

LIVRABLES DU PROJET

SUIVI DE PROJET B3 INFORMATIQUE 2023-2024

Datasheet du projet

AUTEUR :

BAPTISTE GADEBILLE

19 juin 2024

Table des matières

1 Section Code	2
1.1 Introduction	2
1.2 Architecture du Code	2
1.3 Utilisation des PID	2
1.3.1 Fonctionnement des PID	2
1.3.2 Calibration des PID	3
1.3.3 Application dans la Simulation	3
1.3.4 Performance et Stabilité	3
1.3.5 Optimisation et Améliorations	3
1.4 Visualisation des Données	4
1.5 Fonctions Mathématiques et Physiques	4
1.5.1 Moment d'inertie	4
1.5.2 Loi de la dynamique	4
1.5.3 Calculs et Transformations	4
1.6 Fonctions Python du Code	5
1.6.1 Classe RobotSimulator	5
1.6.2 Fonctions de Contrôle PID	5
1.6.3 Mise à Jour et Affichage	5
1.7 Instructions d'installation et d'exécution du code Python	6
2 assemblage du robot	7
3 montage électronique	8
4 section Matériel	10
4.1 Liste du matériel	10
4.1.1 Outils	10
4.1.2 Visserie	10
4.1.3 Modules et carte du projet	10
4.1.4 Autre matériels	10
4.1.5 Logiciels	10

1 Section Code

1.1 Introduction

Le projet de simulation de robot présenté ici vise à modéliser le comportement dynamique d'un robot basé sur les principes de la physique et du contrôle. À travers une interface utilisateur graphique développée en Python avec tkinter, ce simulateur permet de visualiser et d'analyser le mouvement d'un robot soumis à différentes conditions et paramètres physiques.

L'objectif principal de ce projet est de comprendre comment les contrôleurs PID peuvent être utilisés pour stabiliser et contrôler l'orientation d'un robot. En ajustant les paramètres PID (Proportional-Integral-Derivative), le système peut maintenir une position angulaire cible malgré les perturbations externes et les variations des paramètres physiques du robot.

1.2 Architecture du Code

L'architecture du code repose sur une classe principale RobotSimulator qui gère à la fois la simulation physique du robot et son interface utilisateur. Cette classe est construite autour de plusieurs variables membres qui représentent les états physiques du robot tels que l'angle de rotation, la position, la masse, la longueur, et d'autres paramètres liés à l'inertie.

L'interface utilisateur est réalisée à l'aide de widgets tkinter qui permettent à l'utilisateur de modifier en temps réel les paramètres du robot tels que l'angle initial, la masse, la hauteur, la longueur et l'inertie. Ces paramètres influencent directement le comportement du robot simulé.

1.3 Utilisation des PID

Les contrôleurs PID (Proportional-Integral-Derivative) sont des composants essentiels du projet de simulation de robot. Ils permettent de réguler et de stabiliser l'angle de rotation du robot en réponse à des perturbations et des changements dans l'environnement. Cette subsection explore comment les PID sont intégrés dans la simulation pour assurer un contrôle précis et efficace du système.

1.3.1 Fonctionnement des PID

Un contrôleur PID est composé de trois termes principaux :

- **Proportionnel (P)** : Ce terme est proportionnel à l'erreur actuelle entre l'angle désiré et l'angle réel du robot. Il agit immédiatement pour corriger cette erreur.

- **Integral (I)** : Le terme intégral est la somme cumulative des erreurs passées sur une période de temps. Il permet de corriger les erreurs systématiques persistantes et d'atteindre plus rapidement la valeur cible.
- **Dérivé (D)** : Ce terme est proportionnel à la variation instantanée de l'erreur. Il anticipe les changements futurs dans l'erreur et permet de réagir rapidement aux fluctuations.

En ajustant les coefficients K_p , K_i et K_d de ces termes, on peut optimiser les performances du contrôleur PID pour différentes conditions et exigences du système.

1.3.2 Calibration des PID

Avant de démarrer la simulation, les paramètres PID doivent être calibrés pour s'adapter aux caractéristiques spécifiques du robot et aux conditions de simulation. Cela est réalisé à l'aide d'un processus itératif où différentes combinaisons de K_p , K_i et K_d sont testées pour minimiser l'erreur entre l'angle désiré et l'angle réel du robot.

La méthode `calibrate_pid` dans le code effectue cette calibration en simulant le comportement du robot avec différentes valeurs de PID et en sélectionnant celles qui offrent les meilleures performances de contrôle.

1.3.3 Application dans la Simulation

Dans la fonction `simulate_pid`, les coefficients PID calibrés sont utilisés pour calculer la force F appliquée au robot à chaque itération. Cette force est convertie en accélération angulaire α , qui influence directement la vitesse de rotation du robot.

Le contrôleur PID permet ainsi au robot de maintenir sa position cible malgré les perturbations externes, en ajustant continuellement l'angle en fonction de l'erreur détectée et des prévisions de l'erreur future.

1.3.4 Performance et Stabilité

La performance du contrôleur PID est évaluée en surveillant l'évolution de l'angle du robot, ainsi que des composantes P , I et D du contrôle. Ces données sont enregistrées et peuvent être visualisées sous forme de graphiques pour analyser la stabilité et l'efficacité du contrôle PID sur la durée de la simulation.

1.3.5 Optimisation et Améliorations

L'utilisation des PID offre une base solide pour contrôler le robot dans des conditions diverses. Des améliorations potentielles incluent l'optimisation des paramètres PID pour différents scénarios d'utilisation, ainsi que l'implémentation de techniques avancées de contrôle pour répondre à des exigences spécifiques telles que la robustesse contre les perturbations externes.

1.4 Visualisation des Données

Les données collectées pendant la simulation, telles que l'angle du robot, les valeurs de P, I et D du PID, sont enregistrées à chaque étape pour analyse ultérieure. À la fin de la simulation ou à la demande de l'utilisateur, ces données sont tracées à l'aide de bibliothèques graphiques comme matplotlib. Les graphiques obtenus permettent de visualiser l'évolution temporelle de l'angle du robot et des composantes du contrôleur PID, offrant ainsi une compréhension visuelle de la performance du système de contrôle.

1.5 Fonctions Mathématiques et Physiques

Pour calculer le comportement physique du robot simulé, plusieurs fonctions mathématiques et physiques sont essentielles. Ces fonctions sont utilisées pour déterminer la dynamique du mouvement, la résistance à la rotation et d'autres paramètres critiques du système.

1.5.1 Moment d'inertie

Le moment d'inertie I du robot est crucial pour évaluer sa réponse à la rotation. Il est calculé en fonction de la masse m , de la hauteur h , et de la longueur l du robot, selon la formule :

$$I = \frac{1}{12}m(h^2 + l^2)$$

Cette formule est dérivée de la géométrie du robot, où m est la masse totale, h est la hauteur du robot par rapport à l'axe de rotation, et l est sa longueur.

1.5.2 Loi de la dynamique

La loi de la dynamique est appliquée pour convertir la force calculée par le contrôleur PID en une accélération angulaire α , qui affecte directement la vitesse de rotation ω du robot. Pour un système rotatif avec une force F appliquée à une distance L du centre de rotation, et un moment d'inertie I , l'accélération angulaire α est donnée par :

$$\alpha = \frac{F \cdot L}{I}$$

Cette équation permet de modéliser comment la force appliquée par le contrôleur PID influence la vitesse angulaire du robot au fil du temps, en prenant en compte sa structure physique et son inertie.

1.5.3 Calculs et Transformations

Pour intégrer ces concepts dans la simulation, des calculs itératifs sont effectués à chaque pas de temps Δt . La fonction `simulate_pid` utilise ces formules pour calculer la réponse du robot aux forces appliquées par le contrôleur PID, en estimant l'angle

de rotation en fonction des paramètres physiques du robot et des valeurs actuelles des paramètres PID K_p , K_i , et K_d .

Ces fonctions mathématiques et physiques sont cruciales pour le bon fonctionnement et la précision de la simulation de robot, permettant ainsi une modélisation réaliste du comportement dynamique du système sous contrôle PID.

1.6 Fonctions Python du Code

Le projet de simulation de robot repose sur une implémentation détaillée en Python, utilisant plusieurs fonctions pour modéliser la dynamique du robot et gérer l'interface utilisateur. Cette subsection explore les principales fonctions du code et leur rôle dans la simulation.

1.6.1 Classe RobotSimulator

La classe principale `RobotSimulator` est responsable de l'intégralité de la simulation et de l'interaction avec l'utilisateur. Voici les fonctions clés de cette classe :

- `init_variables()` : Initialise toutes les variables nécessaires pour la simulation, y compris l'angle initial, la position, la masse, la hauteur, la longueur, et les paramètres PID.
- `create_widgets()` : Crée les éléments d'interface utilisateur à l'aide de tkinter, tels que les entrées pour modifier les paramètres du robot et les boutons pour démarrer, arrêter et réinitialiser la simulation.
- `start_simulation()` : Démarrer la simulation en initialisant les paramètres PID avec la méthode `calibrate_pid()` et en appelant la méthode `update_robot()` pour mettre à jour l'état du robot à chaque intervalle de temps.
- `stop_simulation()` : Arrête la simulation en appelant la méthode `plot_pid()` pour afficher les données collectées.
- `reset_simulation()` : Réinitialise tous les paramètres de la simulation et efface l'interface utilisateur pour permettre une nouvelle configuration.

1.6.2 Fonctions de Contrôle PID

Les fonctions liées au contrôle PID dans la classe `RobotSimulator` comprennent :

- `calibrate_pid()` : Cette fonction calibre les paramètres PID en testant différentes combinaisons de valeurs et en sélectionnant celles qui minimisent l'erreur de contrôle, en utilisant la méthode `simulate_pid()`.
- `simulate_pid(Kp, Ki, Kd)` : Simule le comportement du robot avec les paramètres PID spécifiés (ou calibrés), calculant la force appliquée au robot à chaque itération et ajustant l'angle en fonction de cette force.

1.6.3 Mise à Jour et Affichage

Les fonctions responsables de la mise à jour de l'état du robot et de l'affichage des données comprennent :

- **update_robot()** : Met à jour graphiquement la position et l'angle du robot sur l'interface utilisateur tkinter en fonction des forces calculées à l'aide du contrôleur PID.
- **plot_pid()** : Affiche les données collectées pendant la simulation sous forme de graphiques à l'aide de la bibliothèque matplotlib, montrant l'évolution de l'angle du robot et des composantes P, I, et D du contrôle PID au fil du temps.

Ces fonctions interagissent ensemble pour simuler et contrôler dynamiquement le comportement du robot, offrant une plateforme interactive pour étudier les principes de la robotique et du contrôle automatique.

1.7 Instructions d'installation et d'exécution du code Python

1. Installez la dernière version de Python :

Téléchargez et installez la dernière version de Python à partir du site officiel : [python.org](https://www.python.org). Assurez-vous d'inclure Python dans votre PATH lors de l'installation.

2. Installez pip :

pip est généralement inclus avec les installations de Python à partir de la version 3.4. Vous pouvez vérifier si pip est installé en exécutant :

```
pip --version
```

Si pip n'est pas installé, suivez les instructions [ici](#) pour l'installer.

3. Installez les bibliothèques nécessaires :

Utilisez pip pour installer les bibliothèques requises. Ouvrez un terminal ou une invite de commandes et exécutez :

```
pip install numpy matplotlib
```

4. Importez les modules dans votre code Python :

Assurez-vous que votre script Python importe les modules nécessaires. Voici un exemple d'importation de ces modules dans votre script `main.py` :

```
import tkinter as tk
import math
import numpy as np
import matplotlib.pyplot as plt
```

5. Exécutez votre script Python :

Une fois toutes les bibliothèques installées et votre script prêt, vous pouvez exécuter votre script `main.py` en utilisant la commande suivante :

```
python3 main.py
```

2 assemblage du robot

Pour assembler le robot, commencez par positionner les trois pièces principales : les deux supports de moteurs, situés respectivement à gauche et à droite de la pièce centrale. Chaque pièce est pourvue de quatre extrusions, qui permettent non seulement de les maintenir solidement ensemble à l'aide de vis, mais aussi de fixer des composants additionnels pour optimiser la fonctionnalité globale du robot.

Une fois les trois pièces principales correctement alignées et fixées ensemble, passez à l'installation de la pièce supplémentaire au-dessus de la pièce centrale. Cette pièce s'emboîte parfaitement, offrant une base stable et sécurisée pour le montage des éléments électroniques. Vous pouvez ensuite fixer le "pont en H", une composante essentielle pour le contrôle précis des moteurs. Sous le "pont en H", glissez la carte esp32, un élément central pour la gestion du robot. Ajoutez ensuite les capteurs cruciaux tels que l'accéléromètre et le gyroscope sur la pièce, ces capteurs sont indispensables pour assurer une navigation précise et stable.

Cette configuration garantit une structure robuste et fonctionnelle, facilitant l'ajout et le maintien des composants nécessaires au bon fonctionnement du robot. De plus, elle offre une grande flexibilité pour d'éventuelles modifications ou améliorations futures, permettant d'adapter facilement le robot à des besoins spécifiques ou à des évolutions technologiques.

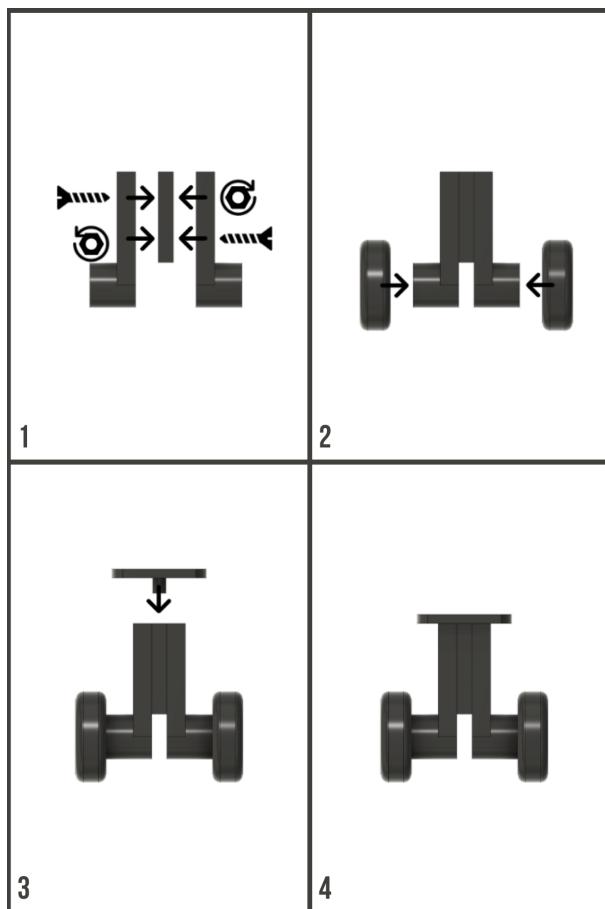


FIGURE 1 – Assemblage des pièces du robot

3 montage électronique



FIGURE 2 – Voici les partie principales en électronique du projet

s

Voici les composants principaux de mon projet robotique :

- L298N : Module de commande de moteur Le L298N est un module permettant de contrôler deux moteurs à courant continu ou un moteur pas à pas. Capable de gérer des tensions élevées et des courants importants, il est idéal pour les applications robotiques nécessitant une commande précise et fiable des moteurs.
- MMA8451Q : Accéléromètre numérique 3 axes L'accéléromètre numérique MMA8451Q mesure les accélérations sur trois axes avec une résolution de 14 bits ou 8 bits. Ce capteur détecte les mouvements et l'orientation du robot, fournissant des données précises pour la navigation et le contrôle de la stabilité.
- MULTICOMP PRO MM28DC : Moteur à courant continu Le moteur MM28DC de MULTICOMP PRO est un moteur à courant continu de 6 V, tournant à 9600 tours par minute (rpm) avec un couple de 20 g-cm. Avec une puissance de 1,64 W et un diamètre de 20,1 mm, ce moteur compact et puissant est parfait pour les applications robotiques où l'espace est limité.
- MPU6050 : Module gyroscope 3 axes Le MPU6050 combine un gyroscope et un accéléromètre 3 axes, fournissant des données sur l'orientation et le mouvement du robot. Ce capteur est essentiel pour suivre avec précision la position et la direction, contribuant à la stabilité et au contrôle du robot.
- Module ESP32 : ESP-WROOM-32 Le module ESP32 ESP-WROOM-32 est une puce puissante et polyvalente qui combine un microcontrôleur avec des capacités de connectivité Wi-Fi et Bluetooth. Il est idéal pour les projets robotiques nécessitant une communication

sans fil, la gestion des capteurs et des actuateurs, ainsi que des capacités de traitement avancées pour le contrôle et l'automatisation du robot.

Je prévoyais d'utiliser une batterie de 7 V avec le moteur sur l'image ci-dessus, mais je n'ai pas eu l'occasion de réaliser des tests en temps réel et de sélectionner une batterie. Le dimensionnement des moteurs et de la batterie sont liés, il est crucial d'effectuer ces tests pour garantir une performance optimale.

voici un schéma du montage électronique :

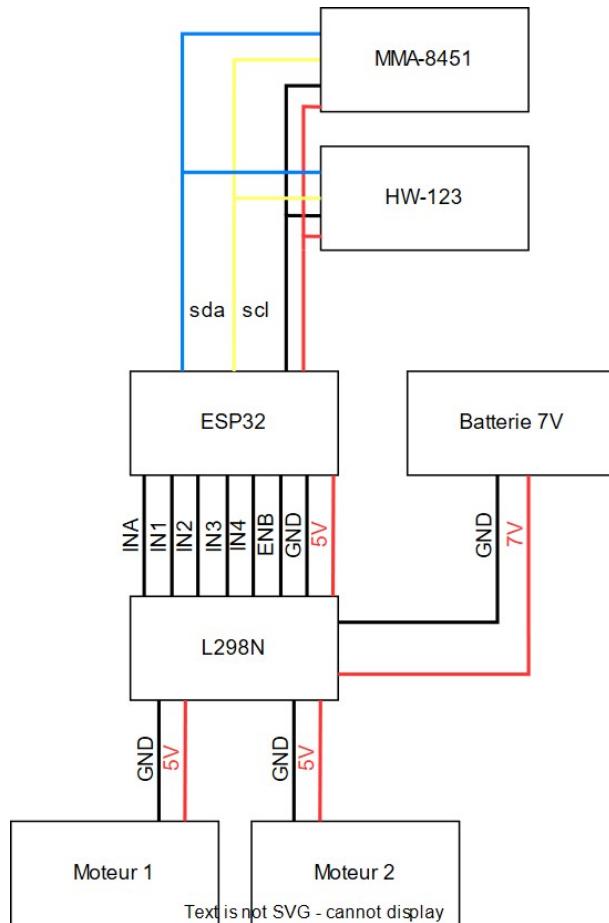


FIGURE 3 – montage électronique

4 section Matériel

4.1 Liste du matériel

Voici la liste du matériel nécessaire pour le projet :

4.1.1 Outils

- Multimètre
- Un générateur électrique portable
- Un oscilloscope
- Une pince plate
- Un réglet
- Un pied à coulisse
- Un fer à souder
- Éponge de fer à souder
- Tournevis cruciforme ×2 (taille de vis M1 et M2)

4.1.2 Visserie

- Vis M2 ×8
- Vis M3 ×8
- Insert fileté thermofixés taille M2 ×8
- Insert fileté thermofixés taille M3 ×8

4.1.3 Modules et carte du projet

- Une ESP32
- Un cable Micro-USB
- Gyroscope HW-123
- Accelerometer MMA8451
- Pont en H L298N
- Moteurs DC ×2

4.1.4 Autre matériels

- Un ordinateur portable
- Une souris
- Un écran (Facultatif)
- Une Alimentation pour ordinateur portable
- Imprimante 3D (un plateau d'au moins 150 × 150)
- Un cable d'alimentation pour l'imprimante 3D
- Bobine de filament PLA
- Bobine d'étain
- Cable Dupont

4.1.5 Logiciels

- VSCode
- Esptool

- Extension Pymakr
- Python3 + MicroPython firmware
- Fusion 360 ou autre logiciel de modélisation 3D CAO/DAO
- Slicer BambuStudio ou Ultimaker Cura...
- Git (logiciel de gestion de versions décentralisé)
- Conda (gestionnaire de packages et d'environnement)
- OS Windows et Linux pour de bonnes conditions de développement
- Moteur de recherche Firefox (Free & OpenSource)