Kit éducatif mettant en œuvre une technologie d’Intelligence Artificielle

**Interface ordinateur pour le pilotage et la récupération des données**

Ce document présente l’interface de pilotage pour le pilotage et la récupération des données issus du robot

# Introduction

La présente documentation décrit la mise en œuvre du pilotage d’un robot en Bluetooth Low Energy (BLE) par l’intermédiaire d’une interface graphique exécutée sur un PC, en utilisant le pack P-NUCLEO- WB55.

Cette documentation est associée à l’archive logicielle Robot\_WB55\_BLE\_v1.0.zip, contenant toutes les ressources nécessaires au fonctionnement du système.

## Présentation du matériel

Le pack P-NUCLEO-WB55 est constitué :

* D’une carte Nucleo-68 (à base de microcontrôleur STM32WB55RG) intégrée au robot et programmée en C++ via l’environnement de développement Arduino
* D’un dongle (à base de microcontrôleur STM32WB55CG) branché sur le PC de pilotage et programmé en MicroPython

Ces deux microcontrôleurs permettent de communiquer en BLE 5.0.

Le choix du MicroPython pour programmer le dongle a été motivé par le fort intérêt pédagogique de ce langage.

Le principe de fonctionnement est le suivant :

* La carte Nucleo-68 lit les capteurs et commande les actionneurs intégrés au robot. Elle a le rôle « Peripheral » dans la communication BLE. Elle reçoit les commandes de pilotage provenant du dongle et lui renvoie les données mesurées
* Le dongle a le rôle « Central » dans la communication BLE. Il remplace un smartphone, plus traditionnellement utilisé dans ce cas de figure.

Il est branché sur un port USB de l’ordinateur hôte et communique avec ce dernier via un port série virtuel. Il apparaît donc dans le gestionnaire de périphérique du PC dans la catégorie « Ports COM et LPT » mais fonctionne à une vitesse très élevée (2000000 bauds)

* L’ordinateur hôte exécute le logiciel MyViz, interface graphique qui communique avec le dongle sur cette liaison série virtuelle afin d’envoyer les ordres de pilotage (provenant de boutons, curseurs,…) et de récupérer les données mesurées dans le but de les afficher sous la forme de courbes et d’afficheurs numériques

D’un point de vue macroscopique, le dongle agit comme une passerelle remplaçant un câble USB qui pourrait relier le PC au robot. Mais il apporte en réalité un avantage indéniable car, dans la mesure où il est programmable, il permet de faire du pré-traitement et de la mise en forme de données entre le robot et le PC.

Il est par ailleurs important de noter que même si ce projet repose sur une interface graphique (logiciel MyViz) pour interagir avec le robot, le protocole de communication mis en œuvre sur le dongle (voir chapitre 8) pourrait être utilisé par n’importe quel autre type de programme capable de communiquer avec un port série.

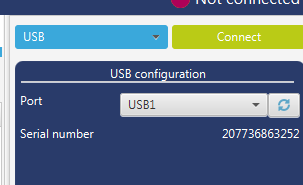
# Installation du firmware MicroPython sur le dongle

Lors de sa réception, le dongle n’est pas prévu pour être programmé en MicroPython. Il faut pour cela installer un firmware spécifique en respectant la procédure suivante :

* Télécharger le dernier firmware MicroPython sur la page suivante : <https://micropython.org/download/NUCLEO_WB55/>

Attention : télécharger le .hex ([https://micropython.org/resources/firmware/NUCLEO\_WB55-](https://micropython.org/resources/firmware/NUCLEO_WB55-20220117-v1.18.hex)  [20220117-v1.18.hex](https://micropython.org/resources/firmware/NUCLEO_WB55-20220117-v1.18.hex) (également fourni dans le répertoire « FirmwareDongle » de l’archive fournie avec cette documentation)

* Télécharger et installer STM32CubeProgrammer (<https://www.st.com/en/development-tools/stm32cubeprog.html>)
* Placer le dongle en mode bootloader (interrupteur vers le connecteur USB), le brancher sur l’ordinateur et lancer STM32CubeProgrammer
* Ce dernier doit reconnaître le dongle. Cliquer alors sur « Connect » :



* Ouvrir ensuite le .hex précédemment téléchargé en cliquant sur le + à droite de « Device memory » et sélectionner « Open file ».
* Cliquer enfin sur « Donwload » pour programmer le firmware sur le dongle

Quand la programmation est finie, débrancher le dongle, mettre l’interrupteur sur l’autre position et le rebrancher sur l’ordinateur. Des LEDs verte et rouge clignotent pendant un certain temps (non négligeable) puis apparaît enfin un nouveau « lecteur » nommé PYBFLASH, contenant les programmes MicroPython qui seront exécutés.

Le programme principal est main.py. Il fait appel à ble\_advertising.py. Ces deux scripts (fournis dans le répertoire « ProgrammesDongle » de l’archive) intègrent a priori suffisamment de commentaires permettant de comprendre leur fonctionnement.

# Modification à apporter à la bibliothèque STM32duinoBLE

La connexion entre le robot et le dongle est impossible sans la modification du fichier

BLEAdvertisingData.cpp situé dans le répertoire « src » de la bibliothèque STM32duinoBLE (traditionnellement située dans le répertoire Documents\Arduino\libraries de l’ordinateur).

Il faut remplacer la ligne 286 suivante :

advField = BLEFieldIncompleteAdvertisedService128;

par :

advField = BLEFieldCompleteAdvertisedService128;

La version corrigée de cette librairie est fournie dans le répertoire « LibrairieArduino » de l’archive

# 5 Programmation du robot

Les deux programmes Arduino susceptibles d’être utilisés pour faire fonctionner le robot sont fournis dans le répertoire « ProgrammesRobot » de l’archive.

Les deux modifications les plus importantes apportées à ces programmes par rapport à leur version originale sont :

* Le transfert des envois de message dans la « loop » au lieu d’une fonction exécutée sous interruption
* Le découpage des messages envoyés en morceaux de taille inférieure à 20 caractères

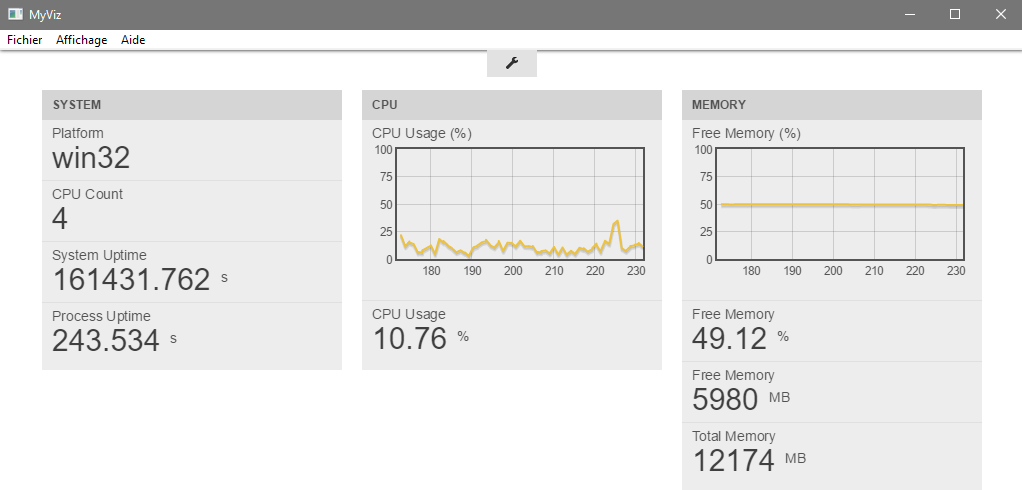
## Installation du logiciel MyViz permettant d’interagir avec le robot

MyViz est le logiciel permettant de piloter et de faire de la télémétrie sur le robot. Il se télécharge depuis la page suivante :

<https://www.3sigma.fr/myviz/MyViz_S2IDIDAC.zip>

Il ne nécessite pas d'installation. Il faut simplement décompresser cette archive dans le répertoire de votre choix (mais de préférence sur C:). Le lancement se fait un double-cliquant sur l'exécutable MyViz\_S2IDIDAC.exe.

Si MyViz n’a jamais été exécuté par l’ordinateur, le tableau de bord initialement affiché sera similaire à la capture d’écran ci-dessous :



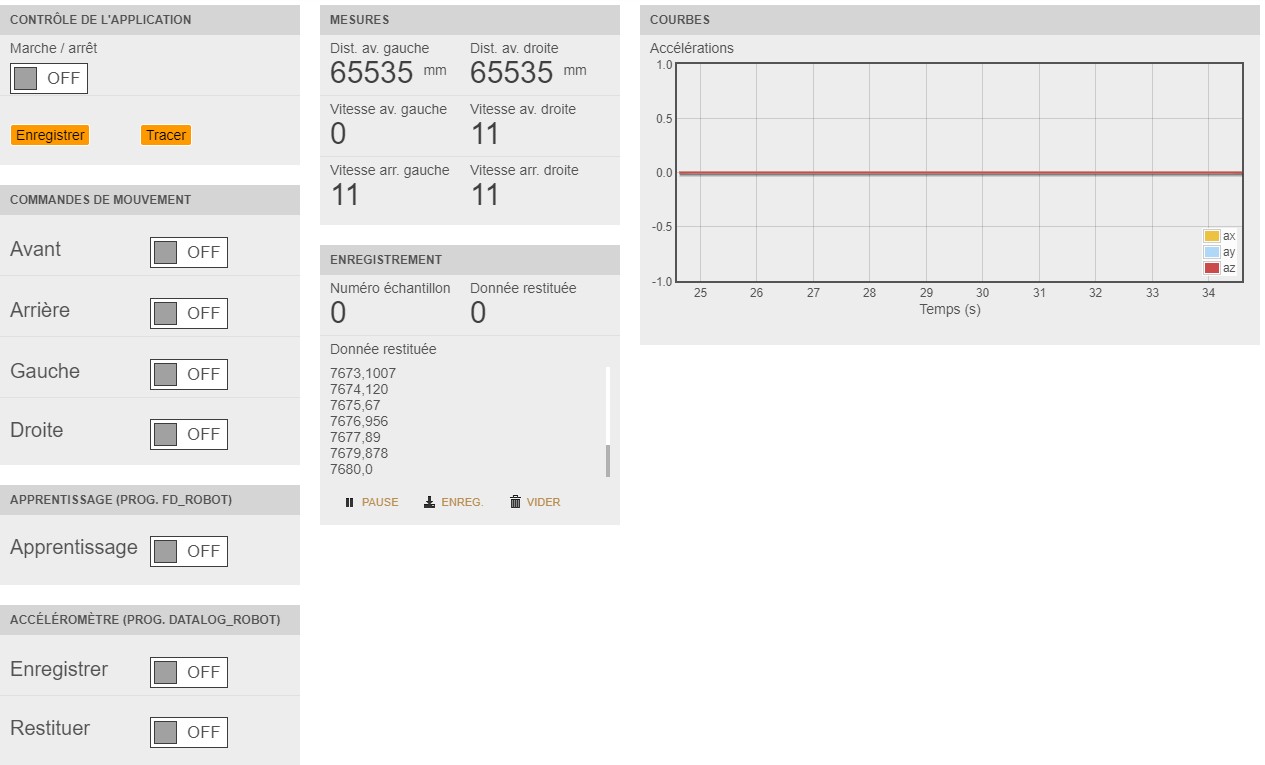
Ce tableau de bord, qui affiche les ressources utilisées de votre système, est un exemple de ce qui peut être réalisé avec MyViz. Bien sûr, ce n’est pas ce que vous souhaitez faire ici !

Si MyViz a déjà été exécuté par l’ordinateur, c’est le dernier tableau de bord ouvert qui s’affiche.

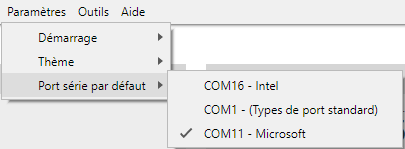
# Mise en œuvre de l'ensemble

La procédure à suivre est simple :

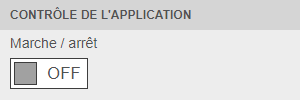
* Mettre le robot sous tension
* Brancher le dongle sur le PC et attendre l’apparition du lecteur PYBFLASH
* Lancer MyViz et ouvrir le tableau de bord PilotageRobotWB55\_1.json (situé dans le répertoire « MyViz » de l’archive fournie avec cette documentation) :



* Sélectionner le port série sur lequel est branché le dongle via le menu Paramètres → Port série par défaut :



* Cliquer sur le bouton Marche / arrêt en haut à gauche pour démarrer l’application



Après quelques instants, les données mesurées s’affichent sur les courbes et les composants numériques

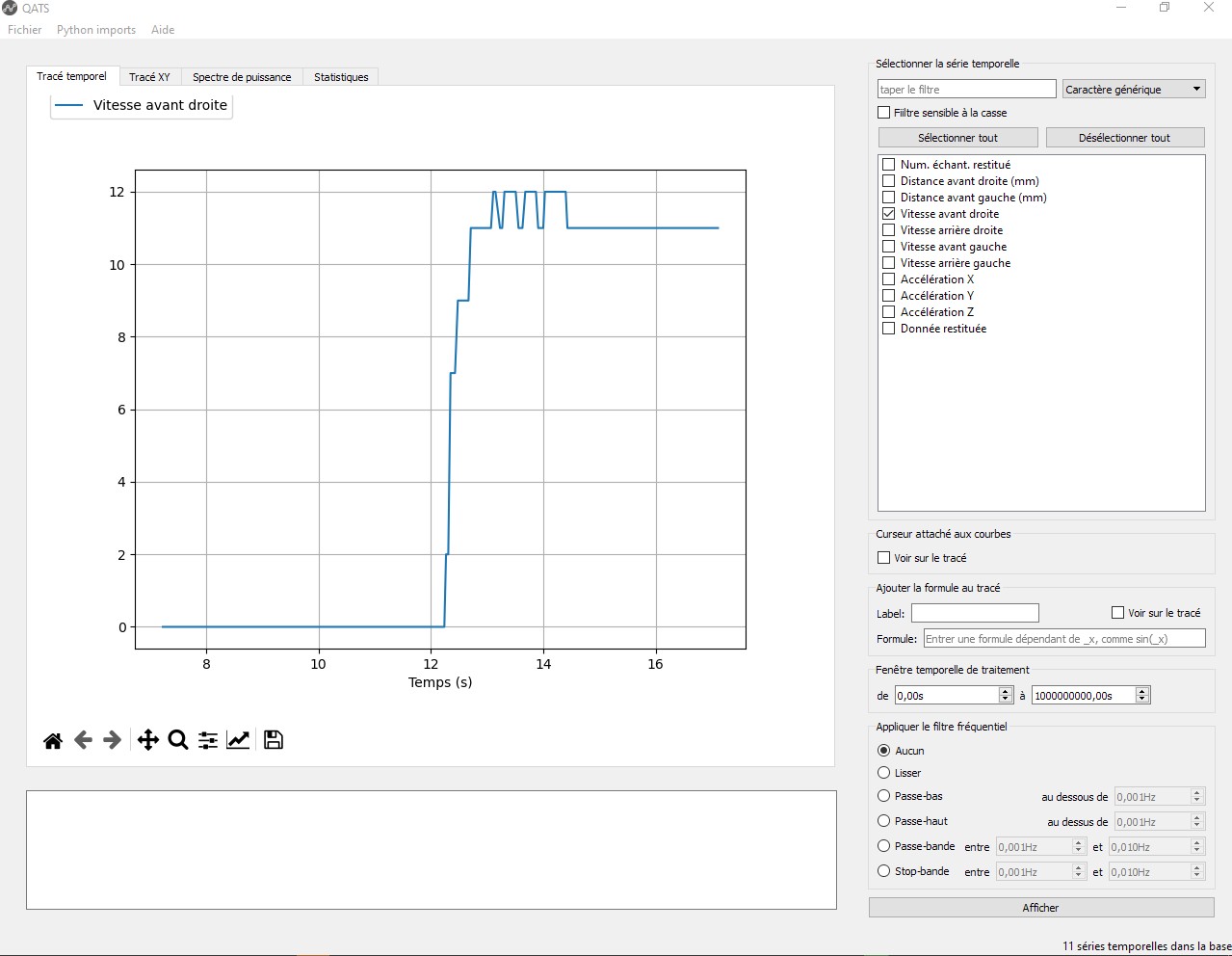
* Il est alors possible de piloter le robot en utilisant les différents boutons prévus à cet effet

L’interface présentée ci-dessus possède les fonctionnalités suivantes :

* Zone « Contrôle de l'application » :
  + Marche-arrêt : permet de se connecter au port série virtuel du dongle (spécifié dans le menu Paramètres → Port série par défaut) et de démarrer l'application exécutée par ce dernier
  + Résultats :
    - Un clic sur le bouton « Enregistrer » ouvre une boite de dialogue permettant d'enregistrer les données mesurées dans un fichier CSV
    - Un clic sur le bouton « Tracer » ouvre l’outil de tracé permettant d’explorer les résultats d’une exécution (voir plus loin)
* Zone « Commande de mouvement » (utilisable avec les deux programmes potentiellement exécutés par le robot) : un clic sur ces boutons mutuellement exclusifs permet d’avancer, de reculer, d’aller à gauche ou à droite
* Zone « Apprentissage » (utilisable uniquement avec le programme FD\_Robot) : lance l’apprentissage. Le bouton revient automatiquement sur « OFF » lorsque l’apprentissage est terminé
* Zone « Accéléromètre » (utilisable uniquement avec le programme Datalog\_Robot) : les deux boutons permettent de lancer l’enregistrement et la restitution. Ils reviennent automatiquement sur « OFF » lorsque l’action est terminée. Pour sauvegarder les données restituées dans un fichier, voir plus loin le descriptif de la zone « Enregistrement »
* Zone « Mesures » : exemple d’affichage numérique de certaines données
* Zone « Courbes » : exemple d’affichage de certaines données sous forme de courbes
* Zone « Enregistrement » : permet d’afficher les données restituées et de les enregistrer. Pour ce dernier point, procéder de la manière suivante :
  + Avant de cliquer sur le bouton « Restituer » dans la zone « Accéléromètre », cliquer sur « Vider » en bas à droite de cette zone d’enregistrement
  + Cliquer maintenant sur le bouton « Restituer »
  + A la fin de la restitution, cliquer sur « Enreg. » en bas de la zone d’enregistrement pour sauvegarder les données restituées dans un fichier CSV. Noter que ce fichier possède 2 colonnes : le numéro de l’enregistrement et la donnée restituée. Ceci permet de vérifier que les données sont complètes. Il est par conséquent nécessaire de supprimer la première colonne (ainsi que l’en-tête) avant de pouvoir exploiter le fichier

Remarque : lors du premier démarrage de l’application, le dongle se connecte au robot si celui-ci est allumé. La LED bleue du dongle s’allume alors. Lors du passage sur OFF, le PC ne reçoit plus de données mais la LED bleue reste allumée car le dongle est toujours connecté.

Enfin, voici une capture d’écran de l’outil d’exploration de tracé mentionné plus haut, lorsqu’on clique sur le bouton « Tracer » :

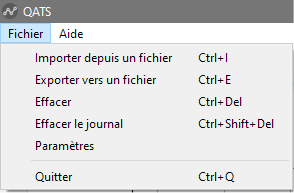


Cet outil possède les fonctionnalités suivantes :

* 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 :



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.

# Protocole de communication entre le dongle et le PC et performances

**Dongle vers PC**

Les données renvoyées par le dongle central à l’ordinateur hôte suivent un protocole de communication simple (csv) :

data1,data2,…,dataN\r\n

Chaque élément est un entier. Les données sont les suivantes :

* + 1. Temps (s)
    2. Numéro d’échantillon restitué
    3. Distance avant droite (mm)
    4. Distance avant gauche (mm)
    5. Vitesse avant droite
    6. Vitesse arrière droite
    7. Vitesse avant gauche
    8. Vitesse arrière gauche
    9. Accélération X
    10. Accélération Y
    11. Accélération Z
    12. Donnée restituée
    13. Indicateur d’apprentissage
    14. Indicateur d’enregistrement
    15. Indicateur de restitution

Le BLE imposant des messages de 20 caractères maximum entre le périphérique (le robot) et le central (le dongle), ce message ne provient pas du robot tel quel, mais il est découpé en plusieurs morceaux par ce dernier. C’est le dongle qui réalise l’aggrégation.

Par conséquent, dans les phases de contrôle de mouvement, pendant lesquelles les mesures des données 3 à 11 sont effectuées, celles-ci sont envoyées en 5 fois. Cet envoi est réalisé dans la partie

« loop » du programme Arduino du robot. On constate un écart d’environ 10 ms entre chaque envoi, ce qui permet d’obtenir l’ensemble de ces mesures à une cadence de 50 ms.

Selon les modes de fonctionnement du robot, les éléments envoyés ne sont pas toujours les mêmes.

Mais le programme MicroPython du dongle est conçu pour toujours envoyer le même message à

l’ordinateur hôte, en remplaçant le cas échéant par des zéros les données qui ne sont pas renvoyées

par le robot dans un mode de fonctionnement donné.

Noter que la communication du robot vers le dongle se fait en mode asynchrone : le mode

« notification » est activé, ce qui permet au dongle d’être immédiatement averti de l’arrivée d’un nouveau message, maximisant ainsi les performances d’acquisition par rapport au mode synchrone, dans lequel le dongle devrait aller lire lui-même les messages potentiellement envoyés par le robot.

**PC vers dongle**

Les données envoyées par l’ordinateur au dongle central à l’ordinateur suivent le même protocole de communication (csv) :

data1,data2,…,dataN\r\n

Chaque élément est égal à 0 ou 1 en fonction de l’état de l’interrupteur. Les données sont les suivantes :

* + 1. Avant
    2. Arrière
    3. Droite
    4. Gauche
    5. Apprentissage
    6. Enregistrement
    7. Restitution

**Remarques et limitations**

Les points suivants mériteraient d’être améliorés dans une seconde phase du projet, dans le cas où la preuve de concept objet de cette première phase serait validée :

* L’interface de pilotage est pour l’instant très basique. On pourrait imaginer notamment un composant interactif permettant de donner des consignes de mouvement de façon plus ergonomique
* Le programme du dongle devrait normalement démarrer automatiquement lors du branchement de ce dernier sur l’ordinateur (c’est le fonctionnement standard des plateformes programmables en MicroPython). Or, ce n’est pas le cas. La solution de contournement adoptée pour l’instant est de démarrer le programme de façon masquée à partir du tableau de bord de MyViz, mais ce n’est pas fiable à 100%
* Le « lecteur » correspondant au dongle apparaît lors du branchement de ce dernier sur l’ordinateur, laissant apparaître ses fichiers de programme, qui pourraient alors facilement être détruits. Cette fois-ci, c’est bien le fonctionnement standard des plateformes sous MicroPython. Il serait intéressant de trouver une méthode permettant d’éviter la mise en évidence de ces fichiers

Par ailleurs, quelques défauts ont été relevés sans être corrigés (car leur origine est hors du cadre du projet) :

* Les vitesses des roues ne sont pas signées
* Sur le robot utilisé pour le développement, la vitesse de la roue avant gauche était nulle, alors qu’elle tournait