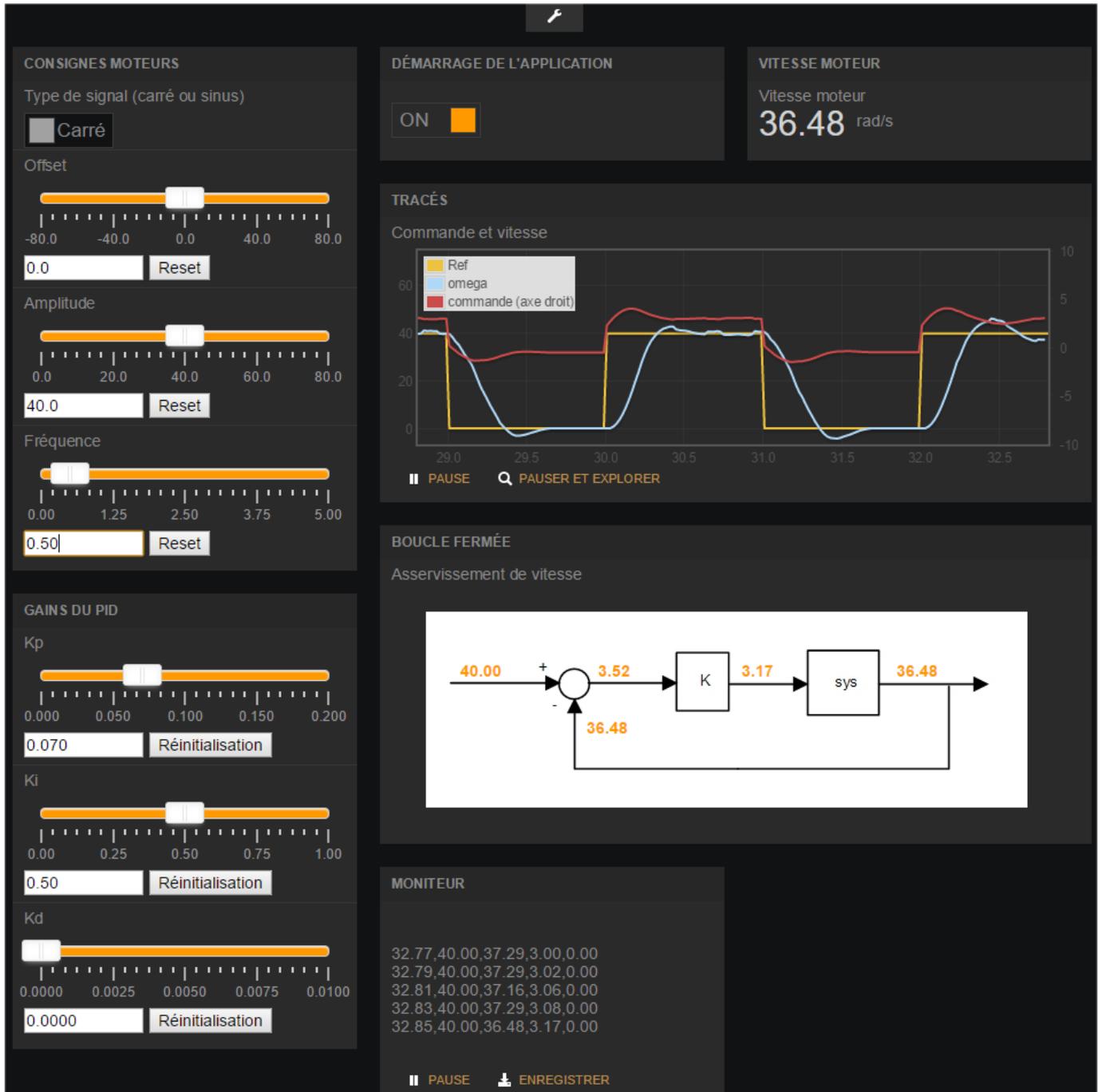
Le tableau de bord s'affiche alors :



Son utilisation est a priori intuitive. Il faut cependant noter les points suivants :

- le démarrage de l'application se fait via la bouton marche-arrêt en haut au centre.  
**Attention** : il faut attendre quelques secondes avant de voir apparaître les courbes de télémétrie et de pouvoir piloter la vitesse
- la tension de commande envoyée aux moteurs est saturée par programme à  $\pm 6V$

En fonctionnement, ce tableau de bord peut avoir l'allure suivante :



Il permet de réaliser diverses expériences :

- réponse indicielle
- mesure de l'amplitude et du déphasage de la vitesse du moteur soumis à une consigne de vitesse sinusoïdale de fréquence variable, ce qui permet de tracer un diagramme de Bode du système en boucle fermée
- réglage de l'asservissement pendant le fonctionnement du système, celui-ci étant par exemple soumis à une consigne de vitesse sous la forme d'un signal carré

### 5.3.3 - Ancienne interface graphique : commande en tension

**ATTENTION : Cette application a été développée pour les systèmes livrés avant le 15 juin 2017. Si votre système est plus récent, vous pouvez avantageusement utiliser l'application MyViz correspondante (voir paragraphe 5.2.1)**

Cette expérience permet de changer la vitesse de rotation du moteur en appliquant une tension variable par l'intermédiaire d'une application qui s'exécute sur votre ordinateur.

Le programme Python gérant cette expérience se nomme `CommandeEnTension.py`. Il est situé dans le répertoire `/root/programmes_python`.

Il comporte de nombreux commentaires permettant de comprendre facilement son fonctionnement.

Le principe de pilotage du moteur consiste à envoyer des signaux PWM sur le shield de commande moteur connecté à la carte pcDuino afin de faire varier la tension d'alimentation du moteur. Cette tension peut être modifiée interactivement via une application qui s'exécute sur votre ordinateur. Elle peut se télécharger à l'adresse suivante:

[http://www.3sigma.fr/Telechargements-Ensemble\\_Commande\\_de\\_moteur\\_a\\_courant\\_continu.html](http://www.3sigma.fr/Telechargements-Ensemble_Commande_de_moteur_a_courant_continu.html)

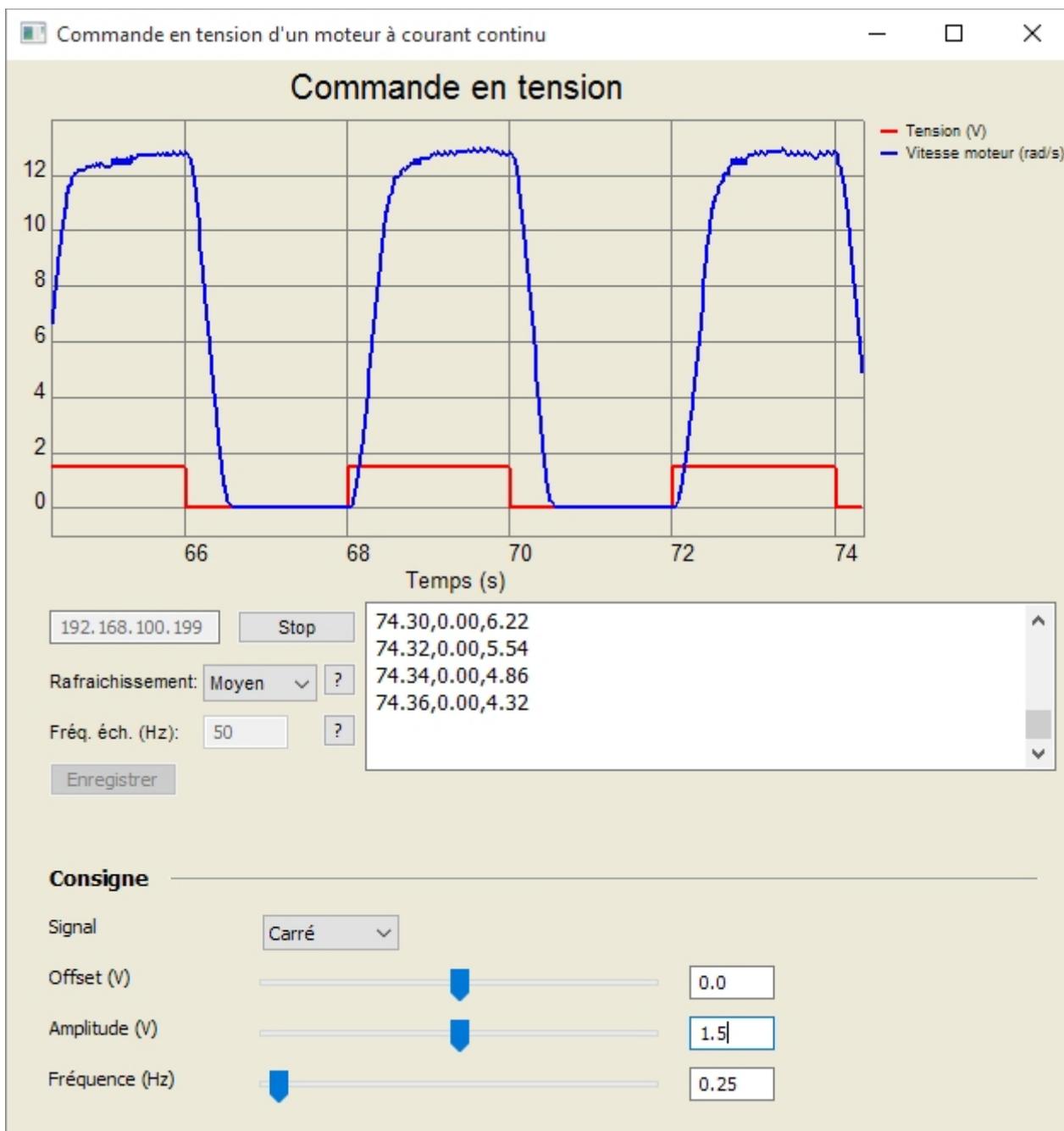
Son nom est de la forme `InterfaceReseauCommandeMoteurEnTension_x.y.zip` (x.y correspond au numéro de version du programme).

Pour l'installer, il suffit de décompresser l'archive dans le répertoire de votre choix. Pour l'exécuter, double-cliquer sur `InterfaceReseauCommandeMoteurEnTension.exe`.

Pour piloter la tension d'alimentation (et donc la vitesse, mais sans asservissement de cette dernière) du moteur depuis votre ordinateur, il vous suffit de suivre les étapes suivantes :

- connecter l'ordinateur hôte à la carte pcDuino (voir chapitre 4)
- allumer le système
- attendre environ une minute le démarrage de la carte pcDuino
- lancer l'application « `InterfaceReseauCommandeMoteurEnTension` »
- entrer l'adresse IP du système (192.168.100.199) dans la zone correspondante et cliquer sur « Connexion ». L'application démarre alors à distance le programme Python de commande du moteur sur la carte pcDuino. Vous pouvez alors interagir avec l'interface graphique pour faire vos expérimentations

Voici une capture d'écran de l'interface de l'application de pilotage:



Les différents éléments sont les suivants (de haut en bas)

- courbes: l'interface permet de visualiser
  - la tension de commande (V, en rouge)
  - la vitesse mesurée (rad/s, en bleu)
- zone de sélection de l'adresse IP: entrer l'adresse IP de votre système et cliquer sur le bouton « Connexion ». Vous verrez alors les courbes se mettre en jour automatiquement et défiler des valeurs numériques dans la zone d'affichage à droite
- « Rafraîchissement »: ceci pilote la fréquence de rafraîchissement des courbes. En fonction de la vitesse de votre ordinateur, vous pouvez choisir parmi les 4 valeurs « Minimum », « Lent », « Moyen » et « Rapide ». Plus la fréquence de rafraîchissement est élevée, moins le tracé est saccadé. Mais si votre ordinateur n'est pas très rapide, vous risquez d'observer un retard entre les consignes et l'affichage. Dans ce cas, il faut choisir un rafraîchissement plus lent
- « Fréq. éch. (Hz) »: ce paramètre correspond à la fréquence d'échantillonnage des mesures dans le programme `CommandeEnTension.py` (20 ms par défaut)
- Bouton « Enregistrer »: il permet d'enregistrer les valeurs numériques affichées dans la zone de « log » vers un fichier texte pour une utilisation ultérieure avec n'importe quel logiciel permettant de lire des données séparées par des virgules dans un fichier texte.  
**Attention** : ce bouton n'est actif qu'après une séquence de mesure. Il est grisé le reste du temps (au démarrage du programme et pendant une séquence de mesure)
- Zone « Consigne »: elle permet de modifier la consigne de tension avec les curseurs. La vitesse du moteur varie alors. Vous pouvez la visualiser, ainsi que la consigne et la tension de commande, sur le graphique qui s'affiche en temps-réel

#### IMPORTANT !

Si vous comparez la vitesse de rotation obtenue sur votre ensemble avec celle affichée sur la capture d'écran au début de ce paragraphe, vous obtiendrez probablement une valeur différente, même si la tension de commande est la même : c'est normal, ceci est dû à la disparité des caractéristiques des moteurs à courant continu.

D'un point de vue pédagogique, ce point permet de souligner la nécessité de réaliser un asservissement de vitesse : si l'on souhaite une vitesse de rotation précise, on ne peut pas se contenter d'une commande en boucle ouverte.

## 5.3.4 - Ancienne interface graphique : asservissement de vitesse

**ATTENTION : Cette application a été développée pour les systèmes livrés avant le 15 juin 2017. Si votre système est plus récent, vous pouvez avantageusement utiliser l'application MyViz correspondante (voir paragraphe 5.2.2)**

Cette expérience permet d'asservir la vitesse de rotation du moteur en appliquant une consigne de vitesse par l'intermédiaire d'une application qui s'exécute sur votre ordinateur.

Le programme Python gérant cette expérience se nomme AsservissementVitesse.py. Il est situé dans le répertoire /root/programmes\_python.

Il comporte de nombreux commentaires permettant de comprendre facilement son fonctionnement.

Le principe de pilotage du moteur consiste à calculer, grâce à un régulateur de type PID, la tension de commande à appliquer au moteur (via commande PWM du shield de commande moteur connecté sur la carte pcDuino) pour qu'il suive la consigne de vitesse spécifiée.

Notez que dans ce programme, la vitesse mesurée est en fait la moyenne glissante des 40 derniers échantillons de mesure de vitesse instantanée, ce qui permet d'avoir une mesure plus lisse. En effet, la résolution de la mesure instantanée est la suivante:

$$2*\pi/(Ts*CPR*ratio)$$

avec :

- Ts : cadence d'échantillonnage (0.01 s)
- CPR : nombre d'impulsions par tour du codeur (48)
- ratio : rapport de réduction du moteur (9.7)

Ceci donne une résolution de  $2*\pi/(0.01*465) = 1.35$  rad/s. Le moyennage permet d'améliorer cette valeur.

La consigne de vitesse peut se fixer interactivement via une application qui s'exécute sur votre ordinateur. Elle peut se télécharger à l'adresse suivante:

[http://www.3sigma.fr/Telechargements-Ensemble\\_Commande\\_de\\_moteur\\_a\\_courant\\_continu.html](http://www.3sigma.fr/Telechargements-Ensemble_Commande_de_moteur_a_courant_continu.html)

Son nom est de la forme InterfaceReseauAsservissementMoteurEnVitesse\_x.y.zip (x.y correspond au numéro de version du programme).

Pour l'installer, il suffit de décompresser l'archive dans le répertoire de votre choix. Pour l'exécuter, double-cliquer sur InterfaceReseauAsservissementMoteurEnVitesse.exe.

Pour changer la consigne de vitesse du moteur depuis votre ordinateur, il vous suffit de suivre les étapes suivantes (remarque: n'exécutez pas de nouveau celles que vous avez déjà effectuées):

- connecter l'ordinateur hôte à la carte pcDuino (voir chapitre 4)
- allumer le système
- attendre environ une minute le démarrage de la carte pcDuino
- lancer l'application « InterfaceReseauAsservissementMoteurEnVitesse »
- entrer l'adresse IP du système (192.168.100.199) dans la zone correspondante et cliquer sur « Connexion ». L'application démarre alors à distance le programme Python d'asservissement du moteur sur la carte pcDuino. Vous pouvez alors interagir avec l'interface graphique pour faire vos expérimentations

Voici une capture d'écran de l'interface de l'application de pilotage:



Les différents éléments sont les suivants (de haut en bas)

- courbes: l'interface permet de visualiser
  - la consigne de vitesse (rad/s, en rouge)
  - la vitesse mesurée (rad/s, en bleu)
  - la tension de commande (V, en vert)
- zone de sélection de l'adresse IP: entrer l'adresse IP de votre système et cliquer sur le bouton « Connexion ». Vous verrez alors les courbes se mettre en jour automatiquement et défiler des valeurs numériques dans la zone d'affichage à droite
- « Rafraîchissement »: ceci pilote la fréquence de rafraîchissement des courbes. En fonction de la vitesse de votre ordinateur, vous pouvez choisir parmi les 4 valeurs « Minimum », « Lent », « Moyen » et « Rapide ». Plus la fréquence de rafraîchissement est élevée, moins le tracé est saccadé. Mais si votre ordinateur n'est pas très rapide, vous risquez d'observer un retard entre les consignes et l'affichage. Dans ce cas, il faut choisir un rafraîchissement plus lent
- « Fréq. éch. (Hz) »: ce paramètre correspond à la fréquence d'échantillonnage des mesures dans le programme AsservissementVitesse.py.
- Bouton « Enregistrer »: il permet d'enregistrer les valeurs numériques affichées dans la zone de « log » vers un fichier texte pour une utilisation ultérieure avec n'importe quel logiciel permettant de lire des données séparées par des virgules dans un fichier texte.  
**Attention** : ce bouton n'est actif qu'après une séquence de mesure. Il est grisé le reste du temps (au démarrage du programme et pendant une séquence de mesure)
- Zone « Consigne »: elle permet de modifier la consigne de vitesse avec les curseurs. La vitesse du moteur varie alors. Vous pouvez la visualiser, ainsi que la consigne et la tension de commande, sur le graphique qui s'affiche en temps-réel
- Zone « Asservissement »: elle permet de modifier les gains du régulateur PID pendant le fonctionnement du système. Cela permet de voir l'influence de ces gains sur les performances de l'asservissement et de régler précisément ce dernier.

## 6 - API Python

**Attention : l'API Python n'est disponible que sur les systèmes livrés après le 15 juin 2017.**

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

- un serveur de d'exécution, qui doit au préalable être lancé (voir plus loin) :
  - soit en ligne de commande
  - soit via une application MyViz
- des commandes exécutées sur la carte pcDuino:
  - soit dans un script Python
  - soit interactivement dans l'interpréteur Python

Plus précisément :

- les commandes de l'API permettant de piloter le moteur (par exemple pour lui donner une consigne de vitesse) envoient des données au serveur d'exécution, qui actionne le moteur en conséquence
- les commandes de l'API permettant de lire les valeurs de variables du système reçoivent ces valeurs lorsqu'elles sont envoyées par le serveur d'exécution

## 6.1 - Fonctions de l'API

---

L'API de pilotage du moteur en Python (« Moteur\_API ») se trouve dans le fichier API.py, dans le répertoire « programmes\_python » de la carte pcDuino.

Elle contient les fonctions suivantes :

### **Moteur\_API()**

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

### **Tension(tension, duree=-1)**

- Paramètres :
  - tension : réel ou chaîne de caractères (expression Python valide du type '3 \* math.sin(t)', t étant reconnu comme le temps courant). Tension de commande (V), saturée en interne entre -6 et 6V
  - duree : réel (optionnel). Durée de la consigne 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 tension d'alimentation au moteur pendant une certaine durée.

### **Vitesse(vitesse, duree=-1, Kp = 0.07, Ki = 0.5, Kd = 0)**

- Paramètres :
  - vitesse: réel ou chaîne de caractères (expression Python valide du type '3 \* math.sin(t)', t étant reconnu comme le temps courant). Vitesse de consigne (rad/s), saturée en interne entre -40 et 40 rad/s
  - duree : réel (optionnel). Durée de la consigne en secondes. Valeur par défaut : -1 (la valeur de la vitesse reste constante jusqu'à ce qu'elle soit changée par une nouvelle commande)
  - Kp : réel (optionnel). Gain proportionnel du régulateur PID de l'asservissement de vitesse. Valeur par défaut : 0.07
  - Ki : réel (optionnel). Gain intégral du régulateur PID de l'asservissement de vitesse. Valeur par défaut : 0.5
  - Kd : réel (optionnel). Gain intégral du régulateur PID de l'asservissement de vitesse. Valeur par défaut : 0
- Description : cette fonction donne une consigne de vitesse au moteur pendant une certaine durée.

### **Position(position, duree=-1, Kp = 1.3, Ki = 0.5, Kd = 0)**

- Paramètres :
  - position: réel ou chaîne de caractères (expression Python valide du type ' $3 * \text{math.sin}(t)$ ',  $t$  étant reconnu comme le temps courant). Position de consigne (rad), saturée en interne entre -6 et 6 rad
  - duree : réel (optionnel). Durée de la consigne en secondes. Valeur par défaut : -1 (la valeur de la position reste constante jusqu'à ce qu'elle soit changée par une nouvelle commande)
  - Kp : réel (optionnel). Gain proportionnel du régulateur PID de l'asservissement de position. Valeur par défaut : 1.3
  - Ki : réel (optionnel). Gain intégral du régulateur PID de l'asservissement de position. Valeur par défaut : 0.5
  - Kd : réel (optionnel). Gain intégral du régulateur PID de l'asservissement de position. Valeur par défaut : 0
- Description : cette fonction donne une consigne de position au moteur pendant une certaine durée.

### **LireVariable(variable)**

- Paramètres :
  - variable : chaîne de caractères. Variable à lire, parmi la liste suivante :
    - Temps : temps courant (s)
    - vref: tension de consigne (V) dans le cas d'une commande en tension
    - omegaref: vitesse de consigne (rad/s) dans le cas d'un asservissement en vitesse
    - omega: vitesse de rotation du moteur (rad/s)
    - thetaref: consigne de position (rad) dans le cas d'un asservissement de position
    - theta: angle de rotation du moteur (rad)
    - commande: tension de commande du moteur (V)
    -
- Description : cette fonction permet de lire la valeur d'une variable du système

## 6.2 - Lancement du serveur d'exécution

Cette étape est indispensable car l'API interagit avec un serveur d'exécution, sans lequel rien ne peut fonctionner (voir au début de ce chapitre 6).

Ce serveur peut être exécuté en ligne de commande ou via une application MyViz.

Tout ce qui suit suppose que le système est allumé et que l'ordinateur y est connecté (voir chapitre 4)

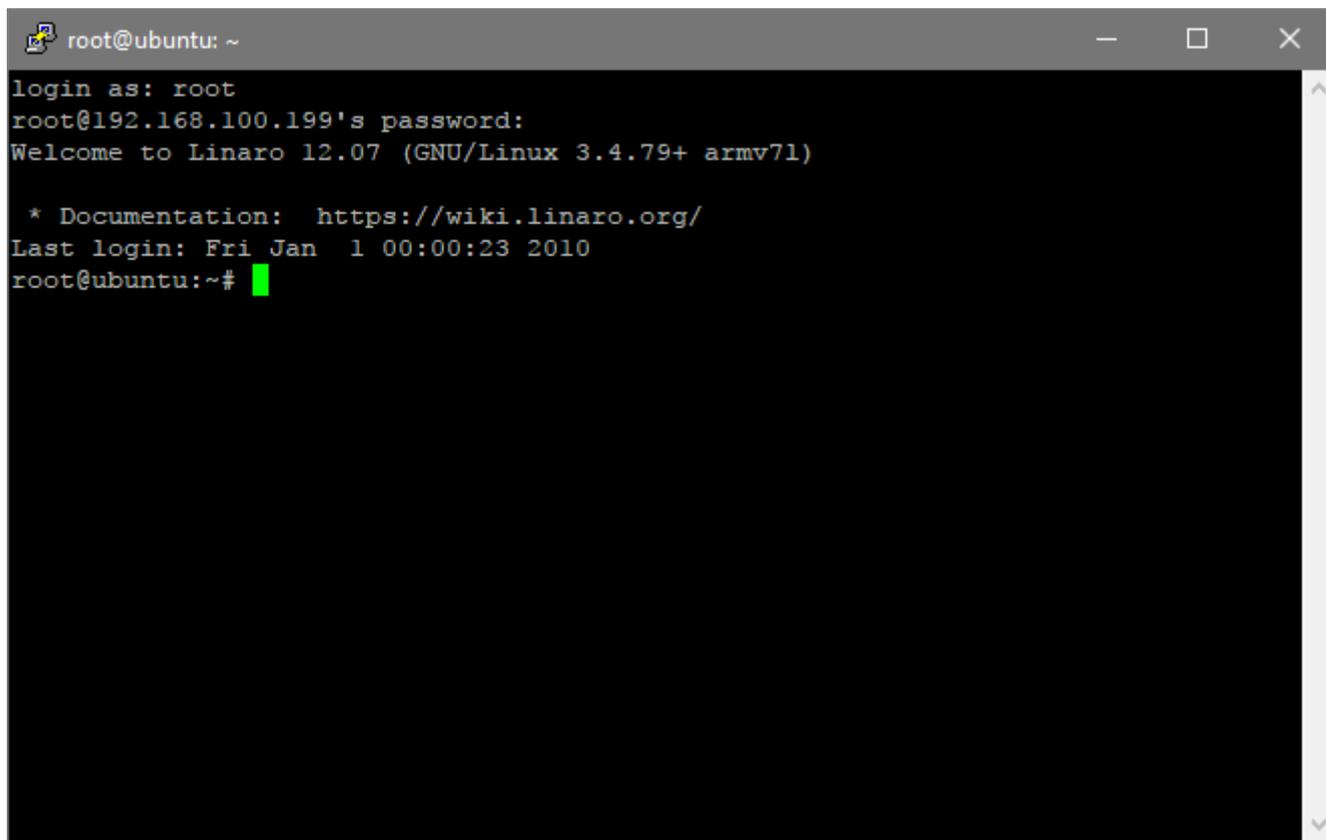
### 6.2.1 - Lancement du serveur d'exécution en ligne de commande

Le lancement du serveur d'exécution en ligne de commande nécessite tout d'abord de se connecter au robot en SSH. Ceci se fait de la façon suivante, en fonction de la plate-forme utilisée :

- Windows : utiliser PuTTY (<https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html>). Après avoir lancé l'exécutable (putty.exe) ouvrir une connexion SSH avec les paramètres suivant :
  - IP : 192.168.100.199
  - port : 22
  - login : root
  - mot de passe : pcdduino
  
- Linux ou macOS :
  - ouvrir un terminal et exécuter la commande suivante :

```
ssh root@192.168.100.199
```
  - entrer le mot de passe : pcdduino

Sur Windows, la console PuTTY est similaire à ceci :

A screenshot of a PuTTY terminal window. The window title is "root@ubuntu: ~". The terminal output shows a login sequence: "login as: root", "root@192.168.100.199's password:", "Welcome to Linaro 12.07 (GNU/Linux 3.4.79+ armv7l)", "\* Documentation: https://wiki.linaro.org/", "Last login: Fri Jan 1 00:00:23 2010", and "root@ubuntu:~#". A green cursor is visible after the prompt.

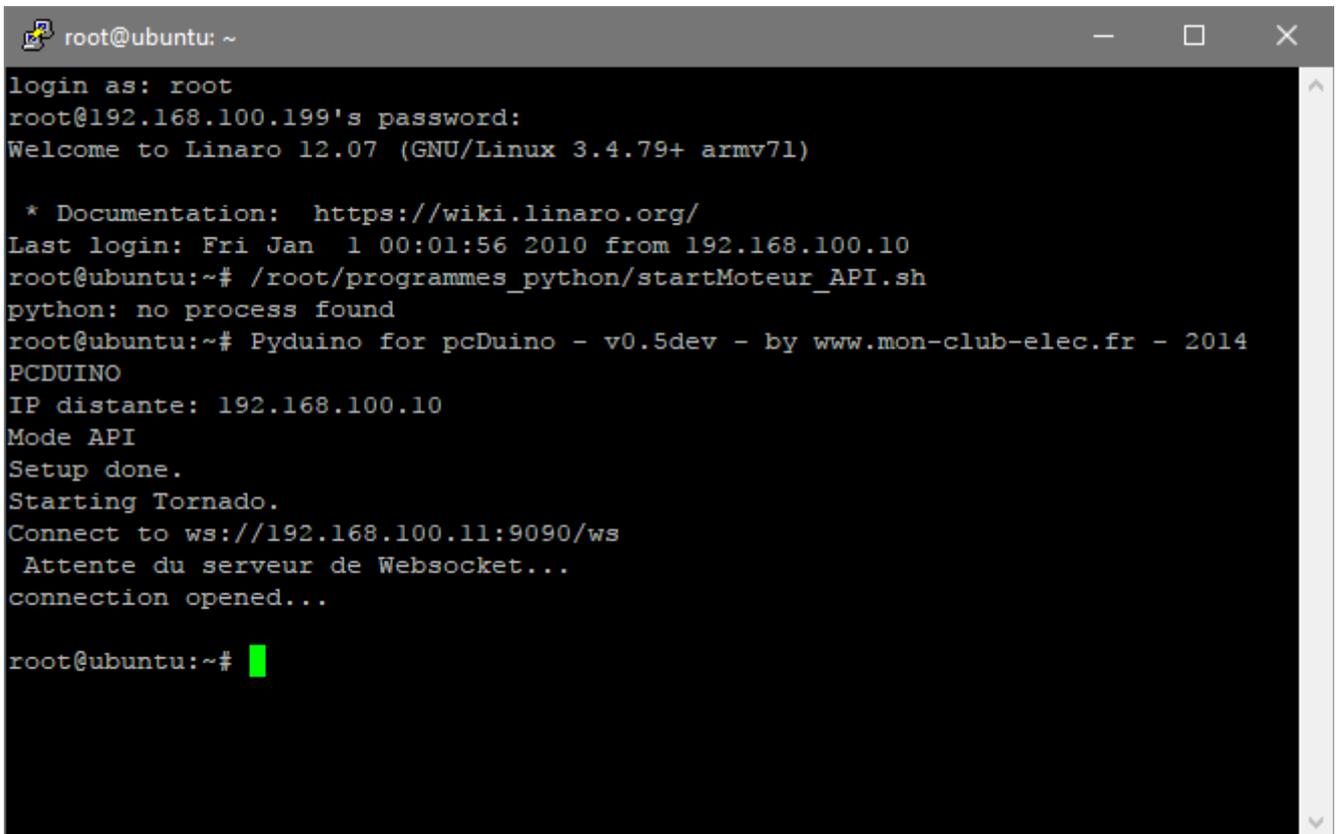
```
root@ubuntu: ~
login as: root
root@192.168.100.199's password:
Welcome to Linaro 12.07 (GNU/Linux 3.4.79+ armv7l)

* Documentation: https://wiki.linaro.org/
Last login: Fri Jan 1 00:00:23 2010
root@ubuntu:~#
```

Le serveur d'exécution se lance alors avec la commande suivante :

```
/root/programmes_python/startMoteur_API.sh
```

Le résultat dans la fenêtre de commande est le suivant :



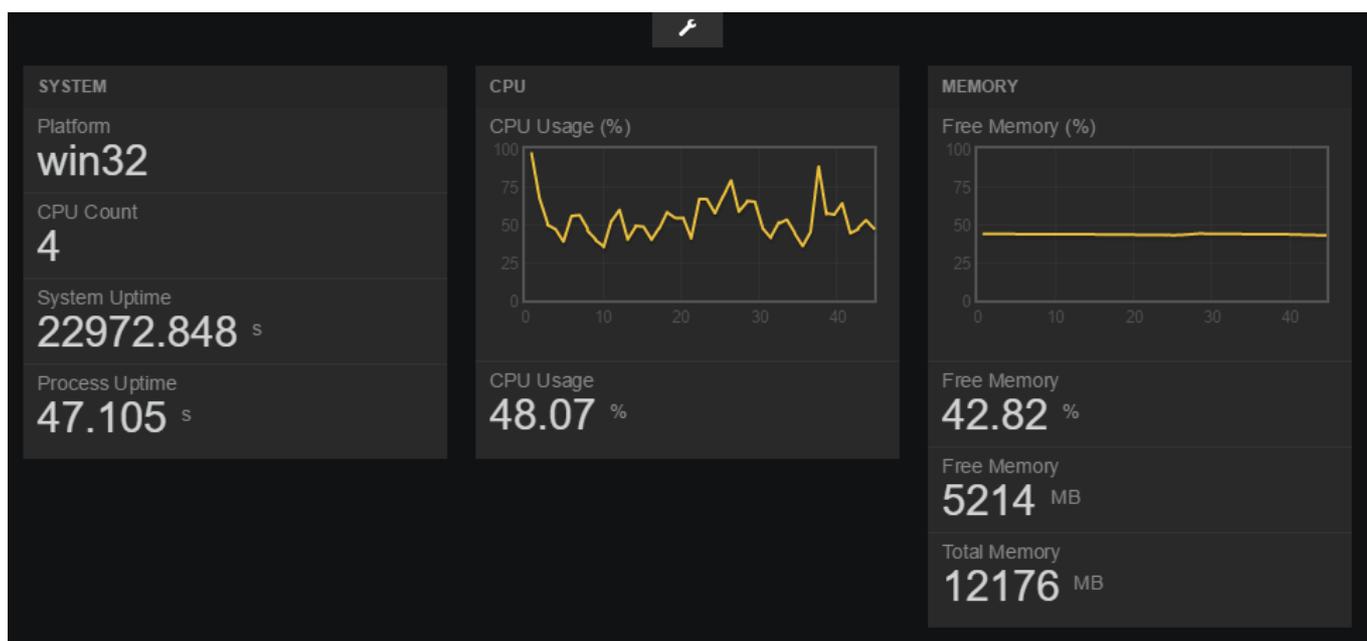
```
root@ubuntu: ~  
login as: root  
root@192.168.100.199's password:  
Welcome to Linaro 12.07 (GNU/Linux 3.4.79+ armv7l)  
  
* Documentation: https://wiki.linaro.org/  
Last login: Fri Jan 1 00:01:56 2010 from 192.168.100.10  
root@ubuntu:~# /root/programmes_python/startMoteur_API.sh  
python: no process found  
root@ubuntu:~# Pyduino for pcDuino - v0.5dev - by www.mon-club-elec.fr - 2014  
PCDUINO  
IP distante: 192.168.100.10  
Mode API  
Setup done.  
Starting Tornado.  
Connect to ws://192.168.100.11:9090/ws  
Attente du serveur de Websocket...  
connection opened...  
  
root@ubuntu:~# █
```

Le serveur d'exécution tourne alors en tâche de fond. Il est maintenant possible d'exécuter des commandes de l'API (voir plus loin).

## 6.2.2 - Lancement du serveur d'exécution via MyViz

L'utilisation de MyViz pour lancer le serveur d'exécution est intéressante car cela permet en parallèle de visualiser en temps-réel les valeurs de certaines variables du système.

Après avoir téléchargé et installé le logiciel (<http://www.3sigma.fr/Telechargements-MyViz.html>), lancez son exécution. Le tableau de bord initialement affiché sera similaire à la capture d'écran ci-dessous :



Ce tableau de bord n'est qu'un exemple de ce qui peut être réalisé avec MyViz.

Charger ensuite le tableau de bord relatif à l'activité « API » dans MyViz. Pour cela, il faut tout d'abord récupérer ce dernier sur votre ordinateur, à partir du lien suivant :

[https://raw.githubusercontent.com/3sigma/CommandeMoteurElectrique/master/MyViz/API2\\_Reseau.json](https://raw.githubusercontent.com/3sigma/CommandeMoteurElectrique/master/MyViz/API2_Reseau.json)

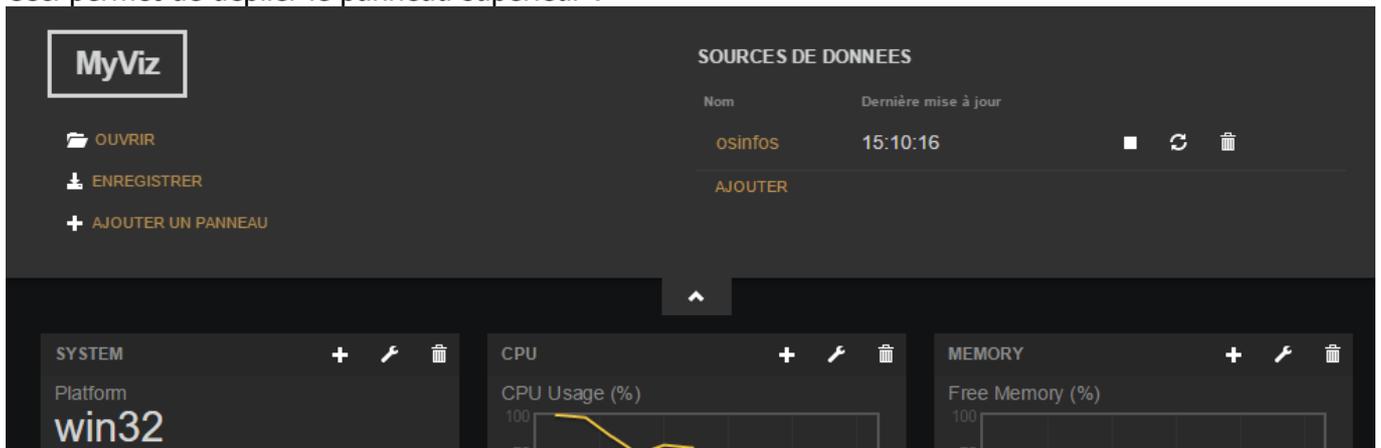
Il se trouve également dans l'archive suivante :

<https://github.com/3sigma/CommandeMoteurElectrique/archive/master.zip>

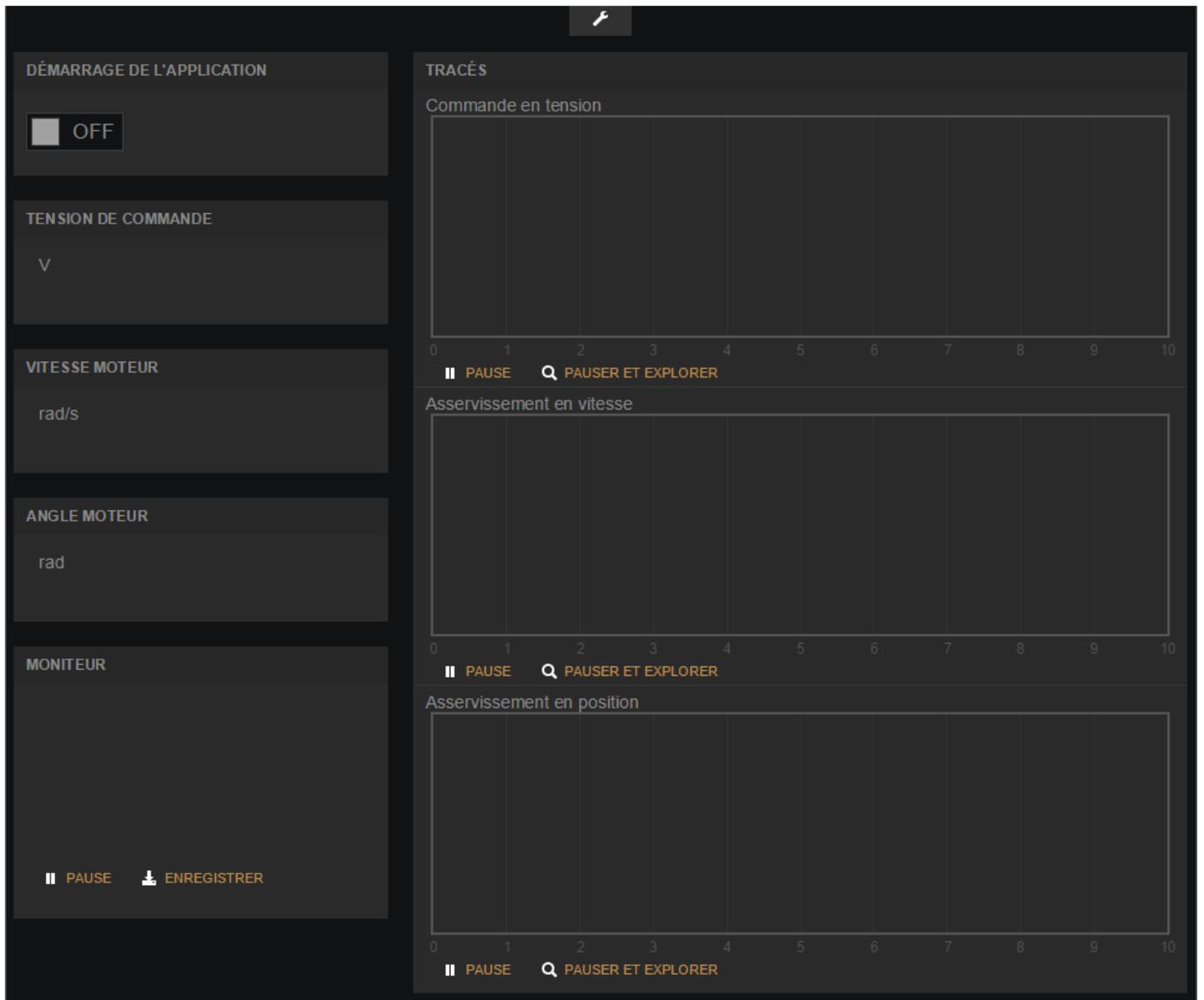
Pour l'ouvrir dans MyViz, il suffit ensuite de cliquer sur la clé en haut de la fenêtre :



Ceci permet de déplier le panneau supérieur :



Cliquez sur « Ouvrir » et sélectionnez le fichier API2\_Reseau.json que vous venez de télécharger. Le tableau de bord s'affiche alors :



Le démarrage du serveur d'exécution se fait via la bouton marche-arrêt en haut à gauche.

**Attention** : il faut attendre quelques secondes avant de voir les courbes s'afficher.

En fonctionnement, ce tableau de bord peut avoir l'allure suivante :



## 6.3 - Exemples d'utilisation de l'API

Les exemples présentés ci-dessous doivent être exécutés sur la carte pcDuino:

- soit dans l'interpréteur Python, lancé depuis le répertoire /root/programmes\_python
- soit sous la forme d'un script préalablement envoyé dans le répertoire /root/programmes\_python

Pour échanger des fichiers entre l'ordinateur hôte et le robot, nous recommandons l'utilisation des logiciels suivants

- FileZilla (<https://filezilla-project.org/>): multi-plateforme
- WinSCP (<https://winscp.net/eng/docs/lang:fr>) sur Windows

### 6.3.1 - Consigne de position sinusoïdale

Dans cet exemple, on donne une consigne de position sinusoïdale (d'amplitude 6 rad et de pulsation 1 rad/s) pendant 10 s :

```
from API import Moteur_API
Moteur = Moteur_API()
```

```
Moteur.Position('6*math.sin(t)',10)
```

## 6.3.2 - Consigne de vitesse suivant un signal carré

Dans cet exemple, la consigne de vitesse est un signal carré d'amplitude 40 rad/s et de fréquence 0.25 Hz. Le signal dure 10 s avec les valeurs nominales des gains. Puis, on modifie la valeur du gain intégral de l'asservissement PID pour une nouvelle durée de 10 s :

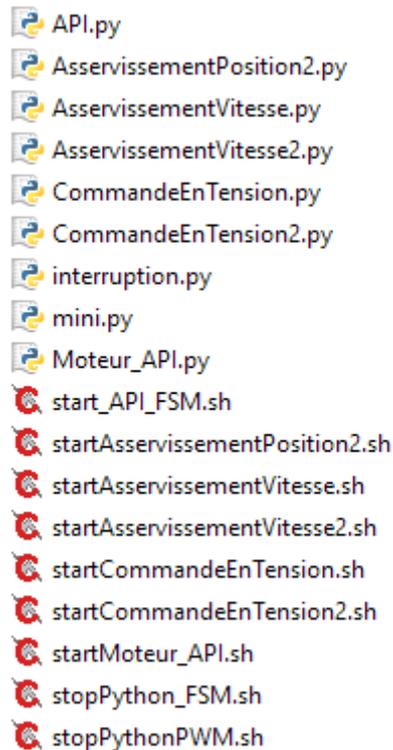
```
import time
from API import Moteur_API
Moteur = Moteur_API()

T0 = time.time()
while time.time() - T0 < 10:
    Moteur.Vitesse(40, 2)
    Moteur.Vitesse(0, 2)

T0 = time.time()
while time.time() - T0 < 10:
    Moteur.Vitesse(40, 2, Kp = 0.07, Ki = 2, Kd = 0)
    Moteur.Vitesse(0, 2, Kp = 0.07, Ki = 2, Kd = 0)
```

## 7 - Accès aux programmes Python

Vous pouvez accéder aux programmes Python de gestion du moteur sur la carte pcDuino dans le répertoire `/root/programmes_python` :



Ces programmes sont librement consultables et modifiables en vous connectant en SSH à la carte pcDuino avec les informations suivantes :

- adresse IP : 192.168.100.199
- login : root
- mot de passe : pcdduino

Nous vous recommandons l'utilisation du logiciel WinSCP (<https://winscp.net/eng/download.php>) qui permet de naviguer très aisément dans l'arborescence des répertoires de la carte pcDuino et de faire facilement des transferts de fichiers entre votre ordinateur hôte et la pcDuino.

## 8 - Important

Cet ensemble est un produit « vivant » en constant développement pour améliorer ou lui ajouter de nouvelles fonctionnalités. Si vous avez des idées ou des besoins pour des développements spécifiques, n'hésitez pas à nous contacter ([info@3sigma.fr](mailto:info@3sigma.fr)).

**Ne restez jamais bloqué sans nous contacter !**

Pour tout problème ou toute requête, contactez-nous à l'adresse [support@3sigma.fr](mailto:support@3sigma.fr).