Université Sultan Moulay Slimane

Faculté des Sciences et Techniques Béni Mellal

Département de Génie électrique

Rapport de Projet

Filière : Cycle Ingénieur Génie Électrique – Option : Systèmes Embarqués et Informatique Industrielle.

Rapport de Conception : Stabilisation d'un Bicopter à Axe Unique via un Contrôleur PID

Réalisé par:

Anass Ghilmi

Abderrazak Aziz

Abdessalam El Asri

Encadré par : Pr L.Elhajjami

Année académique : 2024/2025

1.0 Introduction

Ce rapport a pour objectif de documenter la conception, la construction et la stratégie de contrôle d'un système de *Bicopter* contraint à un axe unique. L'objectif principal de ce projet est de démontrer et d'analyser l'application d'un contrôleur PID (Proportionnel-Intégral-Dérivé) pour la stabilisation active du système. La mise en œuvre s'appuie sur un microcontrôleur Arduino pour le traitement logique et une centrale inertielle (IMU) MPU-6050 pour la mesure de l'orientation. Cette expérience constitue une base fondamentale et une validation de principe pour le développement de systèmes de contrôle de vol plus complexes, notamment pour les drones multi-axes.

Ce document détaillera dans un premier temps la conception matérielle et l'architecture du banc d'essai.

2.0 Conception et Architecture Matérielle

La réussite des tests de contrôle repose sur une conception matérielle robuste et soigneusement équilibrée. La construction d'un banc d'essai à axe unique a été privilégiée afin d'isoler et de calibrer efficacement les dynamiques de tangage (ou de roulis) sans les complexités d'un vol libre. Cette approche permet de se concentrer exclusively sur l'algorithme de stabilisation.

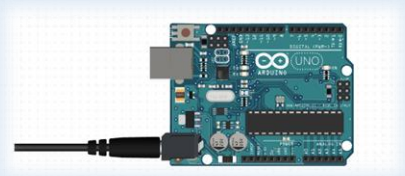
Structure Mécanique du Banc d'Essai

La structure mécanique est conçue pour permettre une rotation libre avec un minimum de friction autour d'un axe central. Elle est composée des éléments suivants :

- **Bras principal :** Une barre en bois, constitue le châssis du *Bicopter*. Les moteurs sont fixés à chaque extrémité.
- **Pivot central :** Un trou est percé exactement au centre de la barre pour y insérer une tige filetée ou un axe.
- **Support :** Le bras fixés à un support en bois, lui-même solidement vissé à une large base en bois. Cette base assure la stabilité de l'ensemble du système, empêchant tout basculement lors des tests à haute vitesse des moteurs.

Composants Électroniques

Le choix des composants électroniques a été guidé par leur accessibilité, leur fiabilité et leur compatibilité avec l'écosystème Arduino. Le tableau ci-dessous résume les composants clés et leur fonction.

Composant	Rôle et Spécification
Microcontrôleur Arduino 	Cerveau du système. Exécute la boucle de contrôle PID, traite les données du capteur et génère les signaux de commande pour les moteurs.
Centrale Inertielle (IMU) MPU-6050 	Capteur 6 degrés de liberté (accéléromètre 3 axes + gyroscope 3 axes). Mesure l'angle d'inclinaison et la vitesse angulaire du bras.
Moteurs Brushless (x2) 	Actionneurs principaux. Fournissent la poussée nécessaire pour contrer les déséquilibres et maintenir l'orientation horizontale.

Contrôleurs de Vitesse (ESC) (x2) 	Interfaces de puissance qui modulent le courant élevé de l'alimentation vers les moteurs en fonction des signaux de commande PWM de faible puissance (1000µs-2000µs) envoyés par l'Arduino.
Source d'alimentation 	Fournit la puissance nécessaire aux moteurs et à l'électronique. Une batterie LiPo 3S de 11.1V.

Avec l'architecture matérielle définie, la section suivante décrit les connexions électriques qui unissent ces composants.

3.0 Schéma Électrique et Connexions

Un câblage correct et soigné est une condition indispensable au bon fonctionnement du système. Il garantit non seulement une distribution de puissance adéquate aux moteurs, mais aussi une communication fiable et sans bruit entre le microcontrôleur et la centrale inertielle.

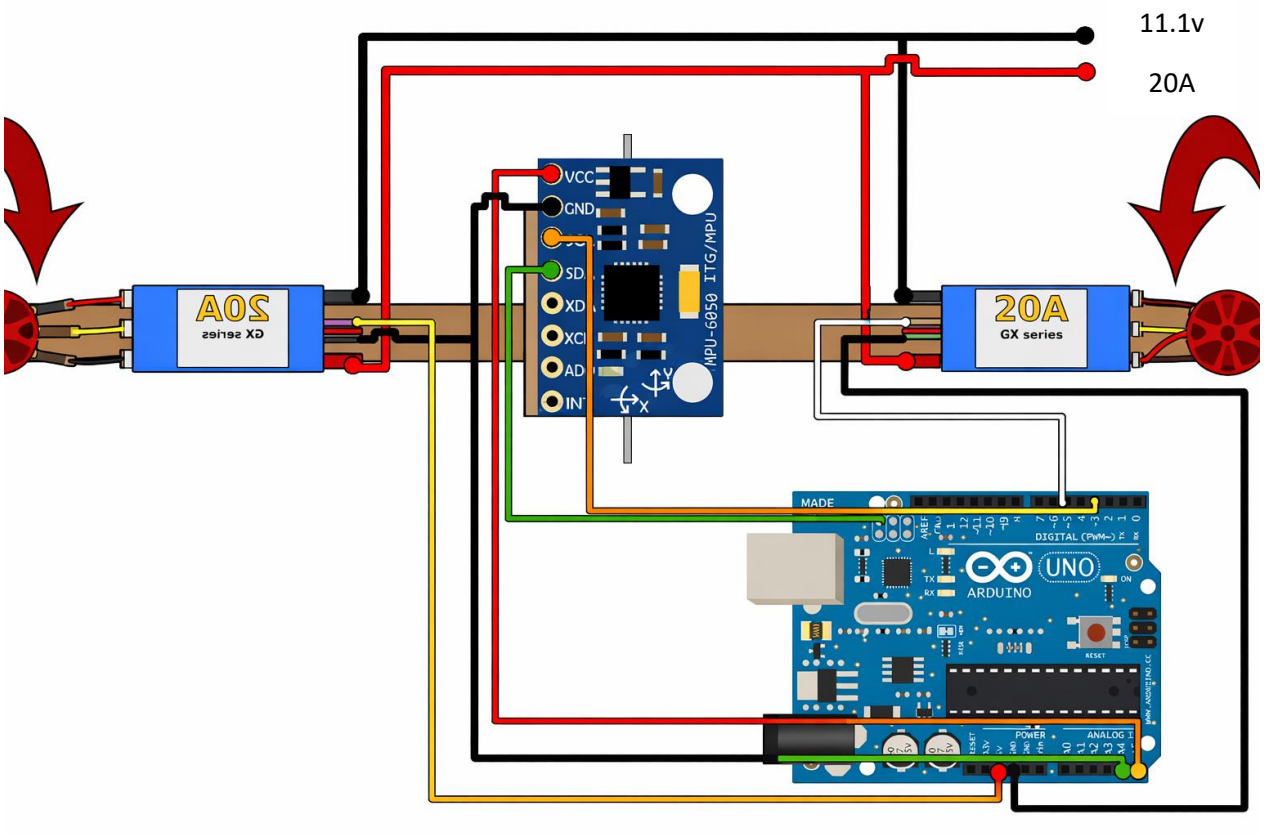


Diagramme de Connexion

Les connexions électriques entre les composants sont établies comme suit :

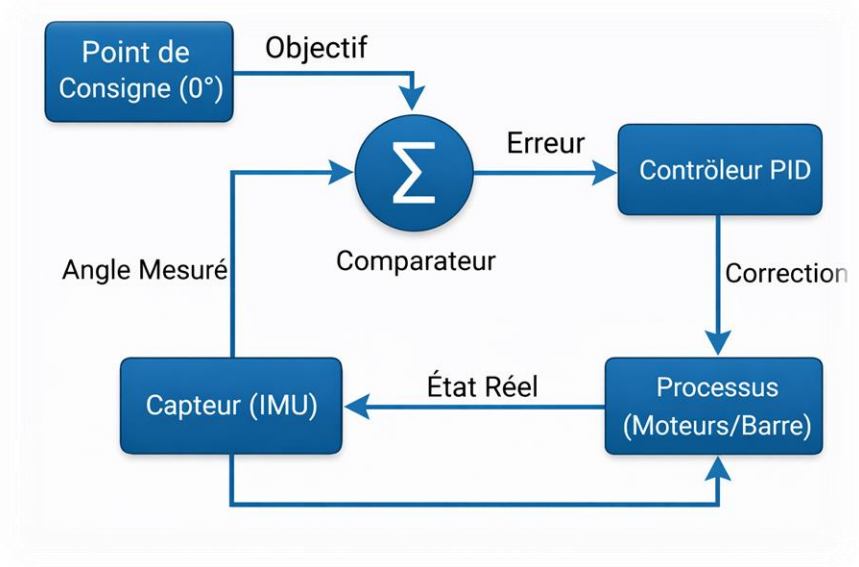
- **Alimentation :**
  - Les deux ESC sont alimentés directement par la source de tension de *11.1V* (batterie LiPo).
  - L'Arduino peut être alimenté via son port USB ou par le circuit d'élimination de batterie (BEC) 5V intégré à l'un des ESC.
- **Contrôle Moteur :**
  - Le fil de signal de l'ESC du moteur droit est connecté à la broche **3** de l'Arduino.
  - Le fil de signal de l'ESC du moteur gauche est connecté à la broche **5** de l'Arduino.
- **Communication IMU :**
  - La communication avec le MPU-6050 s'effectue via le protocole I2C.
  - La broche **SCL** (Serial Clock) du MPU-6050 est connectée à la broche **A5** de l'Arduino.
  - La broche **SDA** (Serial Data) du MPU-6050 est connectée à la broche **A4** de l'Arduino.
- **Placement des Composants :**
  - Il est crucial de placer le module IMU MPU-6050 le plus près possible du centre de rotation du banc d'essai. Un positionnement central garantit que le capteur

mesure l'inclinaison pure du bras sans être affecté par des accélérations parasites dues à son éloignement de l'axe de pivot.

Le matériel étant assemblé et câblé, la discussion peut maintenant se porter sur la logique de contrôle qui anime le système.

4.0 Stratégie de Contrôle et Traitement des Données

La stabilisation du *Bicopter* est réalisée par une boucle de rétroaction logicielle exécutée en continu par l'Arduino. Cette stratégie repose sur trois piliers fondamentaux : l'acquisition des données brutes des capteurs, le filtrage de ces données pour obtenir une estimation fiable de l'angle, et l'application d'un algorithme de contrôle PID pour calculer les corrections de poussée nécessaires.



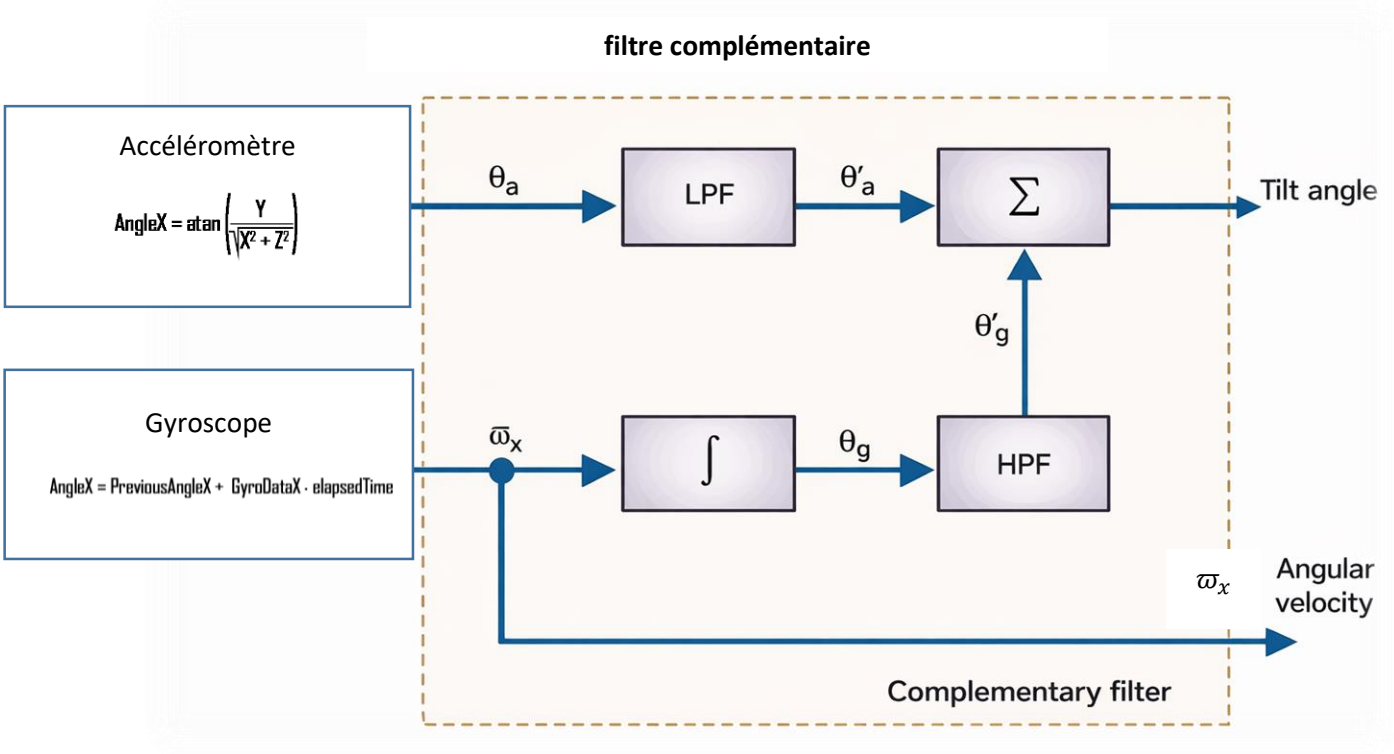
4.1 Le Défi de la Mesure : L'IMU et ses Limites Inhérentes

Le MPU-6050 est au cœur de la boucle de rétroaction. Il intègre deux capteurs essentiels, chacun présentant des avantages et des inconvénients :

- **L'accéléromètre (3 axes) :** Il mesure l'accélération linéaire, y compris l'accélération gravitationnelle terrestre constante (1g). En mesurant la répartition de ce vecteur de gravité sur ses axes, il est possible de calculer l'angle d'inclinaison du système par rapport à l'horizontale. Cependant, cette mesure est sensible au "bruit" et à toute accélération non gravitationnelle (par exemple, les vibrations des moteurs), ce qui la rend peu fiable à court terme.
- **Le gyroscope (3 axes) :** Il mesure la vitesse angulaire (la vitesse de rotation) sur chaque axe. En intégrant cette vitesse au fil du temps, on peut obtenir une estimation de l'angle. Cette mesure est très précise pour les changements rapides, mais elle souffre d'un biais cumulatif appelé "dérive" (drift), qui la rend inexacte sur le long terme.

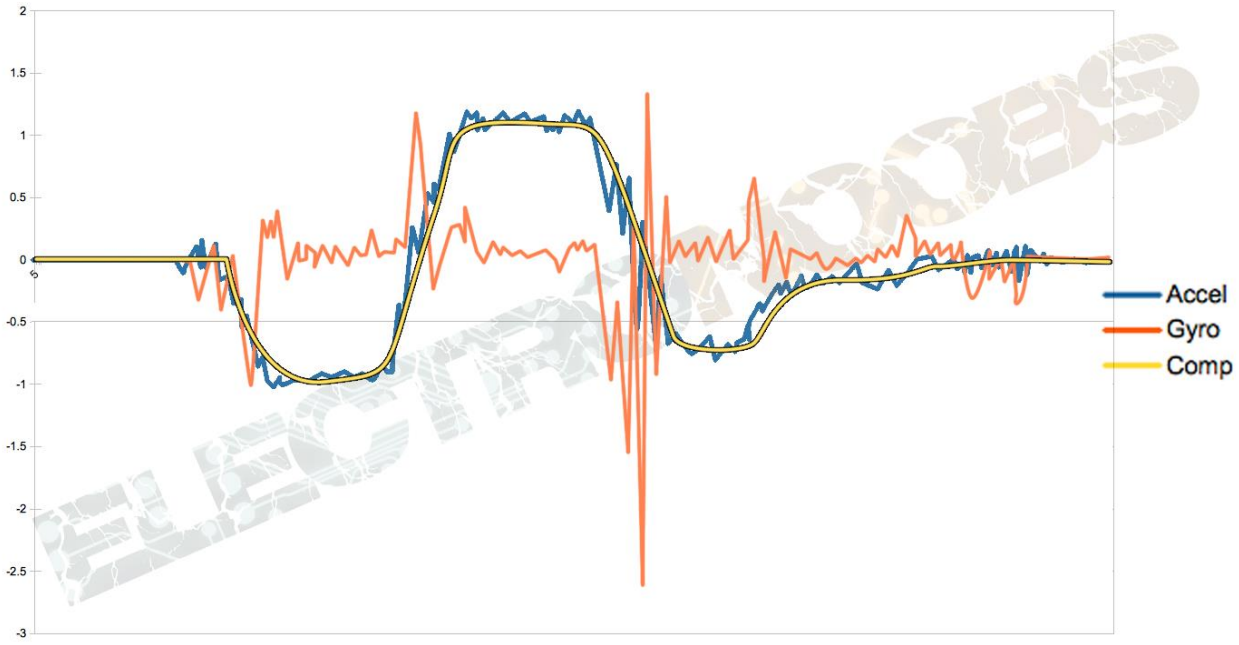
4.2 Filtrage Complémentaire pour l'Estimation de l'Angle

Pour surmonter les limitations individuelles de l'accéléromètre et du gyroscope, une technique de fusion de capteurs est nécessaire. Le filtre complémentaire a été choisi pour sa simplicité de mise en œuvre et son efficacité. Il combine les forces des deux capteurs : la stabilité à long terme de l'accéléromètre et la précision à court terme du gyroscope. Son nom "complémentaire" provient du fait que les poids accordés à chaque capteur s'additionnent pour former un tout (0.98 + 0.02 = 1), signifiant une confiance totale dans la mesure fusionnée.



La formule de filtrage implémentée est la suivante :

```
Total_angle[1] = 0.98 * (Total_angle[1] + Gyro_angle[1] * elapsedTime) + 0.02
* Acceleration_angle[1]
```

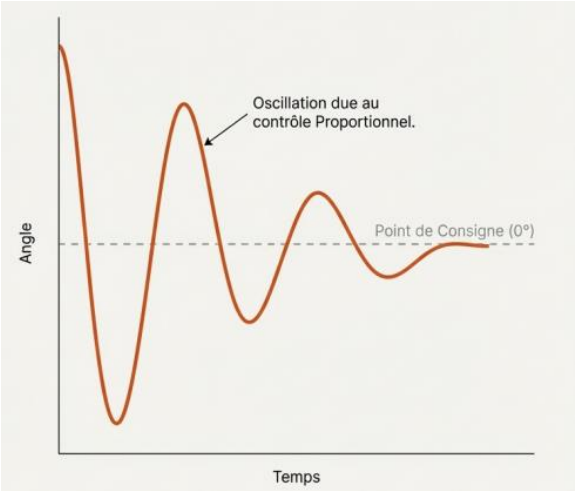


Cette équation attribue un poids prépondérant (98%) à l'angle calculé à partir du gyroscope, qui est fiable pour les changements rapides, tout en le corrigeant lentement (2%) avec l'angle de l'accéléromètre, qui est stable sur le long terme et ne dérive pas.

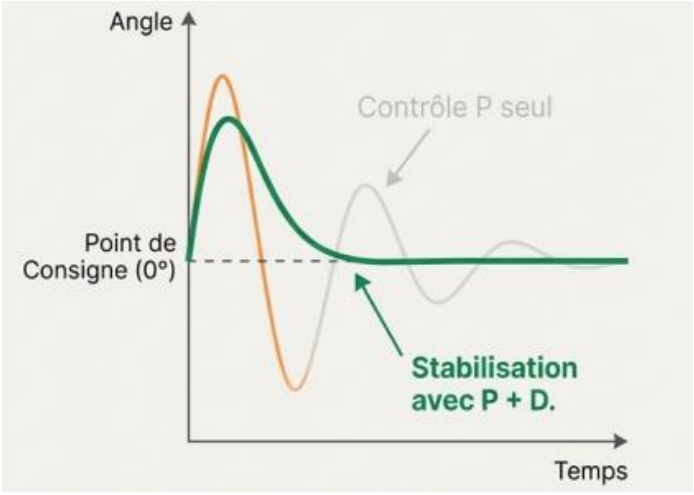
4.3 Conception du Contrôleur PID

Le contrôleur PID (Proportionnel-Intégral-Dérivé) est un mécanisme de rétroaction qui calcule en continu une valeur d'erreur (`error`). Cette erreur est définie comme la différence entre un point de consigne (l'angle désiré, soit `0°` pour une position horizontale) et la variable de processus mesurée (l'angle réel, `Total_angle[1]`, qui correspond à l'axe Y de l'IMU, aligné avec l'axe de rotation). La conception du contrôleur s'est faite de manière itérative, en ajoutant chaque terme pour résoudre une faiblesse spécifique du précédent.

1. **Terme Proportionnel (P) - La Réponse Initiale :**
    - **Calcul :** `pid_p = kp * error`
    - **Analyse :** Ce terme constitue la base du contrôle. Il génère une action corrective directement proportionnelle à l'erreur actuelle. Si le bras est incliné, une force de rappel est appliquée. Cependant, un contrôleur purement proportionnel a une faiblesse inhérente : en corrigeant l'erreur, il prend de la vitesse et dépasse le point de consigne, puis corrige dans l'autre sens, créant une oscillation perpétuelle. Des valeurs de `kp` élevées accélèrent la réponse mais aggravent les oscillations.

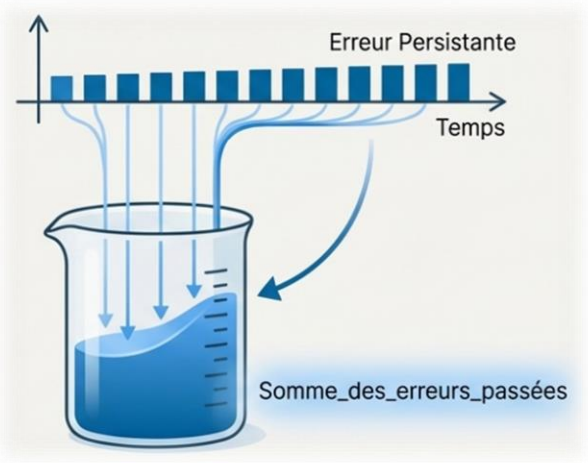


2. **Terme Dérivé (D) - L'Amortisseur :**
    - **Calcul :** `pid_d = kd * ((error - previous_error) / elapsedTime)`
    - **Analyse :** Pour contrer les oscillations, le terme dérivé est introduit. Il agit comme un frein en anticipant le comportement futur du système. Il ne se base pas sur l'erreur elle-même, mais sur sa *vitesse de variation*. Si le bras se rapproche rapidement de l'horizontale (erreur diminue vite), le terme D réduit la correction pour éviter le dépassement. Il est donc essentiel pour amortir les oscillations et stabiliser rapidement le système.



3. **Terme Intégral (I) - Le Réglage Fin :**
    - **Calcul :** `pid_i = pid_i + (ki * error)`
    - **Analyse :** Un système P-D peut se stabiliser, mais souvent avec une légère erreur résiduelle (par exemple, un déséquilibre de poids minime qui le fait stagner à 0.5° au lieu de 0°). Le terme intégral résout ce problème en accumulant les erreurs passées. Si une petite erreur persiste, la somme intégrale augmente progressivement au fil du temps, générant une correction de plus en plus forte jusqu'à ce que l'erreur soit complètement éliminée, garantissant ainsi que le système atteint précisément le point de consigne.

- **Limitation du Terme Intégral (Anti-Windup) :** L'implémentation inclut une protection cruciale : `if(-3 <error <3)`. Le terme intégral n'est activé que lorsque l'erreur est faible (entre -3 et +3 degrés). Cela empêche le terme `pid_i` de s'accumuler jusqu'à des valeurs extrêmes ("integral windup") lors de perturbations importantes et prolongées (par exemple, si l'opérateur maintient le bras incliné manuellement). Cette technique garantit que le terme I ne sert qu'à sa fonction de réglage fin près du point de consigne, sans déstabiliser le système.



La sortie totale du contrôleur, qui sera utilisée pour ajuster la vitesse des moteurs, est la somme de ces trois composantes : `PID = pid_p + pid_i + pid_d`.

La théorie de contrôle étant établie, la section suivante détaillera sa mise en œuvre concrète dans le code Arduino.

5.0 Mise en Œuvre Logicielle

L'architecture logicielle est implémentée sur un microcontrôleur Arduino. Le code est structuré de manière classique autour de deux fonctions principales : `setup()`, exécutée une seule fois au démarrage pour l'initialisation, et `loop()`, qui exécute la boucle de contrôle de manière continue et aussi rapidement que possible.

Fonction `setup()`

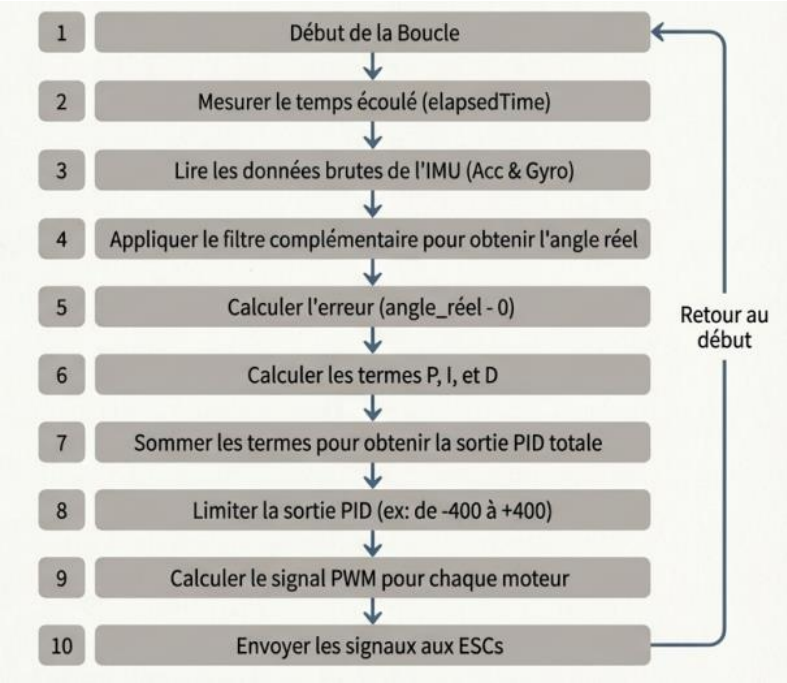
Cette fonction prépare le système pour le fonctionnement. Les étapes clés sont les suivantes :

- Démarrage de la communication I2C avec le MPU-6050.
- Attachement des broches de sortie PWM (3 et 5) aux objets `Servo` pour le contrôle des moteurs.

- Envoi d'un signal minimum (1000µs) aux deux ESC. Cette étape est nécessaire pour les armer et les préparer à recevoir des commandes de vitesse.
- Introduction d'un délai de 7 secondes pour donner à l'opérateur le temps de connecter l'alimentation principale en toute sécurité.

Fonction `loop()` - La Boucle de Contrôle

La fonction `loop()` exécute le cycle de lecture, de calcul et d'actionnement. Le flux logique se déroule comme suit :



A. **Mesure du Temps :** La variable `elapsedTime` est calculée à chaque itération. Elle représente le temps écoulé depuis la boucle précédente et est essentielle pour la cohérence temporelle des calculs du filtre et des termes intégral et dérivé du PID.
B. **Lecture des Capteurs :** Le code envoie une requête via I2C au MPU-6050 pour lire les registres contenant les données brutes de l'accéléromètre et du gyroscope.
C. **Traitement des Données :** Les valeurs brutes des capteurs sont converties en unités physiques (g pour l'accélération, degrés/seconde pour la vitesse angulaire). Le filtre complémentaire est ensuite appliqué pour fusionner ces données et calculer l'angle d'inclinaison final et fiable (`Total_angle[1]`), correspondant à l'axe Y.
D. **Calcul du PID :** L'erreur est calculée (`error = Total_angle[1] - desired_angle`). Ensuite, les termes proportionnel (`pid_p`), intégral (`pid_i` avec sa condition anti-windup) et dérivé (`pid_d`) sont calculés, puis sommés pour obtenir la sortie PID totale.
E. **Limitation de la Sortie :** La valeur de sortie `PID` est contrainte dans une plage de -1000 à 1000. Cela évite que la correction ne devienne excessive et ne sature les commandes des moteurs.
F. **Commande des Moteurs :** Les largeurs d'impulsion PWM finales sont calculées via un mécanisme de **poussée différentielle** : `pwmLeft = throttle + PID;` et `pwmRight =`

`throttle - PID;`. Cette approche est au cœur de l'actionnement. Une sortie PID positive, indiquant une inclinaison vers la droite, augmente simultanément la poussée du moteur gauche et diminue celle du moteur droit, générant un couple de correction puissant pour ramener le bras à l'horizontale.

G. **Saturation des Commandes :** Les valeurs PWM finales (`pwmLeft` et `pwmRight`) sont vérifiées pour s'assurer qu'elles restent dans la plage de fonctionnement valide des ESC (généralement entre 1000µs et 2000µs).
H. **Actionnement :** Les signaux PWM finaux sont envoyés aux ESC respectifs via la fonction `writeMicroseconds()`, commandant ainsi la vitesse des moteurs pour corriger l'inclinaison.

Cette boucle se répète continuellement, permettant au système de réagir en temps réel aux perturbations. L'étape suivante consiste à régler les paramètres de ce contrôleur.

6.0 Paramètres et Résultats

L'efficacité théorique du contrôleur PID ne se traduit en une performance pratique que par un réglage précis de ses constantes (`kp`, `ki`, `kd`). Ces paramètres sont intrinsèquement liés à la dynamique physique du système (poids, longueur du bras, puissance des moteurs) et doivent être déterminés de manière empirique.

Calibrage des ESC

Un prérequis fondamental pour un contrôle équilibré est le calibrage des ESC. Cette procédure garantit que les deux contrôleurs de vitesse répondent de manière identique à la même plage de signaux PWM, soit de 1000µs (arrêt moteur) à 2000µs (pleine puissance). Sans ce calibrage, un même signal de commande pourrait entraîner des poussées différentes, introduisant un biais que le contrôleur PID aurait du mal à compenser.

Constantes PID Finales

Après de multiples tests et ajustements manuels, le jeu de constantes suivant, implémenté dans le code final, a fourni les résultats de stabilisation les plus performants pour ce banc d'essai spécifique.

Constante	Valeur
`Kp`	3.55
`Ki`	0.005
`Kd`	2.05

Il est important de noter que ces valeurs sont le fruit d'un réglage empirique. Un autre jeu de constantes (`kp=3.44`, `ki=0.048`, `kd=1.92`) a également été identifié comme fonctionnel durant les tests, ce qui renforce l'idée que le réglage PID est hautement spécifique à un système physique donné et que plusieurs solutions peuvent exister.

Analyse des Résultats

Avec les constantes PID correctement réglées, le système a démontré une excellente capacité d'auto-stabilisation. Le banc d'essai maintient une position horizontale stable, avec des oscillations minimes. Plus important encore, il réagit de manière efficace et rapide aux perturbations externes, telles qu'une poussée manuelle sur l'un des côtés du bras. Le système revient rapidement et sans dépassement excessif à sa position de consigne (0°), validant ainsi l'efficacité de la stratégie de contrôle mise en œuvre.

Ces résultats concluants permettent de dresser le bilan final du projet.

7.0 Conclusion

Ce projet a permis de réaliser avec succès la conception, la construction et la programmation d'une plateforme de *Bicopter* à axe unique capable d'une auto-stabilisation active. Les objectifs fixés ont été pleinement atteints, démontrant la viabilité de l'approche choisie.

L'évaluation de la solution montre que la combinaison d'un microcontrôleur Arduino, d'une centrale inertielle MPU-6050, d'un filtre complémentaire pour la fusion des données et d'un contrôleur PID bien réglé constitue une architecture robuste, économique et accessible pour le contrôle de vol. La méthodologie de décomposition du problème en un seul axe s'est avérée particulièrement efficace pour isoler et résoudre les défis liés à la stabilisation.

En conclusion, les principes de contrôle, l'architecture matérielle et le code développés dans le cadre de ce projet constituent une base solide et validée. Ils peuvent servir de point de départ fiable pour des applications plus complexes, notamment l'extension du contrôle à deux axes pour la stabilisation complète d'un drone de type quadricoptère.

Références

Anghel, A. G. (2025 / 12 / 21). *electronoobs*. Récupéré sur https://electronoobs.com: https://electronoobs.com/eng_robotica_tut6_1.php