

Compte rendu NSI

Description du projet:

Le projet **Wall-E** est la création d'un robot contrôlé à partir d'une manette. On a utilisé un **Raspberry Pi ZERO W** qui fonctionne comme le cerveau du robot, lié à l'ordinateur par lequel on le contrôle avec la manette. Le robot peut bouger ses épaules, bouger ses bras, et enfin fermer les pince, ainsi qu'avancer, reculer et tourner à la manière d'un **tank**.

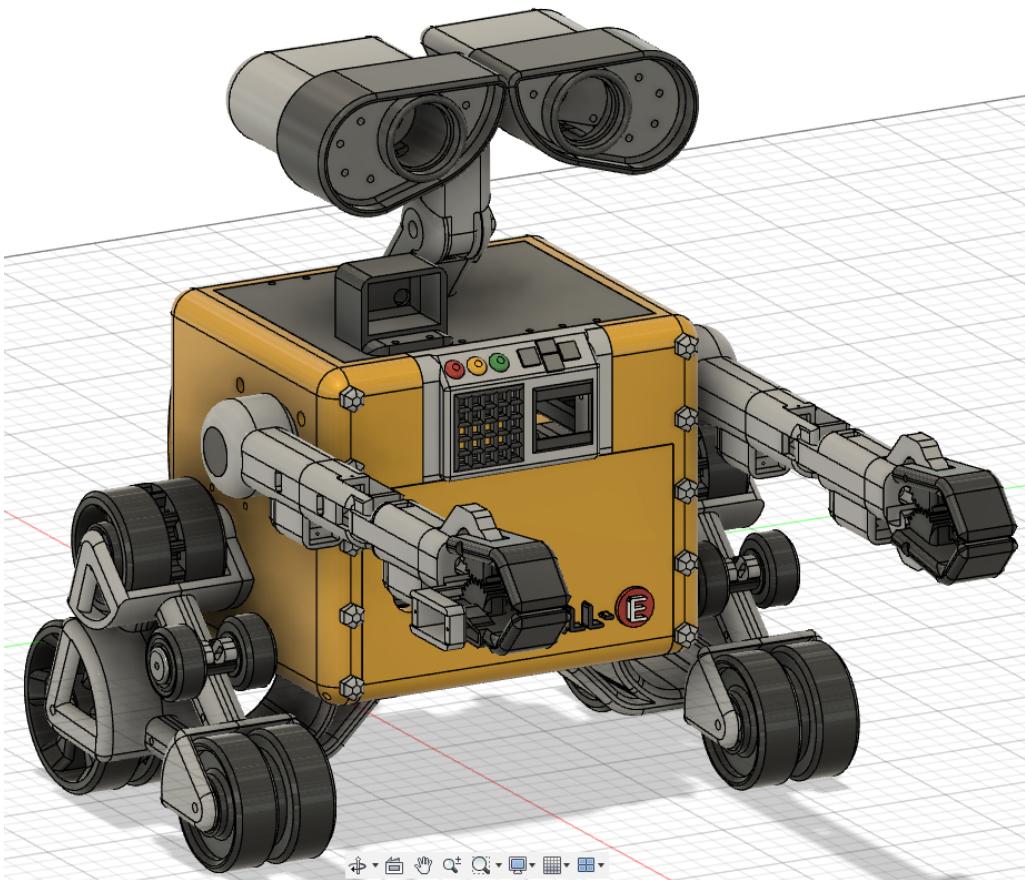
Etat du projet:

Le projet est dans l'étape **finale** au niveau de la programmation des mouvements des servomoteurs, des moteurs DC et de l'envoie **par wifi** des commandes à partir d'un **poste de commande externe**.

Réalisation:

La conception a commencé par la **modélisation 3D** de la coque du robot inspiré du dessin animé Wall-E. Après l'impression et la peinture des différentes pièces, on a installé les différentes parties mécanique (**servomoteur** et moteur réduit 12V) ainsi que le cerveau Raspberry Pi Zero.

Ensuite on a entamé les premiers codes du robot en partant d'une conception simple avec des **fonctions** qui contrôlerait les différentes parties du robot puis on a simplifié en utilisant la **programmation orienté objet**.



Difficultés rencontrées pendant la conception du projet

I. Les servomoteurs:

Les servomoteurs sont des mini moteurs qui permettent de bouger les bras et la tête à un certain angle. Il était facile de les initialiser, mais il fallait bien savoir lesquels actionner dépendant de l'action voulue. On avait des problèmes sur les **variables** des servo-moteurs, ce qui nous causait de les **confondre** et appeler les mauvais servomoteurs, mais aussi de **limiter l'angle** des servomoteurs afin de ne pas les endommager.

II. La Manette:

La manette permet de choisir un servomoteur en particulier, et de l'actionner, ainsi que de faire bouger les moteurs. C'est ce qui permet de contrôler le robot. Nous avons eu le problème de premièrement définir toutes les **actions possibles** et de les attribuer aux boutons de la manette, et ensuite comment utiliser les informations que nous donnent ces valeurs pour actionner le robot. Enfin, le raspberry n'arrivait plus à traiter les informations qu'on lui envoyait assez vite, ce qui causait des **crashs** récurrents à cause du manque de batterie.

III. Le code:

Nous avons divisé le projet en trois fichiers. Le fichier serveur, qui s'exécute sur le raspberry pi. Il reçoit les informations du client grâce au module **socket** de python. Ensuite il

IV. Les moteurs DC et bibliothèque GPIO:

Les moteurs fonctionnent grâce à la **bibliothèque GPIO** qui permet de les **contrôler** et de **changer leur vitesse**. Le problème qu'on a eu est le fait que cette bibliothèque GPIO, lorsqu'on la setup, causait l'arrêt de la connexion internet.

Connection des différents dispositif

I. Démarrage du Raspberry PI

A. Placement des batteries

Le Robot comporte deux ports pour une ou deux batterie de 12V Parkside. Une des batteries alimente le Raspberry et les servomoteurs, l' autre les deux moteurs de 12V. Si les deux batteries sont en place assurez-vous que l'interrupteur sur le port des batteries est sur la position off, comme ceci : ⇒

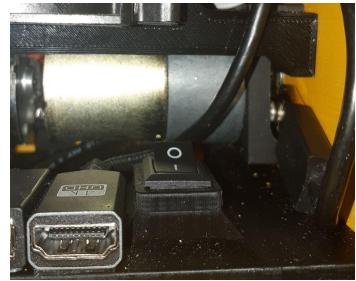


Dans le cas où vous avez qu'une batterie vous pouvez comme même utiliser les moteurs en allumant l'interrupteur au dos du robot, comme ceci : ⇒



B. Démarrage du robot

Le robot contient à l'intérieur de sa coque un interrupteur, afin qu'il se mette en marche, ouvré le capot et allume l'interrupteur.



C. Connection au robot

Le système d'exploitation du Wall-E est RASPBIAN qui est une distribution de Linux. Après que le robot s'est allumé; ce qui prend environ 30 à 50 sec; il va afficher son adresse IP. ⇒

Pour ce connecter au système il y a deux solution:

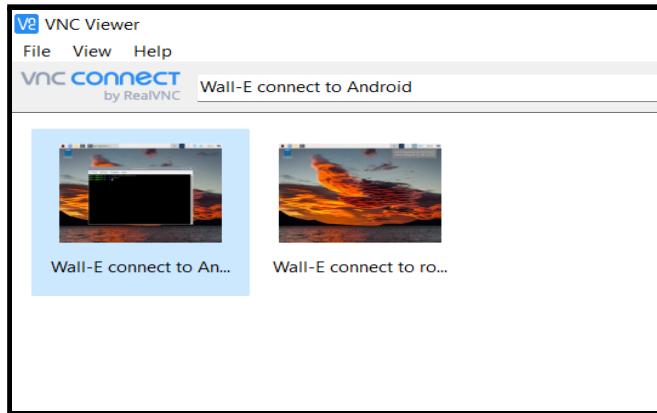


a. Connection par VNC

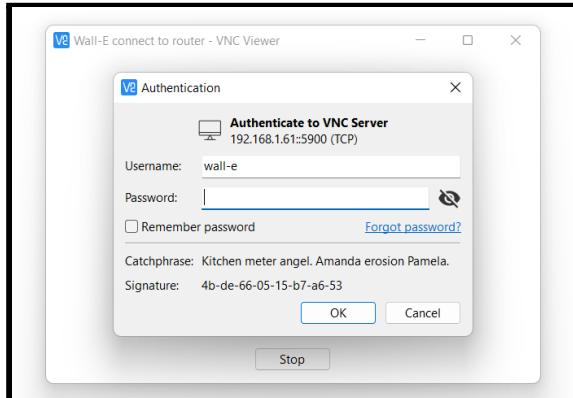
La connexion par VNC permet de voir ce qui se passe sur le bureau du Raspberry, comme si on avait branché un écran, souri et un clavier, mais tout ceci est contrôlé par le biais d'un autre ordinateur (ici le poste de commande).

Pour ce connecter en VNC il faut le programme VNC Real téléchargeable ici :
<https://www.realvnc.com/en/connect/download/viewer/>

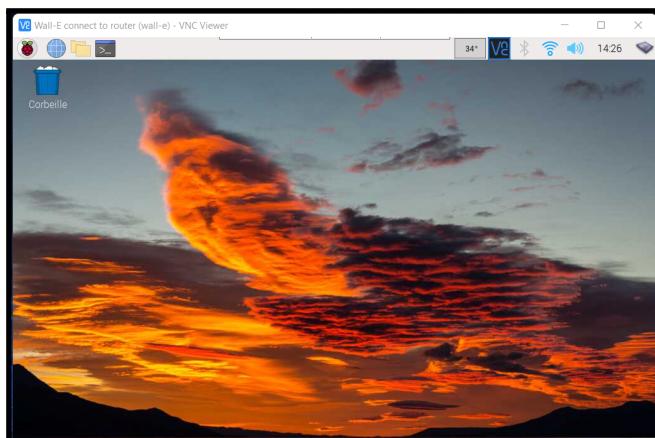
Après installation, il faut mettre dans la barre de recherche l'adresse IP de la machine et tapez ENTER:



Ensuite on demandera les coordonnées de connection, qui sont
username : wall-e
password : wall



Si tous ce passe correctement vous devriez obtenir quelque chose comme ça:



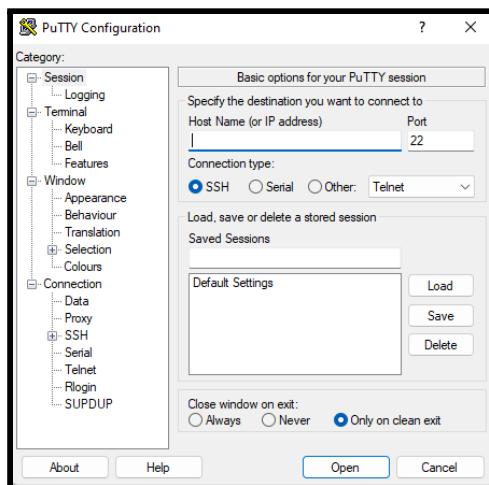
Vous êtes connecté au Wall-e.

b. Connection par SSH et Putty

La connexion par SSH permet une interaction uniquement dans le terminal du Raspberry, autrement dit vous n'aurez pas accès à l' espace du bureau du système.
Pour ce connecter en SSH sur Windows il faut le programme Putty téléchargeable ici :

<https://www.putty.org>

Après installation vous devez indiquer l' adresse IP de la machine dans la barre qui vous le demande :



Apres connection à l' adresse demandée, vous devrez vous connecter avec les informations suivante:

username : wall-e

password : wall



Si tous ce passe correctement vous devriez obtenir ça:

A screenshot of a terminal window showing a successful SSH connection to a Raspbian system. The title bar says 'wall-e@wall-e: ~'. The window displays the following text:

```
wall-e@wall-e: ~
[~] login as: wall-e
[~] wall-e@192.168.1.61's password:
Linux wall-e 5.15.32+ #1538 Thu Mar 31 19:37:58 BST 2022 armv6l

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Wed Apr 27 14:17:37 2022
wall-e@wall-e:~ $ sudo python3 SERVEUR_Rasbian.py
```

The text is in a monospaced font on a black background.

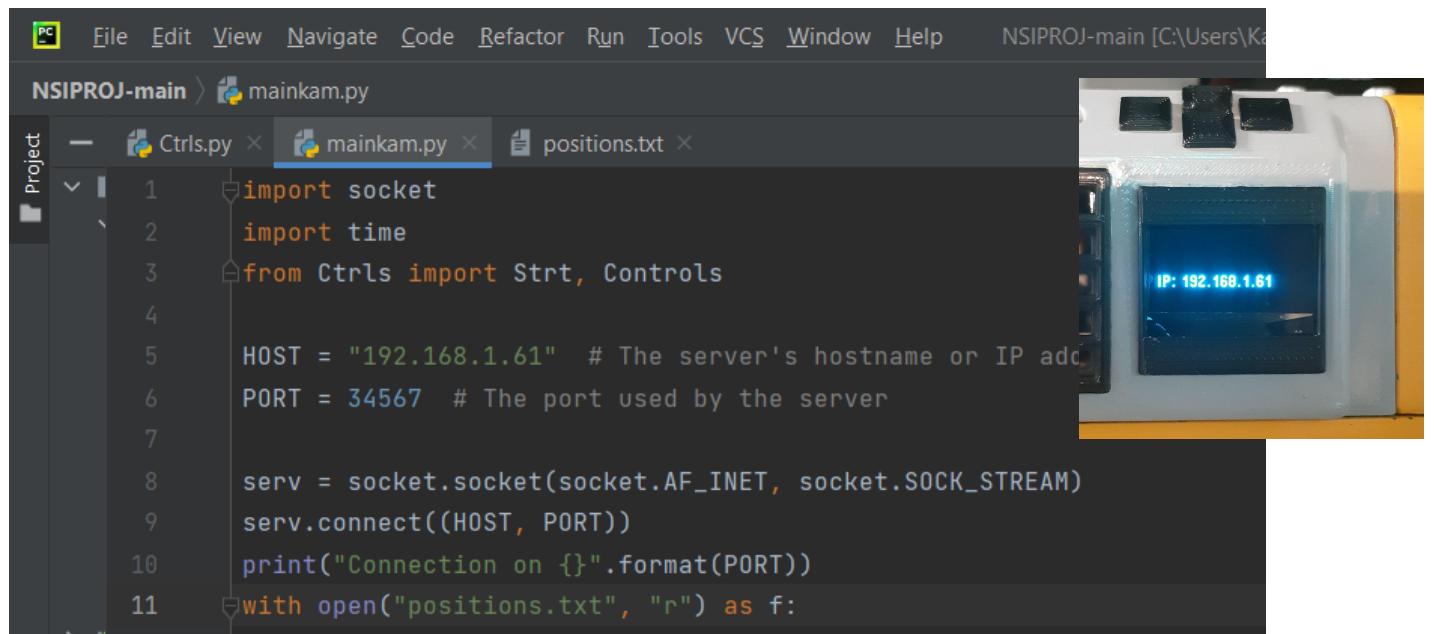
Si c'est le cas vous êtes connecté au Wall-E.

II. Connection au poste de commande

La connexion au Raspberry doit se faire à partir d' un ordinateur externe.

IL EST IMPOSSIBLE DE CONNECTER LA MANETTE DIRECTEMENT AU ROBOT !!

Pour ce connecter il suffit d' ouvrir le code du poste de commande et vérifier si l' adresse IP du robot correspond à celui du code du poste de commande.



```
File Edit View Navigate Code Refactor Run Tools VCS Window Help NSIPROJ-main [C:\Users\Kamal\OneDrive - Université de Lorraine\NSIPROJ-main]
```

NSIPROJ-main > mainkam.py

```
Project Ctrl.py x mainkam.py x positions.txt x
```

```
1 import socket
2 import time
3 from Ctrl import Strt, Controls
4
5 HOST = "192.168.1.61" # The server's hostname or IP address
6 PORT = 34567 # The port used by the server
7
8 serv = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
9 serv.connect((HOST, PORT))
10 print("Connection on {}".format(PORT))
11 with open("positions.txt", "r") as f:
```

Manuel de contrôle

Permet de choisir le servo de 1 à 12 à travers les boutons correspondant..

Contrôle de la tête ou du servomoteur choisi auparavant avec L1

Permet l'exécution du mode de position choisi

Permet le choix de mode enregister

Le maintien simultané de L1,L2,R1,R2 permet la déconnexion de robot du poste de contrôle

L2 et R2: contrôle des moteurs
Maintenir l'un ou l'autre pour tourner ou les deux pour avancer

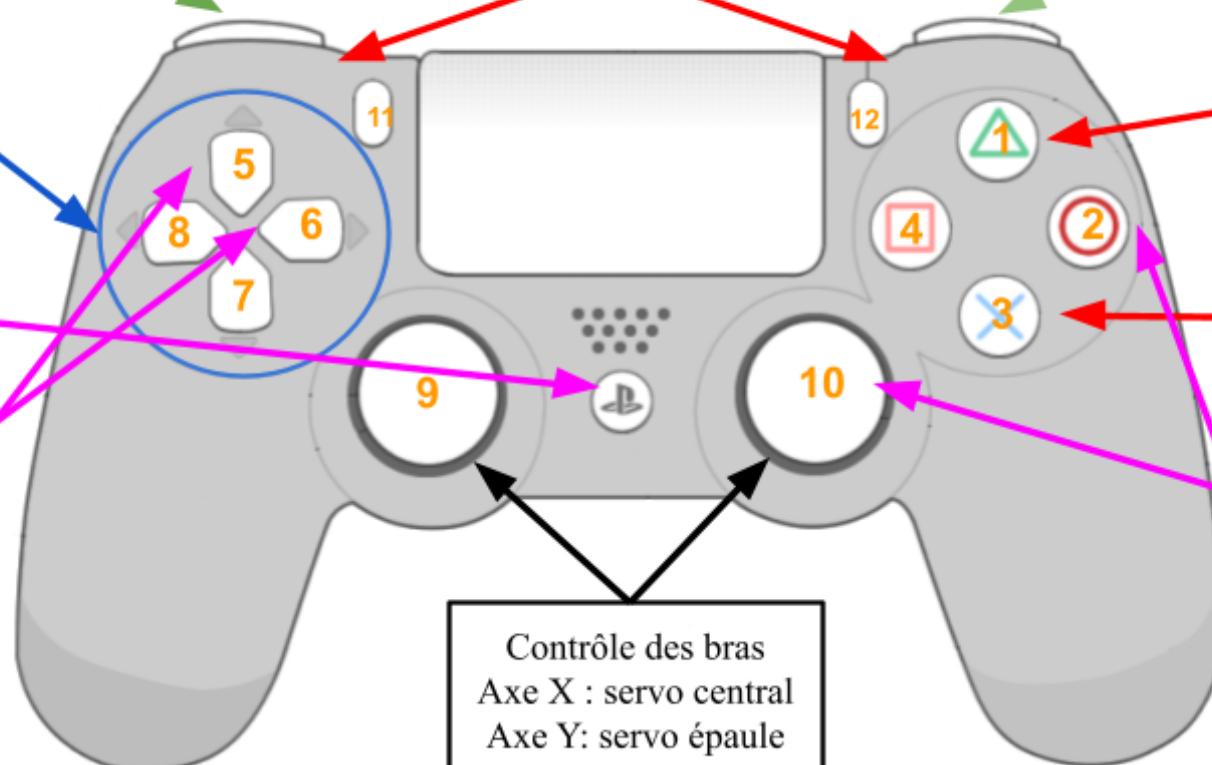
Maintenir le bouton permet d'activer le mode bouger l'unique servo choisi.

Maintenir le bouton pour pouvoir avancer avec L2 et R2

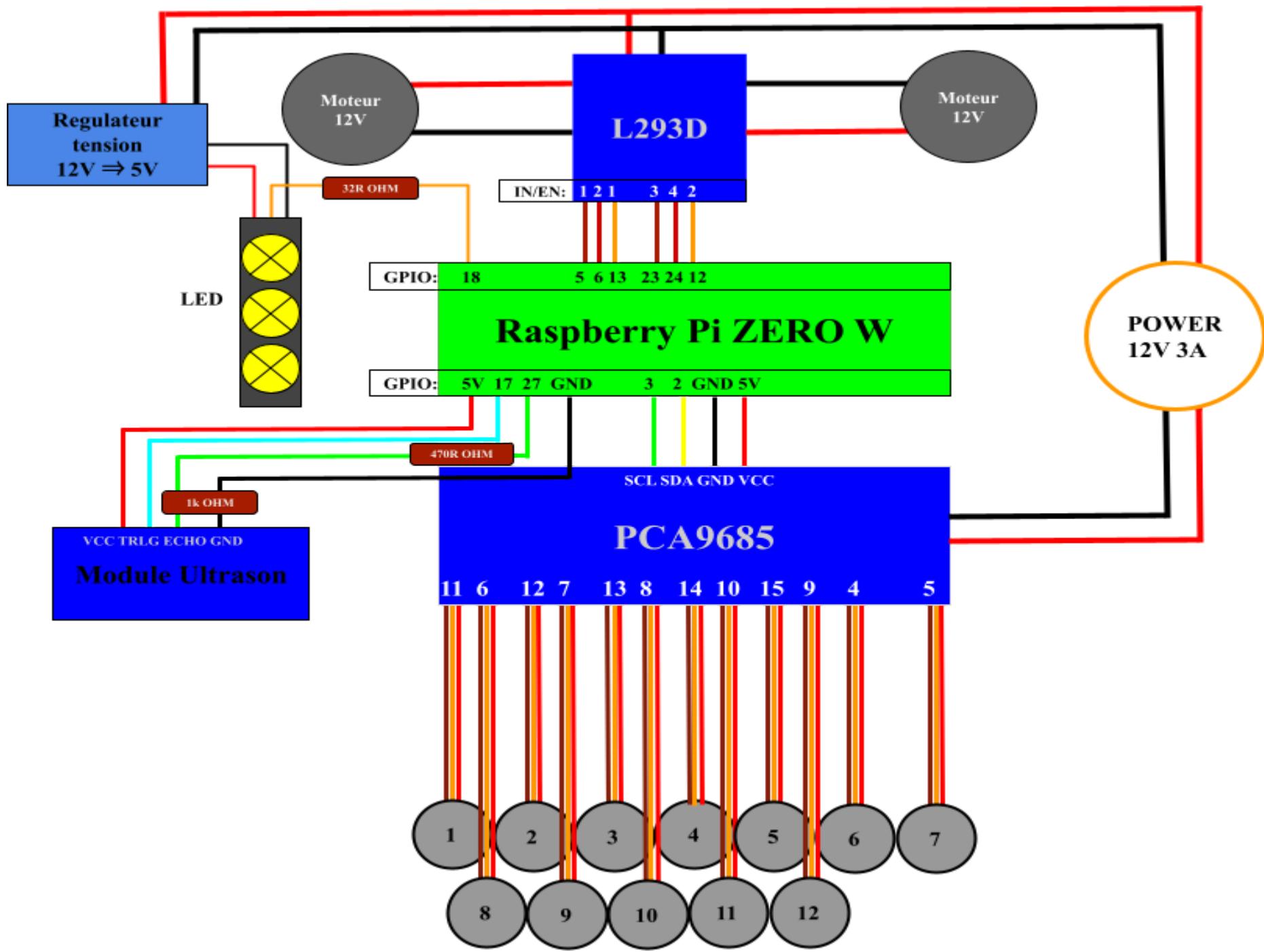
Maintenir le bouton pour pouvoir reculer avec L2 et R2

Maintenir pour entrer dans le mode de choix de position

Permet d'enregister la position actuel du robot



Schema electrique



Limite de servo

SERVO	MIN	CENTRE	MAX	INDICE	COMMENTAIRE	DEFAULT
Servo 1	0 ↑	85	150 ↓	0		100
Servo 2	0	60	180	1		180
Servo 3	0 ←	100	180 →	2		0
Servo 4	0	85	180	3		180
Servo 5	70 close	85	130 open	4	Pince	70
Servo 6	30 ↓	85	180 ↑	5		70
Servo 7	0	120	180	6	Position à 90 == 0	0
Servo 8	0 ←	65	180 →	7		170
Servo 9	0	85	180	8		85
Servo 10	40 open	85	94 close	9	Pince	85
Servo 11	0 ←	75	180 →	10	Tête yeux	75
Servo 12	85 ↓	160	180 ↑	11	Le cœur	160

