REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE M'HAMED BOUGARA-BOUMERDES

Faculté de **T**echnologie

Département **I**ngénierie des **S**ystèmes **E**lectriques

Mémoire de Master

Présenté par

Djedir Chahira

Filière : **A**utomatique

Spécialité : **A**utomatique et **I**nformatique **I**ndustrielle

Etude et Commande d'un Robot Sous-Marin Semi-Autonome

Soutenu le 06 /10 /2020 devant le jury composé de:

M.Yakhlaf	Yacine	MCA	UMBB	Président
M.Riache	Samah	MCB	UMBB	Examinateur
M.Nafa	Fares	MCA	UMBB	Encadrent

Année Universitaire : 2019/2020

ملخص

هذا العمل يتضمن التحكم و المحاكاة للغواصة الشبه مستقلة بكاميرا وبالتالي ، بعد وصف جميع الخطوات والمواد اللازمة لتكوين هذه الغواصة (بطاقة اردوينو انو و محرك خطوة بخطوة و محرك بتيار مستمر) الذي يتضمن نمذجة ميكانيكية، كهربائية و برمجية قمنا بإعداد كل المحاكاة اللازمة لتحقيق عمل و تشغيل هذه الغواصة المقترحة وذلك باستعمال البرامج المناسبة:(اردوينو IDE بروتيوس .... إلخ) بالنسبة لجزء التحكم ، قمنا بتجميع قوانين تحكم قوية باستخدام أمر وضع الانزلاق هذا القانون المعمول به بطريقة مفصلة عن الاستقرار حسب نظرية ليابونوف. نتائج المحاكاة عن طريق برنامج مطلب توافق وتحقق صحة العمل النظري الذي تم تطويره .

الكلمات المفاتيح:

محرك بتيار مستمر، محرك خطوة بخطوة، IDEاردوينو ، ليابونوف ، مطلب ،بروتيوس ، بطاقة اردوينو انو

Résumé

Ce travail consiste à la commande, et la simulation d'un drone sous-marin semi autonome. Ainsi, après avoir décrit toutes les étapes et le matériel nécessaire pour la construction d'un tel robot (Arduino UNO, moteur CC, et moteur Stepper) qui inclut la modélisation mécanique, électronique et software, nous avons établis toutes les simulations nécessaires pour valider le fonctionnement du robot proposé en utilisant les logiciels nécessaires : Arduino, Proteus, . . . etc. Pour la partie commande, nous avons synthétisé des lois de commande robustes en utilisant la commande par les modes glissants. Cette synthèse de loi de commande est complémentée avec une étude détaillée de la stabilité en utilisant la théorie de Lyapunov. Les résultats de simulation via Matlab approuve et valide l'analyse théorique développée.

Mot clé :

Arduino UNO, moteur CC, moteur Stepper, Proteus, commande par mode glissant, Lyapunov, Matlab.

Abstract

This work consists of the control and simulation of a semi-autonomous underwater drone. Thus, after having described all the steps and the material necessary for the construction of such a robot (Arduino UNO, DC motor, and Stepper motor) which includes mechanical, electronic and software modeling, we have established all the simulations necessary to validate the operation of the proposed robot using the necessary software : Arduino, Proteus,. . . etc. For the control part, we synthesized robust control laws by using the control by the sliding modes. This control law synthesis is complemented with a detailed study of stability using Lyapunov theory. The simulation results via Matlab approve and validate the theoretical analysis developed.

Keywords :

Arduino UNO, motor CC, motor Stepper, Proteus, command by sliding mode, Lyapunov, Matlab.

Remerciement :

Nous remercions « Allah » de nous avoir donné la force et le courage pour réaliser ce modeste travail.

Nous remercions mon promoteur Mr. NAFA qui a suivi de très près ce travail, pour son aide, ses orientations sa disponibilité, et tous les conseils qui m'a prodigué pendant toute la durée de ce projet. Que ce travail soit un témoignage de notre gratitude et notre profond respect.

Nous adressons nos plus vifs remerciements aux membres du jury pour l'honneur qu'ils nous ont fait en acceptant d'être rapporteurs de notre mémoire.

Nous tenons à remercier tous les enseignants du Département Ingénierie des Systèmes électriques pour le dévouement avec lequel ils ont mené ma formation théoriquement et pratiquement pendant le cursus universitaire.

Dédicace :

Je dédié mon travail à :

Mes parents qui m'ont toujours encouragé, conseillé et soutenu inconditionnellement moralement et physiquement. Que ce modeste travail soit l'exaucement de vos vœux tant formulés, le fruit de vos innombrables sacrifices, bien que je ne vous en acquitterai jamais assez. Puisse Dieu, le Très Haut, vous accorde la santé, le bonheur, une longue vie et faire en sorte que jamais je ne vous déçoive.

Mes sœurs Amina et Chemes Dhiet que j'aime profondément que Dieu, le tout puissant, vous protège, et nous garde toujours soudé.

Mes chers grands-parents « Bababa et Mimi » qui m'ont accompagné par leurs prières, leurs douceur, puisse Dieu leurs prêter une longue vie, beaucoup de santé, et de bonheur dans les deux vies.

La mémoire de mon grand-père Sidou, ma grand-mère Ayi, et mon oncle Amo Mahrez, j'aurais tant aimé que vous soyez présents. Que Dieu ait vos âmes dans sa sainte miséricorde.

Mes oncles, mes tantes, mes chers cousins et cousines exceptionnellement « Tata Mouna », veuillez trouver dans ce travail l'expression de mon respect le plus profond et mon affection la plus sincère.

Mes copines Nabila et Kamou preuve que la famille ne s'arrête pas au song, merci d'etre toujours à mes côtés, et de m'encourager, merci Nabila pour ton précieux soutien, tes conseils, et de me rappeler d'etre positive. Merci Kamou pour ton aide dans tout les sens, et ta participation dans ce mémoire.

Mes amis proche Chahra, et Nadjet déjà 5ans, des souvenirs inoubliables pleins de joie et de folie votre amitié est exceptionnelle, vous resterez le meilleure cadeau de mon expérience universitaire, merci pour tout.

Mon conseiller, et mon ami fidèle « Souhil » le graphiste compétant et créatif qui m'a assisté dans les moments difficiles, merci pour tout le temps que tu as consacré pour moi et se donner à fond pour rendre mes photos professionnelles.

Mes chères amies Hynd, Fella, Katia merci pour votre soutien pendant 5ans sans faille on a eu pas mal de souvenirs , merci pour l'ambiance chaque matin avec vous, vous êtes magnifiques et adorable.
Mes collègues et tous mes amis merci pour votre soutien moral et intellectuel .

Table des matières

Table des figures vi

Liste des tableaux viii

Abréviations ix

Introduction générale 1

1 Généralités sur la robotique sous-marine 3
- 1.1 Les Applications . . . 6
  - 1.1.1 Civil . . . 6
  - 1.1.2 Militaire . . . 7
- 1.2 Les exemples . . . 7
  - 1.2.1 Véhicule sous-marin téléguidé « ROV » . . . 7
  - 1.2.2 Sous-marin Deutchland SM U-155 . . . 8
  - 1.2.3 Triomphant S616 . . . 8
  - 1.2.4 DRSV-1 Mystic . . . 9
  - 1.2.5 Sous-marin à propulsion humaine . . . 9
- 1.3 Le Futur . . . 9
  - 1.3.1 Drone soluble dans l'eau . . . 10
  - 1.3.2 Espion anguille . . . 10
  - 1.3.3 Vaisseau mère moitié raie moitié requin . . . 10
  - 1.3.4 Poissant volant d'attaque . . . 10
- 1.4 Concept théorique . . . 10
  - 1.4.1 Le principe de la poussée d'Archimède . . . 10
  - 1.4.2 Modèle mathématique de la poussée d'Archimède . . . 11
  - 1.4.3 Quand-est-ce-qu'un solide flotte, nage ou coule ? . . . 12

2 Les composants et les logiciels utilisés. 15
- 2.1 Carte Arduino . . . 15
- 2.2 La carte Arduino UNO . . . 16
- 2.3 Les moteurs avec leurs drivers . . . 18
  - 2.3.1 Moteur DC . . . 18
  - 2.3.2 Moteur Stepper (pas à pas) : . . . 22
- 2.4 Driver L293 . . . 26
- 2.5 Les logiciels utilisés dans notre projet . . . 27
  - 2.5.1 Arduino IDE . . . 27
  - 2.5.2 Proteus . . . 29
  - 2.5.3 Autocad . . . 30

2.6 Conclusion 31

3 Programmation et Simulation **32**
3.1 Explication du programme sur Arduino IDE 32
3.2 Simulation sur Proteus 33
3.3 L'harmonisation entre le programme et sa simulation 36
3.4 Les modèles des sous-marins sur Autocad 37
3.4.1 Le modèle 1 37
3.4.2 Le modèle 2 37
3.4.3 Les deux modèles sur Autocad 38
3.5 Conclusion 40

4 Commande non linéaire par les modes glissants Modélisation et Simulation sur Matlab **41**
4.1 Définition du mode glissant 41
4.2 Système à structure variable 41
4.3 Objectif de la commande par mode glissant 42
4.4 Les notions de base de la commande par mode glissant 42
4.4.1 La modélisation mathématique de la commande à structure variable 42
4.4.2 La condition d'existence par mode glissant 42
4.4.3 La surface du glissement 43
4.5 Condition de convergence 44
4.6 Fonction directe de la commutation 44
4.7 Définition et expression analytique de la commande 44
4.8 Phénomène de Chattering 47
4.8.1 La fonction sat 47
4.8.2 La fonction adoucie $SMOOTH$ 47
4.9 Domaines d'application de la commande par mode glissant 48
4.10 Modélisation cinématique du sous-marin 48
4.10.1 Définition des matrices de passage 51
4.10.2 Transformation des vitesses de translation et de rotation 51
4.10.3 Modélisation de l'angle de dérape en Lacet 52
4.11 Synthèse de la commande par les modes glissants 53
4.11.1 Commande de la profondeur z par fs 53
4.11.2 Commande de la profondeur z par U 54
4.11.3 Commande de la profondeur z avec l'angle de plongé Θ et une sur-tension u constante 56
4.12 Simulation sur Matlab et résultats obtenus 56
4.12.1 Commande 1 et 2 56
4.12.2 Commande 3 66
4.13 Conclusion 70

Conclusion Générale **71**

Bibliographie **72**

Annexe **1**

Table des figures

1.1 Sous marin de bushnell « Turtle » 4
1.2 Nautilus en 1797 en acier 4
1.3 Véhicule sous-marin téléguidé « ROV » 8
1.4 Sous-marin Deutchland SM U-155 8
1.5 Triomphant S616 9
1.6 Parallélépipède immergé dans un liquide 11
1.7 Corps qui coule 12
1.8 Corps qui nage 13
1.9 Corps qui flotte 13

2.1 Carte Arduino Uno 16
2.2 Microcontrôleur ATmega328 17
2.3 Moteur à courant-continu 19
2.4 Mécanisme d'un moteur à courant continu 21
2.5 Une coupe d'un moteur pas à pas à aimant permanent 24
2.6 Une coupe d'un moteur à rotor à réluctance variable 24
2.7 Une coupe d'un moteur à rotor hybride 25
2.8 L293DA sur logiciel Proteus 27
2.9 Fenêtre principale d'un Arduino IDE 28
2.10 Fenêtre principale d'un ISIS Proteus 30
2.11 Fenêtre principale d'Autocad 31

3.1 Première étape de simulation 33
3.2 Deuxième étape de simulation 34
3.3 Troisième étape de simulation 35
3.4 Le montage électrique complet avant simulation 35
3.5 Assiette positive et négative 37
3.6 Vue de dessus 38
3.7 Vue de derrière 38
3.8 Vue de face 39
3.9 Vue de coté 39
3.10 Vue de dessus 39
3.11 Vue de derrière 39
3.12 Vue de face 40
3.13 Vue de coté 40

4.1 Démonstration du mode de glissement 43
4.2 Commande appliquée aux systèmes à structure variable 45
4.3 Représentation de la fonction Sign 46
4.4 Représentation de la fonction SAT 47

4.5 Représentation de la fonction SMOOTH 48
4.6 Repère 3D 49
4.7 Définition du lacet - Vue de dessus 49
4.8 Définition du tangage - Vue de côté 50
4.9 Définition du roulis - Vue de face 50
4.10 Chemin du drone sous-marin par rapport au temps 57
4.11 Erreur pour atteindre la profondeur désirée 57
4.12 Angle de plongé 58
4.13 Angle de renforcement d'entrée 58
4.14 Représentation de la commande u 59
4.15 Chemin de notre drone sous-marin par rapport au temps 60
4.16 Erreur pour atteindre la profondeur désirée 60
4.17 Angle de plongé 61
4.18 Angle de renforcement d'entrée 61
4.19 Représentation de la commande u 62
4.20 Chemin de notre drone sous-marin par rapport au temps 62
4.21 erreur pour atteindre la profondeur désirée 63
4.22 surtension sur l'axe x 63
4.23 Chemin de notre drone sous-marin par rapport au temps sous l'effet du bruit 64
4.24 erreur pour atteindre la profondeur désirée 65
4.25 surtension sur l'axe x 65
4.26 Profondeur du robot sous marin 66
4.27 erreur pour atteindre la profondeur désirée 67
4.28 Angle de plongé 67
4.29 Angle de renforcement d'entré 68
4.30 Profondeur du robot sous marin 68
4.31 Angle de plongé 69
4.32 Angle de renforcement d'entré 69
33 Carte Arduino UNO R3 avec Microcontrôleur ATmega328 1

Liste des tableaux

4.1 Symboles utilisés pour décrire le 6 DDL 50

Abréviations

F_A :Poussé d'Archimède.
$\overrightarrow{F}_A$:Vecteur de poussée d'Archimède.
$f_s(s)$:La poupe d'entrée.
$G_\Theta(s)$: La fonction de transfert de la boucle de profondeur interne.
$G(s)$: La fonction de transfert de la boucle extérieure reliant l'angle désiré d'entrée (Θ_d) la profondeur du véhicule (z).
n : La vitesse en $min^{-}1$.
$\overrightarrow{P}$: vecteur de poids.
p :Roulis ou bien vitesse de rotation autour de l'axe Ox (en rad/s).
q :Taux de tangage ou bien vitesse de rotation autour de l'axe Oy (en rad/s). r :Taux de Lacet ou bien vitesse de rotation autour de l'axe Oz (en rad/s).
v :Balancement ou bien vitesse de linéaire sur l'axe Oy (en m/s).
v_1 :Le vecteur vitesse local de translation.
v_2 :Le vecteur vitesse local de rotation.
w :Soulèvement ou bien vitesse de linéaire sur l'axe Oz (en m/s).
x : Vecteur d'état transposé. ρ_{liq} :Masse volumique du liquide.
ρ_{corps} :Masse volumique du corps.
η :le vecteur d'état représentant la position et l'orientation du véhicule.
η_1 :le vecteur de position du centre de gravité.
η_2 :le vecteur d'altitude.
ϕ :Angle de roulis du repère Rv par rapport au repère R0.
θ :Angle de tangage du repère Rv par rapport au repère R0.
Ψ :Angle de lacet du repère Rv par rapport au repère R0.
$J_1(\eta_2)$:Matrice de passage de changement de base.
$\Theta(s)$:L'angle de pas de sortie de véhicule.
AUV : Autonomous Underwater Vehicle.
IDE : Integrated Development Environment (Environnement de développement intégré).
ROV : Remotely Operated underwater Vehicle.
VSS **:** Variable System (Système Variable Structure).

Introduction générale

La curiosité de l'homme l'a fait rechercher et explorer son environnement, depuis longtemps il a montré son intérêt pour la navigation sous-marine. Cependant des zones dans cet univers restent quasiment inaccessibles aux être-humains. De même, certaines populations aquatiques faune ou flore restent encore très mystérieuses. L'exploration des fonds sous-marins ainsi que les capacités furtives, dans le milieu sous-marin ont orienté l'homme vers la fabrication d'engins amphibies.

Ce n'est que lors de guerres que les premiers engins sous-marins ont été utilisés. Ces engins, appelés torpilles, ont pour objectif de détruire des navires à l'aide de charge explosive. Ce n'est qu'au XXe siècle que l'idée de torpilles sous-marines s'est peu à peu transformée en création de sous-marins habités et inhabités à des fins civiles. Les sous-marins inhabités sont appelés des drones sous-marins aussi des robots sous-marins. A travers l'histoire de la technologie sous-marine, il est évident que ces appareils ont gardé des fonctions militaires et civiles

Les robots marins peuvent être sous l'eau, mais aussi sur l'eau, ils sont utilisé comme plateforme pour observer, recueillir des informations sur l'environnement marin. À l'aube du développement des flottilles de véhicules, ils ont acquis désormais une certaine maturité industrielle et scientifique. Leur usage est largement répondu, principalement pour des applications pétrolières off-shore, pour les applications militaires (surveillance, déminage, etc.) et pour les applications scientifiques (océanographie, climatologie, etc.).

De plus, il est vrai que les drones pilotés à distance ou autonomes, permettent d'éloigner l'homme du danger du milieu sous-marin. Cet environnement est très hostile à l'homme notamment en matière de déplacement, de visibilité, de communication, de pression et de luminosité.

C'est dans ce contexte actuel des technologies sous-marines, que la problématique de ce mémoire s'inscrit. Comment concevoir un drone automatisé par des logiciels à la fois rapide, satisfaisant éventuellement à des applications civils, qui serait agile et stable. Cette dualité en ferait un atoût majeur regroupant la stabilité du système par sa commande et la simulation de tout.

En effet, la construction de tels systèmes mécaniques se base essentiellement sur trois axes : la fabrication mécanique, le concept électronique (qui constituent la partie hardware), et enfin la partie software qui résume la partie commande.

Largement appliquée, la commande robuste réserve une grande part des travaux de recherche dédiés aux systèmes mécaniques. Pour commander des tels systèmes perturbés, des techniques de commande à structure variables (par les modes glissants) sont efficaces à condition que les perturbations et les frottements vérifient une condition dite de recouvrement (matching condition).[1]

Cependant, et vu la conjoncture actuelle, l'objectif de notre travail s'est basé sur la partie commande tout en contribuant à l'élaboration des étapes nécessaires pour la construction d'un robot sous-marin.

Ainsi, ce mémoire est structuré en quatre chapitres comme suit :

Dans le chapitre 1, nous introduisons le sujet des drones sous-marins par un bref historique et un état de l'art des drones déjà construits. Pour finir on va mentionner une conception théorique concentrée sur la poussée d'Archimède.

Dans le chapitre 2, nous allons très bien détailler le matériel et les logiciels utilisés pour effectuer notre simulation. Dans cette partie nous allons citer la composition, les modèles et les types de la carte Arduino ainsi que les moteurs qu'on a choisis pour notre sous-marin.

Dans le chapitre 3, nous procédons à l'établissement du modèle non linéaire du drone. Nous présenterons dans le chapitre 3 la programmation et la simulation de notre robot. On va se focaliser sur ses deux points importants, tel qu'on va donner une explication approfondie sur le contrôle des moteurs dans notre programme établi sur l'IDE d'Arduino, leurs simulations sur Proteus, et le modèle de notre sous-marin en 3D établi à travers Autocad en détaillant leurs fonctionnements en réalité.

La partie commande fera part du chapitre 4. En effet, après avoir modélisé le comportement cinématique du drone, nous allons développer des lois de commande basées sur la technique des modes glissants. L'objectif de la commande est de maintenir le drone sur une profondeur bien déterminée en assurant la stabilité et la robustesse contre toute perturbation d'une part. Et d'autre part, assurer une poursuite d'une trajectoire bien déterminée. Par ailleurs, et afin de valider les lois de commandes développées dans le cadre de ce mémoire et d'évaluer leurs performances et de pouvoir les comparer, nous avons effectués des simulations sous la conduite de ces lois de commande.

Enfin, une conclusion générale clôturera notre travail.

Chapitre 1

Généralités sur la robotique sous-marine

Introduction

Dans un premier chapitre, nous allons introduire des généralités sur les sous-marins, en commençant par un bref historique, en parlant des différents types qui existent d'après leurs domaines d'utilisation, et en donnant une vision sur les futures modèles. Nous terminons par une explication du principe de la poussée d'Archimède.

Historique

L'histoire des véhicules sous-marins n'est pas récente. On peut citer par ordre chronologique l'évolution de la recherche dans ce domaine. On rapporte les premières tentatives à la mise en œuvre pratique de cette idée dès l'époque d'Alexandre le Grand qui, d'après Aristote, avait développé un submersible primitif pour des missions de reconnaissance en 322 av. J.-C. Une machine similaire avait été développée en Chine vers 200 av. J.-C [2].

Durant l'ère moderne, l'Anglais William Bourne a effectué un premier pas vers le concept de véhicule sous-marin en 1578, en concevant un modèle étanche pouvant être alimenté en oxygène. Mais ses idées ne dépassèrent pas le stade de la conception. Entre 1620 et 1624, le hollandais Cornelis Van Drebbel aurait navigué à quelques mètres sous les eaux de la Tamise. Il finira en 1664 par proposer un premier véhicule sous-marin avançant à l'aide de 12 rameurs équipés de rames spéciales. Ayant une forme ovoïde, il a pu être actionné de façon à réaliser des mouvements verticaux. Il a même été testé expérimentalement [2].

En 1775, le premier véhicule sous-marin américain 'Turtle' représenté dans la figure (1.1) a été présenté par David Bushnell et son frère. Construit en acier, il pouvait se déplacer de façon autonome. C'est le premier à être, d'une part équipé d'hélices pour sa propulsion et d'autre part d'une valve pour l'immersion et la remontée en surface. Il constitua le premier submersible autonome engagé dans un combat naval [2].

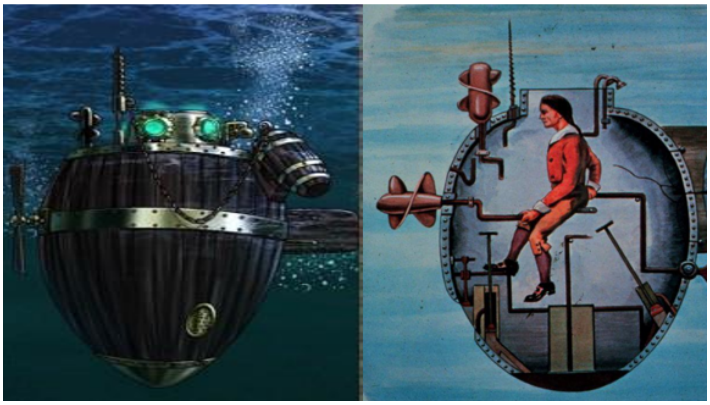

FIGURE 1.1 – Sous marin de bushnell « Turtle »
[3]

Robert Fulton construit le Nautilus représenté dans la figure (1.2), en acier recouvert de cuivre. Long de 6, 50 m, il est propulsé par une hélice actionnée à la main par les trois membres d'équipage. Il est équipé d'une charge explosive qu'il doit fixer sous les navires ennemis et déclencher à distance (difficile dans la pratique). Fulton propose son invention à la France puis à la Grande-Bretagne qui la refusent tour à tour. En 1811, le Nautile sous-marin des frères Coëssin, construit en bois et propulsé par quatre rameurs, est assemblé et testé au Havre. Ses nombreux défauts font abandonner le projet [3].

FIGURE 1.2 – Nautilus en 1797 en acier
[3]

Depuis cette époque, les véhicules sous-marins ont fortement évolué du point de vue de leur source de puissance. La recherche visait à se passer de la propulsion humaine et à exploiter d'autres sources d'énergie. Le plongeur de la Marine Nationale Française, lancé en 1863, est le premier sous-marin équipé d'un moteur à air comprimé et de 23 réservoirs de 180 PSI de pression (12 bar) [4].

Le premier sous-marin réellement opérationnel est le Gymnote de 1887 (1.3), construit par les Français Henri Dupuy de Lôme et Gustave Zédé. Long de 17 m, il est propulsé par un moteur électrique de 50 chevaux, atteint 8 nœuds en surface, 4 en plongée. Il est manœuvré par un équipage de cinq hommes. Il est armé de deux torpilles, et son rayon d'action est de 65 milles (en surface, mais seulement le tiers en plongée) [3].

En 1899, les deux sous-marins militaires français, successeurs du Gustave-Zédé, Le Français et l'Algérien, sont construits grâce aux fonds récoltés par une souscription nationale organisée par le journal Le Matin (France) À la fin du *XIX* ème siècle, le perfectionnement de la torpille conduit au développement des torpilleurs puis, en réaction, des contre-torpilleurs, alors que les sous-marins restent difficiles à détecter et à détruire. On conçoit alors le submersible : un torpilleur, assez marin pour naviguer en surface, en haute mer à distance importante de sa base, et qui ne plonge que pour le combat [3].

En 1904, l'ingénieur français Maxime Laubœuf construit le Narval, équipé d'un périscope et de ballasts externes, il obtient la faveur de la marine de l'époque. C'est le premier sous-marin équipé d'une propulsion mixte : machine à vapeur en surface, moteur électrique en plongée. Tous les modèles ultérieurs suivront cette conception fondamentale, jusqu'aux sous-marins nucléaires [3].

De 1914 à 1918, les submersibles fonctionnant grâce à une propulsion Diesel-électrique peuvent être engagés en grand nombre durant la guerre. Une batterie d'accumulateurs alimente un moteur électrique de propulsion. Les batteries sont rechargées par une génératrice entraînée par un moteur diesel, utilisable en surface. En 1944, les Allemands améliorent le schnorchel (invention hollandaise), un tube à air permet aux U-Boots d'utiliser leur moteur diesel à faible profondeur d'immersion, évitant ainsi de venir en surface où ils sont très vulnérables [3].

À partir des années 1950, la propulsion nucléaire apparaît à bord des sous-marins, à la suite de l'USS Nautilus (SSN-571) de 1954. Leur source d'énergie, le réacteur nucléaire est indépendant de l'atmosphère terrestre, il permet aux submersibles de devenir de véritables sous-marins. Dès son origine, à la fin du *XIX*ème siècle, le terme « sous-marin » est employé pour qualifier les premiers navires pouvant plonger sous la mer mais à des profondeurs, autonomies et vitesses très faibles. Si les performances se sont améliorées progressivement c'est seulement à la fin de la Seconde Guerre Mondiale, notamment grâce au perfectionnement du schnorchel et aux progrès accomplis par les ingénieurs allemands dans le domaine de la propulsion électrique en plongée des sous-marins de la Kriegsmarine, qu'il est apparu nécessaire de différencier ces navires : ceux naviguant surtout en surface et pouvant accessoirement plonger, de ceux de nouvelle génération dont l'autonomie sous l'eau est telle qu'ils remontent très rarement à la surface. On requalifie alors les premiers de conception ancienne, souvent construits avant guerre, de « submersibles », et les seconds, de « sous-marins ». Actuellement, l'autonomie en plongée des sous-marins en service, même à propulsion classique, est telle que la qualification de « submersible » est inadaptée, et a fortiori pour les

sous-marins à propulsion nucléaire, dont l'autonomie en plongée est illimitée [3].

Ces véhicules, restant coûteux à l'époque, n'ont connu leur véritable essor qu'à partir des années 2000 et ont été utilisés pour plus d'applications civiles. Ils sont utilisés aussi bien pour la recherche scientifique sous-marine que pour l'implantation et l'entretien d'installations sous-marines ou immergées [3].

1.1 Les Applications

Les applications des véhicules sous-marins sont potentiellement très vastes. Elles touchent en particulier deux principaux secteurs qui sont comme suit :

1.1.1 Civil

On peut distinguer quatre applications principales pour les robots sous-marins nous les citons comme suit :

Transport commercial

Appelant aussi sous-marins de ravitaillement utilisés pour transporter des cargaisons .Ce type est construit pendant la première guerre mondiale les premiers et les seuls deux « Cargos sous-marin » produisaient par l'Allemagne leurs avantage est de traverser sous la calotte glaciaire ainsi que leurs capacités [5].

Recherche océanographique

Afin d'explorer les grandes profondeurs, leurs missions typiques incluent l'observation, la collecte d'échantillons et les mesures mais ils peuvent aussi être affrétés pour des missions différentes comme l'intervention sur les épaves afin d'effectuer l'identification de vieilles épaves comme pour le Titanic, inspection pour la lutte anti-pollution ou en cas de litiges comme avec le pétrolier Prestige ou l'assistance à d'autres sous-marins en difficulté .

Les sous-marins de sauvetages restent cependant l'apanage des forces militaires [6].

Tourisme

Permettant de découvrir les fonds sous-marins sans pratiquer la plongée sous-marine. Ils ont été construit par les suisses, lors de l'exposition nationale suisse de 1964 généralement de dimension modeste, dont la fonction est d'effectuer des plongées dans un but touristique [7].

Echanges commerciaux illégaux

A l'aide des narco sous-marins autopropulsé et semi-submersible artisanal, ils ne peuvent pas plonger. Ce sont donc plus des bateaux à la ligne de flottaison basse. Ces navires sont conçus de façon à avoir une faible signature radar, sonar, infrarouge et visuelle [8].

1.1.2 Militaire

Sauvetage

C'est l'apanage des forces militaires. Dans le but de recueillir l'équipage d'un sous-marin en perdition qui serait posé sur le fond. Depuis les années 1950, environ une soixantaine de sous-marins de recherche a été construite, principalement aux États-Unis pour la recherche et le sauvetage militaire [9].

Attaques

Avec propulsion nucléaire ou classique. Leur mission est la destruction des forces de surface ou sous-marines ennemies par torpilles ou missiles antinavires et la frappe d'objectifs terrestres. Les sous-marins d'attaques sont les plus polyvalents et assurent plusieurs missions [9].

Lancements des missiles de croisière

Dans le but de lancer des missiles aérodynamiques à changement de milieu ce qui veut dire d'amener la charge explosive d'un environnement aqueux à un environnement au contact de l'air et inversement [9].

Lancements d'engins balistiques

Afin d'effectuer la dissuasion nucléaire ainsi que le lancement des missiles balistiques à charge nucléaire, en plongée . Les sous-marins lanceurs d'engins balistiques sont les plus imposants en activité et souvent aussi les plus silencieux. Si nécessaire, garantit à chaque instant la possibilité d'exécuter une frappe nucléaire qui serait ordonnée par le président de la République [10].

1.2 Les exemples

1.2.1 Véhicule sous-marin téléguidé « ROV »

Le ROV (Remotely Operated underwater Vehicle) c'est un petit robot sous-marin contrôlé à distance généralement filoguidé, contrairement au robot sous-marin autonome (AUV : Autonomous Underwater Vehicle). Comme leurs homologues aériens, ces drones

permettent une acquisition rapide et sécurisée d'informations globales ou précises, physicochimiques et visuelles sous forme numérique assez rapidement, à distance de l'opérateur et parfois "en masse". Certains ont une fonction de plateforme pouvant être équipée à la demande de préleveurs ou de divers capteurs [11].

FIGURE 1.3 – Véhicule sous-marin téléguidé « ROV »

1.2.2 Sous-marin Deutchland SM U-155

C'est un sous-marin cargo allemand, apparut pendant la première guère mondiale et mis en service en 1916 avec un équipage de 29 hommes .Sa capacité d'emport est 700 tonnes avec une longueur de 68.5 m. Il contient deux moteurs Diesel 400 Chevaux 6 cylindres ainsi que deux moteurs électriques de 375 Chevaux en plongée, en effet il pouvait naviguer à 15 nœuds en surface et à 7 nœuds en plongé. En 1918 il a été pris par les Britanniques et démoli en 1921 [11].

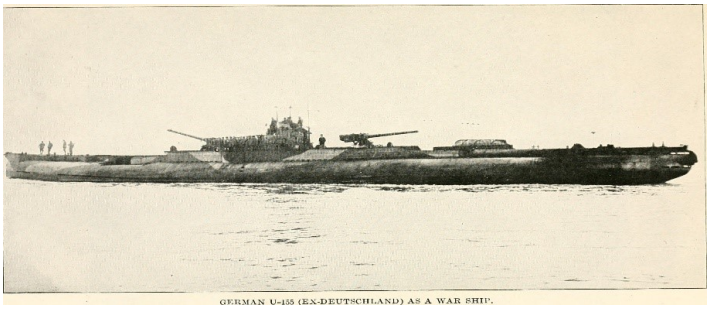

FIGURE 1.4 – Sous-marin Deutchland SM U-155
[10]

1.2.3 Triomphant S616

Un sous-marin nucléaire lanceur d'engins militaire français admit au service actif en 1997 avec deux équipages de 110 marins (16 officiers, 85 officiers mariniers et 12 quartiers maîtres et matelots). Son poids est 14.200 tonnes en plongée avec une longueur de 138 mètres et une largeur de $12,5$ mètres. Sa motorisation se pose sur 1 réacteur nucléaire, 1 groupe turbo-réducteur de $30,5$ MW avec une capacité d'immersion maximale 300 mètres. Ce sous-marin est équipé par un radar, un sonar multifonction, un sonar remorqué d'écoute

très basse fréquence, un système de transmissions par satellite et un système de combat SET ou SYCOBS. Son Armement stratégique contient 16 missiles stratégique M51 et son Armement d'autodéfense est de 4 tubes lance-torpilles de 533 mm pour torpilles F17 et pour missiles SM-39 EXOCET antinavire à changement de milieu [12].

FIGURE 1.5 – Triomphant S616
[12]

1.2.4 DRSV-1 Mystic

C'est un sous-marin de sauvetage à grande profondeur de la classe Mystic pouvant atteindre 1500 mètres anciennement utilisé par la marine de guerre des Etats Unis entre 1970 et 2008. Il fut construit par Lockheed pour le compte de la marine des Etat Unis lancé le 24 janvier 1970. En octobre 2014, le sous-marin a été offert au Naval Undersea Museum [13].

1.2.5 Sous-marin à propulsion humaine

Appelé aussi Omer est un club étudiant dont la mission est la conception et la fabrication de sous-marins à propulsion humaine. Les étudiants participent chaque année à des compétitions d'envergure internationale qui réunissent plusieurs équipes dans une épreuve de course. Mis sur pied en 1990, le club est un chef de file dans son domaine, détenant plus d'une douzaine de records de vitesse absolue. Dès 1996, le sous-marin Omer 2 a établi le premier record de vitesse et depuis, les performances du club ont toujours été croissantes [13].

1.3 Le Futur

Deux sous-marins de nouvelle génération devraient entrer en service d'ici les années 2030 avec la Royal Navy du Royaume-Uni et l'US Navy des États-Unis qui sont par ordre les classes Dreadnought et Columbia. Les futurs sous-marins s'inspirent de plus en plus des animaux afin d'optimiser leurs missions sous l'eau [14].

The **Royal Navy (RN)** la force de guerre navale du Royaume-Uni sollicite plusieurs spécialistes ont planché sur la prochaine génération d'engins sous-marins et ont imaginé quatre concepts biomimétiques formant un ensemble [14] :

1.3.1 Drone soluble dans l'eau

Parmi eux, un mini drone aquatique transporté puis largué par les sous-marins " anguilles ". Ce spécimen fabriqué à base de polymères solubles dans l'eau salée, pourra se dissoudre une fois sa mission accomplie et ne laisser aucune trace [14].

1.3.2 Espion anguille

Leur forme doit pouvoir se camoufler au milieu de la vie marin ce sous-marin aux allures d'anguille glissera dans l'eau avec un mouvement sinusoïdal, sur des centaines de kilomètres en éjectant des drones ou des capteurs, et en échangeant des informations entre eux par laser dans le but d'espionnage[14].

1.3.3 Vaisseau mère moitié raie moitié requin

Capable d'héberger 20 hommes son ossature sera imprimée en 3D avec des matériaux bien spécifiques permettant de supporter des profondeurs à 1000 mètres ou plus, et en absorbant des ondes ce qui permettra au vaisseau de devenir transparent sur les écrans radars [14].

1.3.4 Poissant volant d'attaque

Ce drone arme pourra remplacer les missiles et torpilles car il lui faudra être capable de plonger ou de ressortir de l'eau dès qu'il sera repéré et rester en même temps discret dans une zone proche de la surface, où il y a toujours beaucoup de perturbations [14].

1.4 Concept théorique

1.4.1 Le principe de la poussée d'Archimède

Tout corps solide complètement immergé dans un liquide en équilibre subit de la part du liquide une poussée verticale ascendante dont l'intensité est égale au poids du liquide déplacé, ce qu'on appelle la poussée d'Archimède.

$$F_A = P_{liq\ depl}$$

Le volume du liquide déplacé est égal au volume du corps V. Donc :

$$P_{liq\ depl} = m_{liq\ depl}\ g = \rho_{liq}\ V\ g$$

Finalement, on peut facilement calculer la poussée d'Archimède par la formule :

$$F_A = \rho_{liq}\ V\ g$$

Avec : ρ_{liq}. la masse volumique du liquide et V le volume du corps.

1.4.2 Modèle mathématique de la poussée d'Archimède

Soit un parallélépipède de base S et de hauteur h, plongé dans un liquide de masse volumique ρ_{liq}.

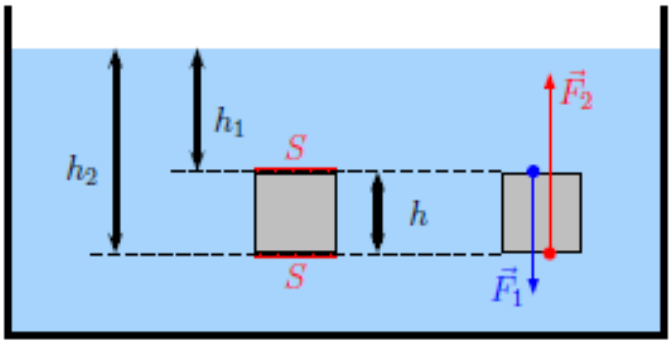

FIGURE 1.6 – Parallélépipède immergé dans un liquide

La face supérieure se trouve à une profondeur h_1, la face inférieure à une profondeur $h_2 = h_1 + h$.

La pression hydrostatique à la profondeur h_1 vaut : $p_1 = \rho_{liq}\ g\ h_1$. En h_2, elle vaut : $p_2 = \rho_{liq}\ g\ h_2$.

Le liquide exerce donc la force pressante ascendante $\vec{F}_2$ sur la face inférieure telle que :

$$\boxed{F_2 = p_2\ S = \rho_{liq}\ g\ h_2\ S}$$

De même : La norme de la force pressante descendante $\vec{F}_1$ exercée par le liquide sur la face supérieure vaut :

$$\boxed{F_1 = p_1\ S = \rho_{liq}\ g\ h_1\ S}$$

Comme $F_2 > F_1$, le corps est soumis à une force résultante $\vec{F}_A = \vec{F}_1 + \vec{F}_2$ dirigée vers le haut :

$$\boxed{F_A = F_2 - F_1 = \rho_{liq}\ g\ S\ (h_2 - h_1)}$$

Or $h_2 - h_1 = h$ d'où :

$$F_A = F_2 - F_1 = \rho_{liq}\ g\ S\ h$$

Or $S\ h = V$ (volume du corps) d'où :

$$F_A = F_2 - F_1 = \rho_{liq}\ g\ V$$

On retrouve la formule de la poussée d'Archimède.On peut montrer que cette formule reste valable pour toutes autres formes que pourrait avoir le corps immergé.

Remarque : On ne doit pas considérer les forces pressantes sur les faces latérales, comme celles-ci se compensent mutuellement.

1.4.3 Quand-est-ce-qu'un solide flotte, nage ou coule ?

Un corps solide immergé dans un liquide en équilibre est soumis à deux forces verticales et de sens contraires : son poids $\overrightarrow{P}$ et la poussée d'Archimède $\overrightarrow{F}_A$.

Remarque : On suppose que le corps solide est homogène. Dans ce cas, son centre de gravité et son centre de poussée se confondent.

Trois cas peuvent se présenter :

1. Le poids est plus grand que la poussée d'Archimède. Le corps va descendre vers le bas.

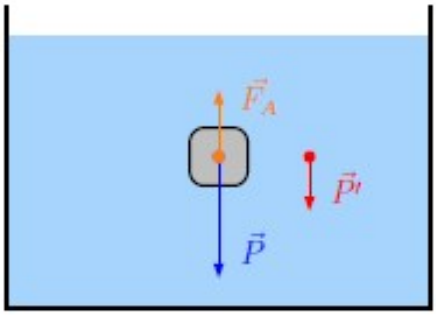

FIGURE 1.7 – Corps qui coule

$$P > F_A$$

Or nous avons les relations suivantes : $P = m\ g = \rho_{corps}\ V\ g$ et $F_A = \rho_{liq}\ g\ V$

$$\rho_{corps}\ V\ g > \rho_{liq}\ g\ V$$

Et donc :

$$\rho_{corps} > \rho_{liq}$$

Résultats : Si la masse volumique d'un corps est plus grande que la masse volumique du liquide dans lequel le corps est plongé, le corps va descendre vers le bas donc il va coulé.

2. Le poids est plus petit que la poussée d'Archimède. Le corps va monter vers le haut.

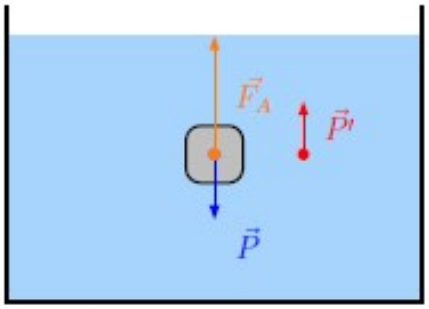

FIGURE 1.8 – Corps qui nage

$$P < F_A$$

Et donc :

$$\rho_{corps} < \rho_{liq}$$

Résultats : Si la masse volumique d'un corps est plus petite que la masse volumique du liquide dans lequel le corps est plongé, le corps va monter à la surface du liquide.

3. Le poids est égal à la poussée d'Archimède. Le corps va rester entre deux eaux..

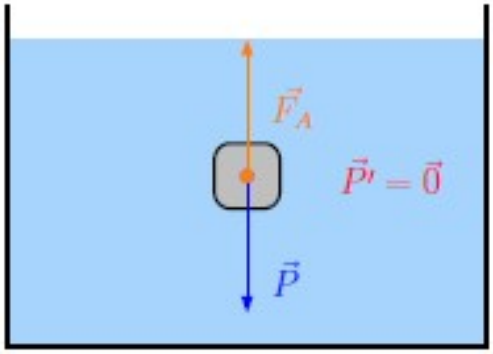

FIGURE 1.9 – Corps qui flotte

$$P = F_A$$

Et donc :

$$\rho_{corps} = \rho_{liq}$$

Résultats : Si la masse volumique d'un corps est égale à la masse volumique du liquide dans lequel le corps est immergé, le corps ne va ni descendre vers le bas, ni monter vers le haut donc il flotte.

Dans notre projet nous allons utiliser le principe de la poussée d'Archimède, elle est constante dans les sous-marins.

Conclusion

D'après ce chapitre, on s'est très bien informer sur les sous-marins en générale et les robots sous-marins spécifiquement en décrivant différents modèles dans le domaine civil et militaire, on a eu une vision sur le développement des futurs engins, et on a fini par une conception théorique sur la poussée d'Archimède.

Chapitre 2

Les composants et les logiciels utilisés.

Introduction

Dans ce deuxième chapitre, nous allons définir chaque détail sur la carte Arduino, les moteurs à courant continu et pas à pas avec la shield permettant de les contrôler. Nous expliquerons par la suite le principe de fonctionnement de ses derniers, nous mentionnerons leurs différents types et nous allons citer leurs points positifs et négatifs. Nous allons finir par la description de tous les logiciels qui ont lien avec la construction de notre drone sous-marin.

2.1 Carte Arduino

Arduino sont des circuits imprimés conçus et vendus par une compagnie Italienne. Ces cartes à 8 bits rendent les microcontrôleurs qui sont essentiellement de minuscules ordinateurs faciles à utiliser pour traiter les données beaucoup plus rapidement, ils sont assez bons pour la plupart des projets visant à contrôler les caméras d'éclairage de motors ou même à construire un robot simple, pour les applications IoT, les environnements portables, l'impression 3D et les environnements embarqués, aussi il existe un processeur Arduino plus puissant qui est le Wi-Fi. Après cela, il suffit d'acheter chez eux leurs innombrables sociétés tierces qui font leur propre variation des conceptions matérielles Arduino, mais ils ne peuvent pas les appeler arduinos, fonctionnellement ce sont les mêmes appelés les Shields qui permettent de contrôler les moteurs et les servos sans avoir à concevoir des circuits de commande de moteur.

Le logiciel est également open-source et se développe grâce aux contributions des utilisateurs du monde entier. Toutes les cartes Arduino sont entièrement open-source, permettant aux utilisateurs de les construire de manière indépendante et de les adapter éventuellement à leurs besoins particuliers.

L'Arduino dispose d'une large communauté de support et d'un ensemble complet de bibliothèques de support et de «boucliers» complémentaires matériels (par exemple, on peut facilement rendre notre Arduino sans fil avec le bouclier Wixel), ce qui en fait une excellente

plateforme d'introduction pour l'électronique embarquée.

L'intérêt principal des cartes Arduino est leur facilité de mise en œuvre. Un environnement de développement (IDE), s'appuyant sur des outils open-source, est fourni. En outre, le chargement du programme écrit dans cet IDE et sa compilation dans la mémoire du microcontrôleur, se fait très simplement par port USB. Enfin, beaucoup de bibliothèques de fonctions sont également fournies pour l'exploitation des entrées-sorties courantes : E/S TOR, génération de signaux PWM ...etc. Il existe plusieurs modèles d'Arduino : Nano, Mega, Uno, Leonardo ...ect.

2.2 La carte Arduino UNO

L'Arduino Uno est une carte microcontrôleur basée sur ATmega328P 8 bits. Cette carte dispose de 14 broches d'entrée / sortie numériques (dont 6 peuvent être utilisées comme sorties PWM et 6 peuvent être utilisées comme entrées analogiques), un résonateur 16 MHz, une connexion USB, une prise d'alimentation, une programmation de système en circuit (ICSP) en-tête et un bouton de réinitialisation. Il contient tout le nécessaire pour prendre en charge le microcontrôleur connecté simplement à un ordinateur avec un câble USB ou alimenté avec un adaptateur CA / CC ou une batterie pour commencer [15].

FIGURE 2.1 – Carte Arduino Uno
[15]

L'ATmega328 est un microcontrôleur Microchip 8 bits basé sur AVR RISC combine une mémoire flash ISP de 32 Ko avec des capacités de lecture en écriture, 1 Ko EEPROM, 2 Ko SRAM, 23 lignes d'E / S à usage général, 32 registres de travail à usage général, trois minuteries flexibles / compteurs avec modes de comparaison, interruptions internes et externes, USART programmable en série, interface série à 2 fils orientée octets, port série SPI, conver-

tisseur A / N 6 canaux 10 bits (8 canaux dans les boîtiers TQFP et QFN / MLF) , minuterie de surveillance programmable avec oscillateur interne et cinq modes d'économie d'énergie sélectionnables par logiciel. L'appareil fonctionne entre 1, 8 et 5, 5 volts. Sa programmation peut être réalisée en langage C/C++ [16].

FIGURE 2.2 – Microcontrôleur ATmega328
[16]

Ce microcontrôleur auxiliaire possède son propre chargeur de démarrage USB, qui permet aux utilisateurs de chargé le programme en mémoire et de le reprogrammer en exécutant des instructions puissantes en un seul cycle d'horloge, l'appareil atteint des débits proches de 1 MIPS par MHz, équilibrant la consommation d'énergie et la vitesse de traitement. Cette carte charge le programme lorsque l'Arduino est mis sous tension, ou réinitialise après la programmation.

Ceci est la 3ème révision de l'Uno (R3), qui a un certain nombre de changements :

La puce du contrôleur USB est passée de ATmega8U2 (flash 8K) à ATmega16U2 (flash 16K). Cela n'augmente pas la mémoire flash ou la RAM disponible pour les croquis.

Trois nouvelles broches ont été ajoutées, qui sont toutes des doubles des broches précédentes. Les broches I2C (A4, A5) ont également été sorties sur le côté de la carte près d'AREF. Il y a une broche IOREF à côté de la broche de réinitialisation, qui est un double de la broche 5V.

Le bouton de réinitialisation se trouve maintenant à côté du connecteur USB, ce qui le rend plus accessible lorsqu'un bouclier est utilisé.

2.3 Les moteurs avec leurs drivers

Les moteurs linéaires sont également disponibles, capables de produire un mouvement de chemise continu. Il existe essentiellement trois types de moteurs électriques conventionnels disponibles : les moteurs à courant alternatif, les moteurs à courant continu et les moteurs pas à pas.

Les moteurs à courant alternatif sont généralement utilisés dans les applications industrielles monophasées ou multi phases de haute puissance où un couple et une vitesse de rotation constante sont nécessaires pour contrôler de grandes charges telles que des ventilateurs ou des pompes.

Dans ce projet, nous examinerons uniquement les moteurs à courant continu avec leurs drivers L298D et les moteurs pas à pas légers simples qui sont utilisés dans de nombreux types de circuits électroniques, de contrôle de position, de microprocesseur, de PIC et de type robotique.

2.3.1 Moteur DC

Les moteurs électriques à courant continu sont des actionneurs continus qui convertissent l'énergie électrique en énergie mécanique. Ce moteur y parvient en produisant une rotation angulaire continue qui peut être utilisée pour faire tourner des pompes, des ventilateurs, des compresseurs, des roues, etc.

Un moteur à courant continu à balais conventionnel se compose essentiellement de deux parties, le corps fixe du moteur appelé le « Stator » et la partie interne qui tourne en mouvement appelé le « Rotor » ou «Armature» pour les machines à courant continu. Le résultat est qu'il existe essentiellement trois types de moteurs à courant continu disponibles : moteur à courant continu « C.C », moteur « Brushless », et « Servo » moteur.

Moteur à balais à courant continu

Ce type de moteurs produit un champ magnétique dans un rotor bobiné (la partie qui tourne) en faisant passer un courant électrique à travers un collecteur et un ensemble de balais de charbon, d'où le terme «brossé». Le champ magnétique des stators (la partie fixe) est produit en utilisant soit un enroulement de champ de stator bobiné, soit des aimants permanents. En général, les moteurs CC à balais sont bon marché, petits et faciles à contrôler [19].

Moteur sans balais Brushless

Ce type de moteur produit un champ magnétique dans le rotor en utilisant des aimants permanents qui y sont attachés et la commutation est réalisé électroniquement. Ils sont généralement plus petits mais plus chers que les moteurs à courant continu de type à balais conventionnels car ils utilisent des commutateurs à «effet Hall» dans le stator pour produire la séquence de rotation du champ de stator requise, mais ils ont de meilleures caractéristiques couple / vitesse, sont plus efficaces et fonctionnent plus longtemps durée de vie que les types brossés équivalents [19].

Servomoteur

Ce type de moteur est un moteur à courant continu à balais avec une forme de commande de rétroaction de position connectée à l'arbre du rotor. Ils sont connectés et contrôlés par un contrôleur de type PWM et sont principalement utilisés dans les systèmes de contrôle de position et les modèles radiocommandés [19].

Caractéristiques des moteurs CC

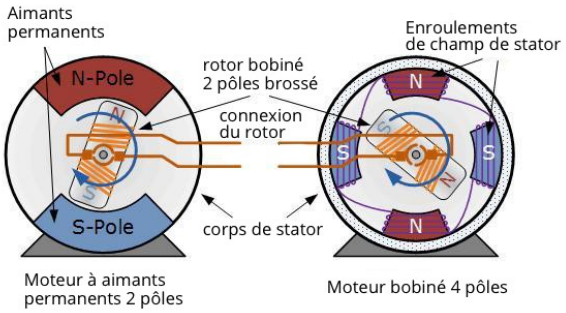

FIGURE 2.3 – Moteur à courant-continu

Tous les moteurs à courant continu normaux ont des caractéristiques presque linéaires, leur vitesse de rotation étant déterminée par la tension continue appliquée, et leur couple de sortie étant déterminé par le courant circulant dans les enroulements du moteur. La vitesse de rotation de tout moteur à courant continu peut se varier de quelques tours par minute (tr / min) à plusieurs milliers de tours par minute, ce qui les rend adaptés aux applications électroniques, automobiles ou robotiques. Comme la vitesse de rotation d'un moteur à courant continu est proportionnelle à la tension à ses bornes, nous pouvons réguler cette tension aux bornes à l'aide d'un transistor. En les reliant à des boîtes de vitesses, ou des trains d'engrenages, leur vitesse de sortie peut être diminuée tout en augmentant en même temps le couple de sortie du moteur à grande vitesse.

L'utilisation d'aimants permanents donne au moteur à courant continu une bien meilleure caractéristique de vitesse / couple linéaire que les moteurs bobinés équivalents en raison du

champ magnétique permanent et parfois très fort, ce qui les rend plus adaptés à une utilisation dans les modèles, la robotique et les servos.

Le stator bobiné des moteurs est un circuit électro-aimant qui se compose de bobines électriques connectées dans une configuration circulaire, pour produire le pôle Nord requis puis un pôle Sud un pôle Nord etc., de type système de champ magnétique stationnaire pour la rotation, contrairement aux machines à courant alternatif dont le champ statorique tourne continuellement avec la fréquence appliquée. Le courant qui circule dans ces bobines de champ est appelé courant de champ du moteur [19].

Ces bobines électromagnétiques qui forment le champ statorique peuvent être connectées électriquement en série, en parallèle ou les deux ensembles (composées) avec l'armature du moteur. Un moteur à courant continu bobiné en série à ses enroulements de champ de stator connectés en série avec l'armature. De même, un moteur à courant continu à enroulement shunt a ses enroulements de champ de stator connectés en parallèle avec l'armature [19].

Lors de la sélection d'un moteur à courant continu à brosse sans noyau pour une application ou lors du développement d'un prototype motorisé, plusieurs principes de base de la physique du moteur doivent être pris en compte pour produire un système d'entraînement de précision sure, fonctionnel et suffisamment alimenté. De ce fait on s'est basé sur des formules, et des détails de calcul importants pour déterminer la puissance de sortie d'un moteur sans noyau, et les calculs théoriques à froid qui estiment les performances du moteur.

En physique, la puissance du mouvement linéaire est le produit de la force et de la distance par unité de temps [18] :

$$P = F.\frac{d}{t} \tag{2.1}$$

Puisque la vitesse est la distance dans le temps, l'équation devient [18] :

$$P = F.s \tag{2.2}$$

Dans le cas d'un mouvement de rotation, le calcul analogue de la puissance est le produit du couple et de la distance angulaire par unité de temps ou simplement le produit du couple et de la vitesse angulaire comme suit [18] :

$$P = M.\theta \tag{2.3}$$

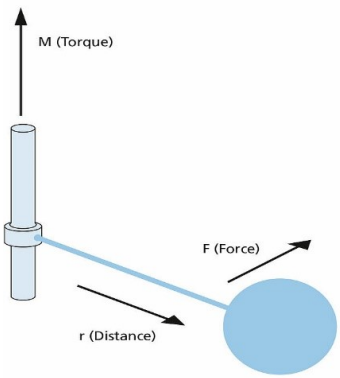

FIGURE 2.4 – Mécanisme d'un moteur à courant continu
[18]

P : Puissance en W (Watt) d : Distance en m (mètre) M : Couple en Nm (Newton par minute)	t : Temps en s (seconde) F : Force en N (Newton) θ : Vitesse angulaire (rad/sec)

Il est important de prendre en compte les unités lors du calcul de la puissance mécanique. Lors du calcul de la puissance, si la vitesse «n» est en min^{-1}, on doit le convertir en vitesse angulaire en unités de « rad/s». Ceci est accompli en multipliant la vitesse par un facteur de conversion unitaire de $\frac{2\pi}{60}$.

De plus, si le couple M est en « mNm », alors on doit le multiplier par « 10^{-3} » (diviser par 1000) pour convertir les unités en Nm à des fins de calcul ce qui donne :

$$\theta = n\,\frac{2\pi}{60} \tag{2.4}$$

$$P = M(10^{-3}) = n\,\frac{2\pi}{60} \tag{2.5}$$

Où n est la vitesse en min^{-1} et M est le couple en mNm.

Avantages et inconvénients des moteurs DC

Les avantages [19] :

⋆ L'encombrement réduit permit de gagner de l'espace notablement.

⋆ Les accélérations et les décélérations sont très douces et progressives.

⋆ Le réglage de la vitesse à couple et puissance constants est très facile, y-étant aussi la possibilité de la régler soigneusement même à un faible régime des tours. A' une vitesse élevée le moteur en courant continu peut modifier largement sa vitesse par désexcitation à puissance constante même jusqu'à trois fois la vitesse de base. En fait en réduisant le

diamètre primitive, la vitesse augmente et le couple diminue linéairement avec le flux, en maintenant ainsi la puissance constante. Cette caractéristique le rende non-remplaçable par un moteur équivalent en courant alternée qui n'ayant pas la même variation de le couple, ne peut pas garder la même performance vers la fin du cycle productive à les vitesses élevées.
⋆ Le rendement élevé du groupe convertisseur-moteur, se traduit en une épargne en énergie électrique qui est plus significatif si on adopte un convertisseur bidirectionnel marchant sur quatre quadrants en réalisant ainsi l'épargne d'énergie.
⋆ Par un programme simple et visant à l'entretien contrôlée, le moteur peut marcher pendant plusieurs années, et les économies de coûts seront considérables.
⋆ La propagation du moteur en courant continu dans le monde permit de retrouver les pièces de rechange très facilement d'ici la réduction des arrêts de la machine.
⋆ Absence des stresses qui soient liés à de surtensions sur l'enroulement du moteur et la vie des enroulements par rapport aux enroulements d'un moteur asynchrone sera majeure.

Les inconvénients [19] :

La commutation du moteur à balais nécessite la mise en œuvre d'un ensemble de pièces mécaniques (porte balais) permettant de réaliser la liaison par frottement entre les charbons et le collecteur. Il en découle que :
⋆ Plus la vitesse de rotation est élevée, plus la pression des balais doit augmenter pour rester en contact avec le collecteur donc plus le frottement est important.
⋆ Il est nécessaire de trouver la qualité de balais la mieux adaptée aux fonctionnements du moteur.
⋆ L'usure des balais, la dégradation de l'état de surface des collecteurs, et la pollution à la poussière de charbon sont des paramètres influents de manière négative sur le bon fonctionnement des moteurs courant continu.

2.3.2 Moteur Stepper (pas à pas) :

Les moteurs pas à pas sont des actionneurs électromécaniques qui convertissent un signal d'entrée numérique pulsé en un mouvement mécanique discret (incrémental) largement utilisé dans les applications de contrôle industriel. Un moteur pas à pas est un type de moteur synchrone sans balais en ce qu'il n'a pas d'induit avec un collecteur et des balais en carbone, mais un rotor composé de plusieurs, certains types ont des centaines de dents magnétiques permanentes et un stator avec des enroulements individuels.

Comme son nom l'indique, le moteur pas à pas ne tourne pas de manière continue comme un moteur à courant continu conventionnel, mais se déplace par «étapes» ou «incréments» discrets, l'angle de chaque mouvement de rotation ou étape dépendant du nombre de pôles du stator et du rotor dents du moteur pas à pas.

Généralement, les moteurs pas à pas ont un rotor interne avec un grand nombre de «dents» d'aimant permanent avec un certain nombre de «dents» d'électroaimant montées

sur le stator qui est la partie du moteur responsable de la création du champ magnétique avec lequel le rotor va s'aligner. Les principales caractéristiques du circuit statorique incluent son nombre de phases et de paires de pôles, ainsi que la configuration des fils. Les électroaimants des stators sont polarisés et dépolarisés séquentiellement, ce qui fait tourner le rotor un «pas» à la fois [20].

Un moteur pas à pas est un moteur électrique dont la caractéristique principale est que son arbre tourne. En appliquant la puissance à chaque ensemble de bobines à son tour, le rotor peut être amené à tourner ou à "avancer" d'une position à la suivante d'un angle déterminé par sa construction d'angle de pas, et en alimentant les bobines en séquence, le rotor produira un mouvement de rotation [20].

Cette caractéristique est obtenue grâce à la structure interne du moteur et permet de connaître la position angulaire exacte de l'arbre en comptant simplement le nombre de pas effectués, sans besoin de capteur, ce qui le rend également adapté à un large éventail d'applications [20].

En raison de leur fonctionnement pas à pas discret, les moteurs pas à pas peuvent facilement être tournés une fraction finie d'une rotation à la fois, telle que 1, 8, 3, 6, 7, 5 degrés, etc. Un moteur pas à pas effectue un tour complet 360° exactement 100 étapes. Pour obtenir le degré d'angle exact pour chaque tour possible on calcule :

$$A = N\ B \tag{2.6}$$

$$D = \frac{360}{A} \tag{2.7}$$

Tel que A est le nombre des positions possible, B le nombre de Bobine et N le nombre de dents. Le D est le degré de chaque tour. Un moteur pas à pas est particulièrement bien adapté aux applications qui nécessitent un positionnement précis et une répétabilité avec une réponse rapide au démarrage, à l'arrêt, à l'inversion et au contrôle de la vitesse et une autre caractéristique clé du moteur pas à pas est sa capacité à maintenir la charge stable une fois que la position requise est atteint. Il existe trois types de base de moteur pas à pas, à aimant permanent, à réluctance variable, et hybride (une sorte de combinaison des deux).

Rotor à aimant permanent

Il s'aligne avec le champ magnétique généré par le circuit du stator. Cette solution garantit un bon couple ainsi qu'un couple de détente. Cela signifie que le moteur résistera, même si ce n'est pas très fortement, à un changement de position, que la bobine soit ou non sous tension. L'inconvénient de cette solution est qu'elle présente une vitesse et une résolution inférieures par rapport aux autres types [20].

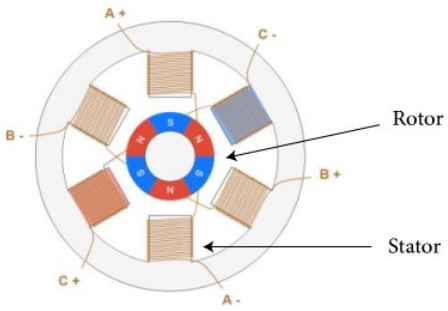

FIGURE 2.5 – Une coupe d'un moteur pas à pas à aimant permanent
[20]

Rotor à réluctance variable

Le rotor est constitué d'un noyau de fer et a une forme spécifique qui lui permet de s'aligner avec le champ magnétique. Avec cette solution, il est plus facile d'atteindre une vitesse et une résolution plus élevées, mais le couple qu'elle développe est souvent plus faible et il n'a pas de couple de détente [20].

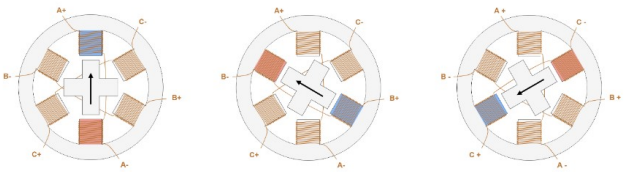

FIGURE 2.6 – Une coupe d'un moteur à rotor à réluctance variable
[20]

Rotor hybride

Ce type de rotor a une construction spécifique, et est un hybride entre les versions à aimant permanent et à réluctance variable. Le rotor a deux capuchons à dents alternées et est aimanté axialement. Cette configuration permet au moteur de bénéficier des avantages des versions à aimant permanent et à réluctance variable, en particulier une résolution, une vitesse et un couple élevés. Ces performances supérieures nécessitent une construction plus complexe, et donc un coût plus élevé. La figure 2.8 montre un exemple simplifié de la structure de ce moteur. Lorsque la bobine A est sous tension, une dent du capuchon N-magnétisé s'aligne avec la dent S-magnétisée du stator. En même temps, en raison de la structure du rotor, la dent S-aimantée s'aligne avec la dent N-magnétisée du stator. Les vrais moteurs ont une structure plus complexe, avec un nombre de dents plus élevé que celui montré sur la photo, bien que le principe de fonctionnement du moteur pas à pas soit le même. Le nombre élevé de dents permet au moteur d'atteindre un petit pas, jusqu'à $0,9°$ [20].

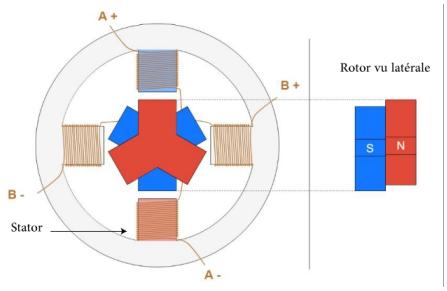

FIGURE 2.7 – Une coupe d'un moteur à rotor hybride
[20]

Caractéristique du moteur pas à pas

Dans un moteur pas à pas, plus il y a de dents de rotor et / ou de bobines de stator se traduirait par plus de contrôle et un angle de pas plus fin. Également en connectant les bobines électriques du moteur dans différentes configurations, des angles Full, Half et micro-step sont possibles. Cependant, pour réaliser un micro-pas à pas, le moteur pas à pas doit être entraîné par un courant (quasi) sinusoïdal coûteux à mettre en œuvre [20].

Il est également possible de contrôler la vitesse de rotation d'un moteur pas à pas en modifiant le délai entre les impulsions numériques appliquées aux bobines (la fréquence), plus le délai est long, plus la vitesse est lente pour un tour complet. En appliquant un nombre fixe d'impulsions au moteur, l'arbre du moteur tournera d'un angle donné [20].

L'avantage d'utiliser une impulsion temporisée est qu'il n'y aurait pas besoin de toute forme de rétroaction supplémentaire car en comptant le nombre d'impulsions données au moteur, la position finale du rotor sera connue avec précision. Cette réponse à un nombre défini d'impulsions d'entrée numériques permet au moteur pas à pas de fonctionner dans un «système en boucle ouverte», ce qui le rend à la fois plus facile et moins coûteux à contrôler [20].

Avec le contrôle de tension, le pilote ne règle que la tension aux bornes de l'enroulement. Le couple développé et la vitesse à laquelle les étapes sont exécutées dépendent uniquement des caractéristiques du moteur et de la charge [20].

Les pilotes de contrôle actuels sont plus avancés, car ils régulent le courant circulant dans la bobine active afin d'avoir un meilleur contrôle sur le couple produit, et donc le comportement dynamique de l'ensemble du système [20].

Avantages et inconvénients de stepper

Les avantages :

⋆ En raison de leur structure interne, les moteurs pas à pas ne nécessitent pas de capteur pour détecter la position du moteur. Puisque le moteur se déplace en effectuant des «étapes», en comptant simplement ces étapes, vous pouvez obtenir la position du moteur à un instant donné.

∗ De plus, le contrôle du moteur pas à pas est assez simple. Le moteur a besoin d'un pilote, mais n'a pas besoin de calculs ou de réglages complexes pour fonctionner correctement. En général, l'effort de commande est inférieur par rapport aux autres moteurs. Avec le micropas, vous pouvez atteindre une précision de position élevée, jusqu'à environ $0,007°$.
⋆ Les moteurs pas à pas offrent un bon couple à basse vitesse, sont parfaits pour maintenir la position et ont également tendance à avoir une longue durée de vie.

Les inconvénients :
⋆ Ils peuvent manquer une étape si le couple de charge est trop élevé. Cela a un impact négatif sur la commande, car il n'y a aucun moyen de connaître la position réelle du moteur. L'utilisation de micro pas rend les moteurs pas à pas encore plus susceptibles de rencontrer ce problème.
⋆ Ces moteurs drainent toujours le courant maximum même lorsqu'ils sont à l'arrêt, ce qui aggrave l'efficacité et peut provoquer une surchauffe.
⋆ Les moteurs pas à pas ont un faible couple et deviennent assez bruyants à haute vitesse.
⋆ Enfin, les moteurs pas à pas ont une faible densité de puissance et un faible rapport couple / inertie.

2.4 Driver L293

Les L293 et L293D sont des pilotes demi-H quadruples à courant élevé. Ces périphériques sont conçus pour piloter une large baie des charges inductives telles que les relais, les solénoïdes, les moteurs pas à pas à courant continu et bipolaires, ainsi que d'autres charges haute tension. Toutes les entrées sont compatibles TTL et tolèrent jusqu'à 7 V [21].

Chaque sortie est un circuit de commande de totem-pole complet, avec un puits de transistor Darlington et un pseudo-Darlington la source. Les pilotes sont activés par paires, les pilotes 1 et 2 activés par 1.2EN et les pilotes 3 et 4 activés par 3.4EN. Lorsqu'une entrée d'activation est élevée, les pilotes associés sont activés et leurs sorties sont actives et en phase avec leurs entrées. Lorsque l'entrée d'activation est faible, ces pilotes sont désactivés et leurs sorties sont désactivées et activées l'état à haute impédance [21].

Avec les entrées de données appropriées, chaque paire de pilotes forme un full-H (ou bridge) réversible variateur adapté aux applications solénoïdes ou moteurs.

Sur le L293, des diodes de serrage de sortie haute vitesse externes doivent être utilisées pour la suppression inductive des transitoires. Sur le L293D, ces diodes sont intégrées pour réduire la complexité du système et la taille globale du système. Un terminal VCC1, séparé de VCC2, est fourni pour les entrées logiques afin de minimiser la dissipation de puissance de l'appareil. Les L293 et L293D sont caractérisés pour un fonctionnement de 0° C à 70° C.

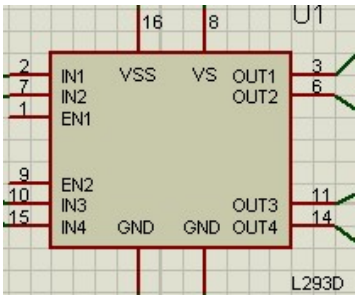

FIGURE 2.8 – L293DA sur logiciel Proteus

2.5 Les logiciels utilisés dans notre projet

2.5.1 Arduino IDE

L'environnement de développement intégré Arduino - ou logiciel Arduino (IDE) **Integrated Development Environment** facilement disponible pour Windows, Mac et Linux , il contient un éditeur de texte pour écrire du code, une zone de message, une console de texte, une barre d'outils avec des boutons pour les fonctions communes et une série de menus. Il se connecte au matériel Arduino et Genuino pour télécharger des programmes et communiquer avec eux.

Les programmes écrits à l'aide du logiciel Arduino (IDE) sont appelés esquisses et sont en languages C ou bien C ++. Ces esquisses sont écrites dans l'éditeur de texte et sont enregistrées avec l'extension de fichier **. ino**. L'éditeur a des fonctionnalités pour couper / coller et pour rechercher / remplacer du texte. La zone de message donne des informations lors de l'enregistrement et de l'exportation et affiche également les erreurs. La console affiche la sortie de texte par le logiciel Arduino (IDE), y compris les messages d'erreur complets et d'autres informations. Le coin inférieur droit de la fenêtre affiche la carte configurée et le port série. Les boutons de la barre d'outils vous permettent de vérifier et de télécharger des programmes, de créer, d'ouvrir et d'enregistrer des esquisses et d'ouvrir le moniteur série [22].

À partir de la version 1.0, les fichiers sont enregistrés avec une extension de fichier **. ino**. Les versions précédentes utilisent l'extension **. pde**. Vous pouvez toujours ouvrir les fichiers nommés **. pde** dans la version 1.0 et les versions ultérieures, le logiciel renommera automatiquement l'extension en **.ino**.

On permet de gérer les esquisses avec plusieurs fichiers (dont chacun apparaît dans son propre onglet). Il peut s'agir de fichiers de code Arduino normaux (sans extension visible), de fichiers C (extension **. c**), de fichiers C ++ (**. cpp**) ou de fichiers d'en-tête (**. h**). Sinon, le logiciel renommera automatiquement l'extension en **.ino**.

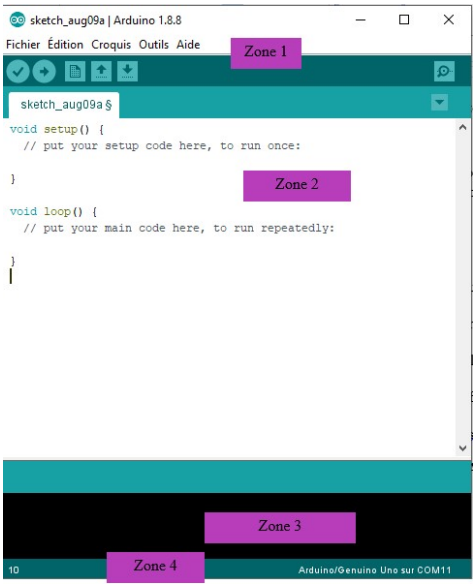

FIGURE 2.9 – Fenêtre principale d'un Arduino IDE

Zone 1 **:** Menu nous permet de choisir le modèle de la carte Arduino, compilation, téléversement du code et la lecture en série.

Zone 2 **:** zone conçut pour écrire le code Arduino.

Zone 3 **:** zone de débogage affiche les informations et les erreurs dans le code écrit après la vérification.

Zone 4 **:** zone affiche le type et le modèle sélectionnés de la carte Arduino ainsi que le port.

2.5.2 Proteus

Proteus est une suite logicielle destinée à l'électronique, développé par la société Labcenter Electronics. Il s'agit d'une suite logicielle contenant des schémas, des simulations ainsi que la conception de circuits imprimés.

Proteus VSM Simulation a été le premier outil de simulation de microcontrôleur basé sur des schémas au monde et est rapidement devenu une norme de facto pour l'enseignement des systèmes embarqués dans l'éducation [23].

Proteus répondent avec succès aux besoins commerciaux et éducatifs depuis plus de vingt-cinq ans. Les étudiants bénéficient d'une exposition à des outils de qualité professionnelle avec une interface utilisateur intuitive et une courbe d'apprentissage pratique rapide et amusant. De plus une solution logicielle permet aux formateurs de préparer et de réutiliser des laboratoires virtuels. Enfin les licences flexibles donnent la liberté pour les cours et les devoirs à accomplir n'importe où [23].

Proteus se compose de deux logiciels principaux :

ISIS

Le logiciel ISIS de Proteus est principalement connu pour éditer des schémas électriques. Par ailleurs, le logiciel permet également de simuler ces schémas en temps réel ce qui permet de déceler certaines erreurs dès l'étape de conception. Indirectement, les circuits électriques conçus grâce à ce logiciel peuvent être utilisé dans des documentations car le logiciel permet de contrôler la majorité de l'aspect graphique des circuits. L'outil de création de prototype virtuel permet de réduire les coûts matériels et logiciel lors de la conception d'un projet [23].

ARES

Le logiciel Proteus ARES dédié à la création de circuit imprimés et de routage qui complète parfaitement ISIS. Il a la fonction de visualiser la sortie dans une vue 3D du PCB conçu avec les composants, ce qui permet au concepteur de développer des dessins 2D pour le produit. Un schéma électrique réalisé sur ISIS peut alors être importé facilement sur ARES pour réaliser le PCB de la carte électronique. Bien que l'édition d'un circuit imprimé soit plus efficiente lorsqu'elle est réalisée manuellement, ce logiciel permet de placer automatiquement les composants et de réaliser le routage automatiquement [23].

Grace à des modules additionnels, ISIS est également capable de simuler le comportement d'un microcontrôleur (PIC, atmel, 8051, ARM, HC11 ...) et son interaction avec les composants qui l'entourent, c'est la raison dans laquelle on a choisi d'utiliser le ISIS de Proteus pour la conception de notre projet.

Voici la fenêtre principale d'un ISIS PROTEUS :

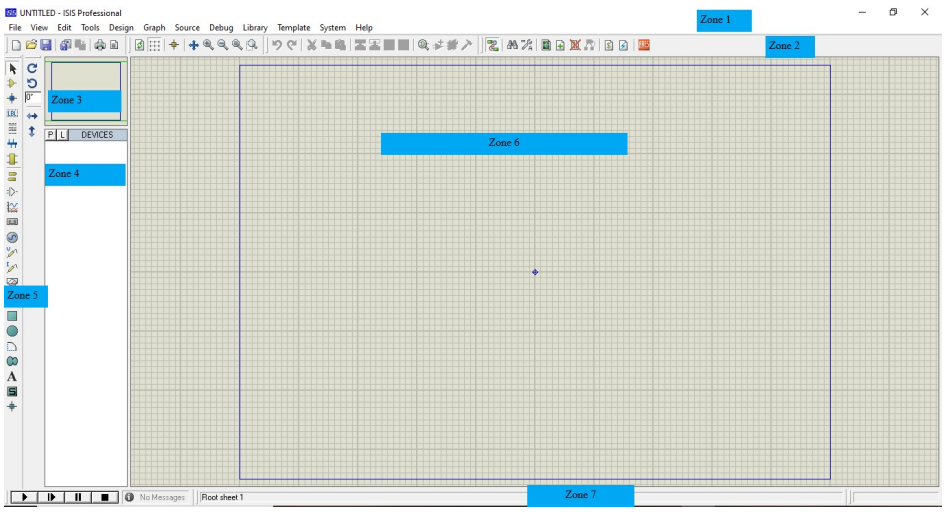

FIGURE 2.10 – Fenêtre principale d'un ISIS Proteus

Zone 1 **:** Barre de titre.

Zone 2 **:** Menu.

Zone 3 **:** Vue de l'ensemble des schémas sur la feuille.

Zone 4 **:** Les composants utilisée.

Zone 5 **:** Barre à outils.

Zone 6 **:** Feuille de travail.

Zone 7 **:** Les boutons de simulation.

2.5.3 Autocad

Autocad est un logiciel de conception assistée par ordinateur sur lequel les architectes, ingénieurs et professionnels de la construction s'appuient pour créer des dessins en 2 dimensions et 3 dimensions précis [25].

Ce logiciel nous permet de dessiner, annoter et concever une géométrie 2D et des modèles 3D avec des solides, des surfaces et des objets maillés. De plus Automatiser des tâches telles que la comparaison de dessins, l'ajout de blocs, la création de nomenclatures, etc. Enfin il est efficace pour personnaliser avec des applications complémentaires et des API [25].

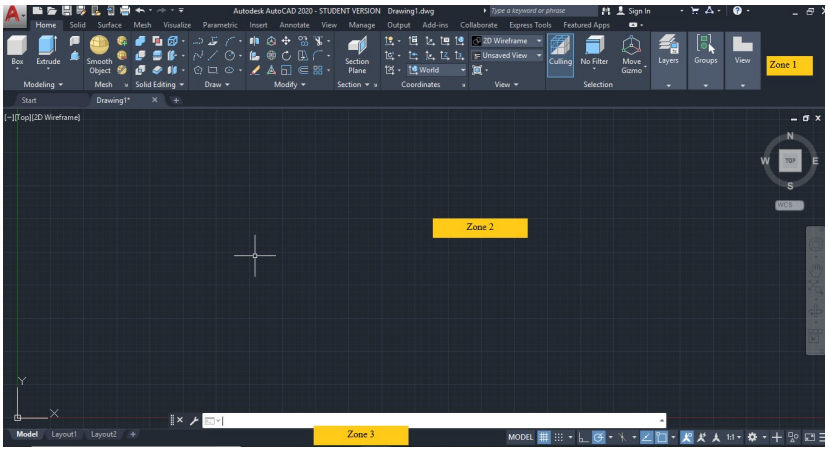

FIGURE 2.11 – Fenêtre principale d'Autocad

Zone 1 **:** Menu.

Zone 2 **:** Feuille de travail.

Zone 3 **:** Barre à outil.

2.6 Conclusion

D'après ce chapitre, on a assimilé les détails, les caractéristiques et le mode de fonctionnement de chaque type des composants ce qui nous permet de choisir bien spécifiquement la carte Arduino UNO, moteur à balais à courant continu et moteur pas à pas pour notre robot. Enfin on s'est familiarisé avec les logiciels nécessaires pour établir le codage, effectuer la simulation, et donner le modèle en 3 dimensions de notre sous-marin.

Chapitre 3

Programmation et Simulation

Introduction

Dans ce chapitre nous allons détailler les étapes pour établir notre robot, dans lequel on va utiliser trois moteurs à courant continu avec leurs drivers L293D, et un moteur pas à pas le tout sous contrôle d'une carte Arduino UNO. Nous expliquerons les étapes de l'enchainement entre la partie software et la partie hardware. Nous allons enfin proposer des modèles graphiques de notre engin.

3.1 Explication du programme sur Arduino IDE

D'abord, il est nécessaire d'inclure la bibliothèque « Stepper » fournie après son téléchargement, et ajouter dans le répertoire « Librairies » de l'IDE de notre Arduino. Celle-ci va nous permettre de communiquer avec le moteur pas à pas au moyen de fonctions simples d'usage.

Ensuite, nous avons consacré les broches digital du 8ème jusqu'à la 13ème de la carte Arduino UNO pour relier les borne des moteurs à courant continu.

Puis, nous avons défini le nombre de tours du moteur pas à pas, aussi nous avons initialisé ses quatre bornes reliées aux ports digital de la carte Arduino du 4ème jusqu'à le 7ème en ordre croissant.

Tous les programmes Arduino contienent la fonction setup () afin de définir les conditions initiales reliées au démarrage du programme. Dans notre code, cette dernière met en tête que les bornes des moteurs utilisés sont des producteurs en mentionnant l'instruction digitalwrite dans le but de donner les valeurs initiales de chaque borne des moteurs.

Par la suite grâce à la fonction "Loop", il est possible d'exécuter différentes opérations d'une manière répétée. Bien que cette fonction soit utile, le fait que "Loop" veut dire en français de manière répétitive cela pose un problème quand il s'agit d'exécuter des instructions toutes les X secondes.

Cette dernière fonction nous a permis d'organiser le nombre de tours de chaque moteur

à courant continu ainsi que le moteur pas à pas, aussi de contrôler son enchainement d'exécution d'après leurs utilité dans notre sous-marin, encore de changer leurs sens de rotation facilement et imposer des intervalles de temps pour assurer par la suite une bonne simulation.

Pour finir nous avons utilisé l'instruction "Delay" a plusieurs reprises, dans le but premièrement de garder le bon enchainement de fonctionnement entre les moteurs, et deuxièmement afin que le programme entier se met en pause plus rien ne va être exécuté, d'après les secondes que nous avons imposé sur notre matériel autrement dit le programme s'arrête en *Y* seconde.

3.2 Simulation sur Proteus

Le logiciel Proteus contient tous les composants dont nous avons besoin, pour faire le montage électrique de notre sous-marin en cliquant seulement sur sa bibliothèque.

Premièrement, nous avons commencé par l'ouverture d'une nouvelle feuille de dessin A2 sur Proteus, et nous avons choisi notre carte Arduino UNO R3, voilà ce que nous avons obtenu en première position (figure 3.1).

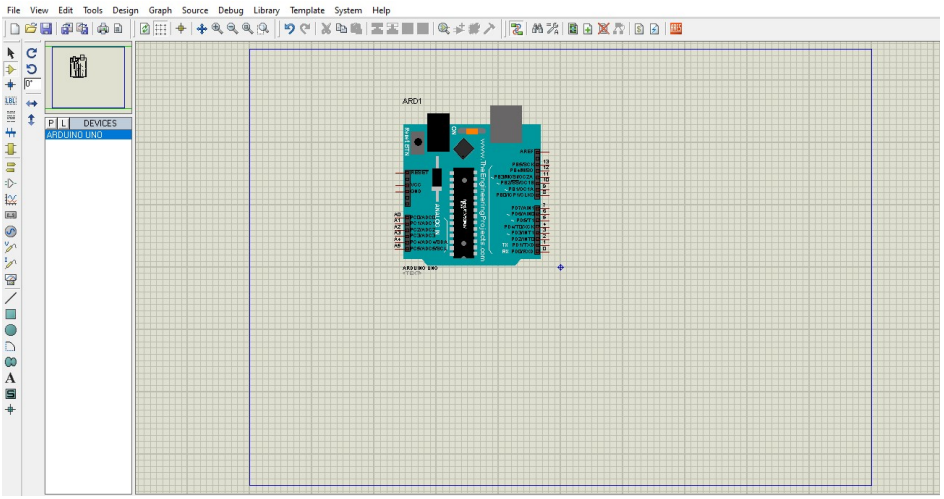

FIGURE 3.1 – Première étape de simulation

Deuxièmement, nous avons extrait les trois moteurs à courant continu (Simple DC Motors), en réglant le voltage de ses bornes à 12V ce qui nous a donné dans la figure 3.2.

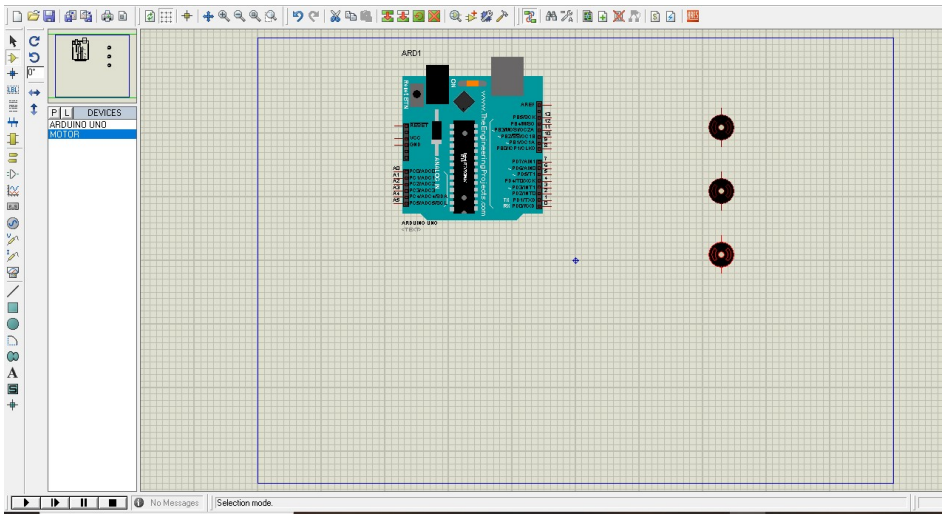

FIGURE 3.2 – Deuxième étape de simulation

Troisièmement, afin de commander les directions et les vitesses de ces trois derniers moteurs, nous avons choisi le driver L293D tel que ses bornes 3, 6, 11, 4 (Les output 1, 2, 3, 4) soient reliées avec deux moteurs à courant continu par ordre. Vu que ce dernier supporte deux moteurs seulement donc nous avons ajouté un autre driver pour le troisième moteur attaché avec ses broches 3 et 6 (output 1, 2).

L'alimentation de ces deux driver se fait à travers deux générateurs DC à 12 volt, branché sur les entrés 16 et 8, et le GND avec les entrées compatible. Le branchement de L293D ainsi que les moteurs avec la carte Arduino UNO se fait comme suit :

Dans le premier driver l'entrée IN1 broche numéro 2 est relié au port digital 13 de la carte Arduino.

⋆ L'entré IN2 broche numéro 7 est reliée au port digital 12 de la carte Arduino.

⋆ L'entré IN3 broche numéro 10 est reliée au port digital 11 de la carte Arduino.

⋆ L'entré IN4 broche numéro 15 est reliée au port digital 10 de la carte Arduino.

Dans le deuxième driver son entrée IN1 broche numéro 2 est reliée au port digital 9 d'Arduino.

⋆ L'entré IN2 broche numéro 7 est reliée au port digital 8 de la carte Arduino.

Donc le branchement des moteurs avec la carte Arduino Uno sur Proteus se fait comme suit :

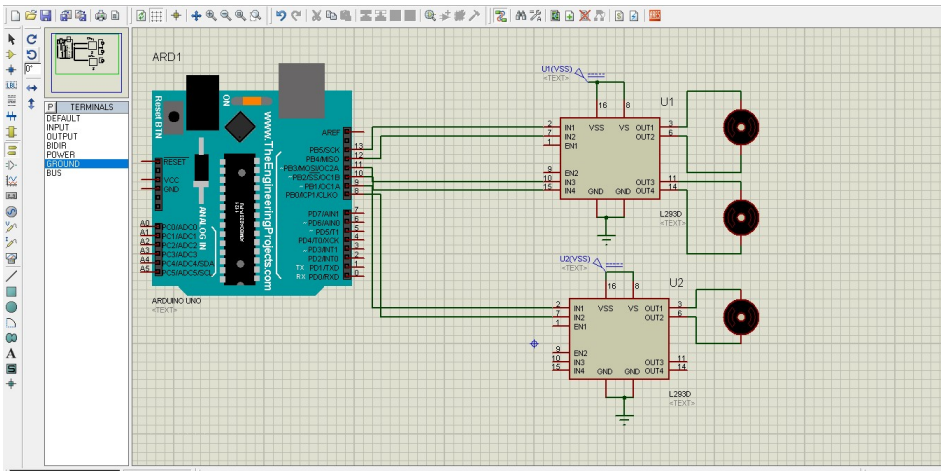

FIGURE 3.3 – Troisième étape de simulation

Pour compléter notre circuit, nous avons ajouté le moteur pas à pas de type bipolaire avec une tension de 12 volts, aussi l'angle entre chaque pas 90° sachant que 360° est l'angle maximal, avec une résistance de bobine 120ohm et son inductance 1000 mH. Ces quatre bornes sont attachées avec la carte Arduino UNO sur les broches digitales du 4ème au 7ème par ordre, comme le montre la figure 3.4 finale de notre circuit sans simulation.

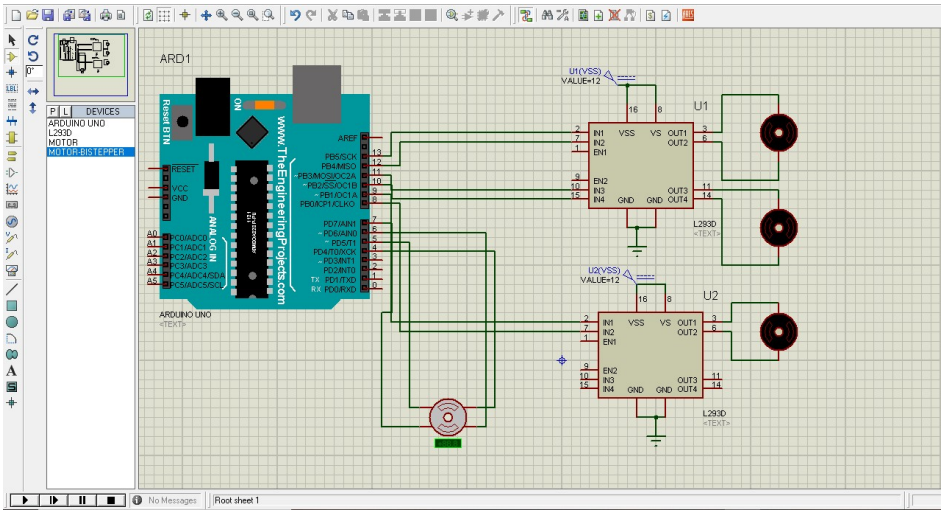

FIGURE 3.4 – Le montage électrique complet avant simulation

La dernière étape pour effectuer la simulation de notre circuit, est de télécharger le programme établi sur l'IDE de l'Arduino en faisant copier sur le lien **.hex** apparu dans sa zone de débogage qui contient , et coller sur la fenêtre apparu en cliquant sur la carte Arduino en Proteus .

3.3 L'harmonisation entre le programme et sa simulation

Le montage réalisé sur Proteus accomplie le programme saisie sur l'IDE d'Arduino. En visualisant l'exécution du code, ça nous a permis de gérer parfaitement la partie électrique de notre sous-marin selon nos consignes.

Après avoir télécharger le code sur le microcontrôleur ATmega328 intégré dans la carte Arduino UNO sur Proteus, nous avons visualisé le fonctionnement électrique de notre sous-marin, ce qui nous a permis d'approuver le bon enchaînement des deux partie hardware et software de notre robot.

Lorsque nous appuyons sur le bouton simulé du Proteus, tout d'abord le moteur pas à pas commence à tourner avec une précision de nombre des pas, ainsi que le temps exacte pour s'arrêter mentionné dans la fonction Loop du code. Vu ses performances intitulées alors on a préféré d'utiliser un moteur pas à pas afin de guider les étapes, pour atteindre les profondeurs désirées sur l'axe des Z dans un intervalle de temps donné, y compris sa vitesse de soulèvement, en changeant le nombre de pas, et en modifiant les instructions détaillées écrites sur l'IDE.

Une fois que ce moteur pas à pas termine sa boucle, un déclenchement du premier moteur à courant continu est planifié, tel que ce dernier est responsable principalement du déplacement de notre sous-marin, droitement sur l'axe des X avec une surtension u dans les fonds. En orientant la rotation du moteur, ce qui nous a donné la main pour l'adapter dans les deux sens sur l'axe des X ce qui induit que le sous-marin pourra marcher vers l'avant ou bien vers l'arrière autrement dit avancer ou reculer sous les eaux.

Dans le cas où nous voulons que notre sous-marin change sa direction du mouvement vers la gauche ou bien la droite, dans ce cas on active l'un des deux moteurs CC resté, consacré pour ça, et programmé sur l'IDE en s'adaptant avec la suite des instructions mentionnées dans la fonction Loop, de tel manière que le moteur responsable du mouvement principale de déplacement garde la même vitesse ce qui causera un petit balancement.

Enfin, dans le cas où nous voulons faire flotter notre sous-marin, ce qui veut dire un déplacement positif sur l'axe des Z, alors on arrête totalement le fonctionnement des moteurs à courant continu à travers une suite des instructions que nous ajoutons sur l'IDE, et de ce fait nous faisons tourner notre moteur à pas à pas dans le sens inverse.

3.4 Les modèles des sous-marins sur Autocad

3.4.1 Le modèle 1

En réalité, ce modèle sera relier à notre ordinateur avec une fibre optique dans le but d'automatiser le système à nos consignes désirées. Ce dernier est alimenté par une batterie de type Lypo à 12V, et caractérisé par une caméra 4K HD Action guidée par wifi de notre smartphone afin de partager les photos en temps réel en pleine exploration. Le volume d'eau dans les ballasts ne varient plus jusqu'à la prochaine immersion. Pour se déplacer verticalement sous l'eau ou bien sur l'axe des Z, le sous-marin utilise l'aileron situé à l'arrière du sous-marin piloté par un moteur pas à pas de type Nema17. Cet aileron peut également être appelé barre de plongée. Lorsqu'on fait tourner ce moteur, une vitesse de soulèvement va apparaitre, alors le sous-marin commence à descendre. L'eau fait ensuite pression sur le dessous de l'aileron à l'arrière. Le sous-marin pivote ensuite, tel que son assiette devient négative et il s'enfonce dans les profondeurs.

Pour remonter, on inverse le sens du stepper tel que les barres de plongées font le mouvement inverse. Ainsi, les pressions exercées par l'eau sont inverses. L'assiette du sous-marin devient alors positive ce qui fait qu'il remonte vers la surface.

Se pose ensuite le problème du déplacement horizontal sur l'axe des X. Afin de permettre notre sous-marin de changer de direction à gauche ou à droite, on propose de monter un moteur CC avec le gouvernail à l'arrière et pour la mise en marche faut placer deux autres moteurs CC avec chaqu'un des hélices

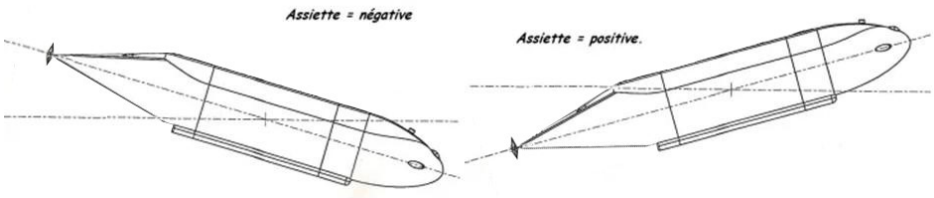

FIGURE 3.5 – Assiette positive et négative

3.4.2 Le modèle 2

En réalité ce deuxième modèle aussi, sera relié à notre ordinateur avec une fibre optique dans le but d'automatiser le système à nos consignes désirées tel que ce dernier est alimenté par une batterie de type Lypo à 12V, et caractérisé par une caméra 4K HD Action guidé par wifi de notre smartphone afin de partager les photos en temps réel en pleine exploration. Pour remplir le balist afin de faire descendre notre robot faut monter une vanne enchainé avec le moteur pas à pas contrôlé par la carte Arduino, ça nous permet de se déplacer verticalement sous l'eau exactement sur l'axe des Z du robot. Son immersion se pose sur le réglage des

degrés de rotation du stepper sur le code. Lorsqu'on fait tourner ce moteur, une vitesse de soulèvement va apparaitre, alors le sous-marin commence à descendre. L'eau fait ensuite pression (principe de la poussé d'Archimède) de ce fait le sous-marin s'enfonce dans les profondeurs.

Pour remonter, on inverse le sens du stepper tel que la vanne tourne en mouvement inverse, et se vide petit à petit. Ainsi, les pressions exercées par l'eau sont inverses donc il remonte vers la surface.

Se pose ensuite le problème du déplacement horizontal sur l'axe des X. Pour cela, on contrôle des trois hélices par trois moteurs CC, tel que l'hélice au milieu chargé du déplacement droit, et les autres hélices vont permettre au sous-marin de virer à gauche et à droite. Dans le cas où nous voulons améliorer la vitesse du déplacement de notre robot, on déclenche les trois moteurs à courant continu en même temps pendant le même intervalle de temps, ce qui engendra plus de puissance et plus de vitesse.

3.4.3 Les deux modèles sur Autocad

Le modèle 1

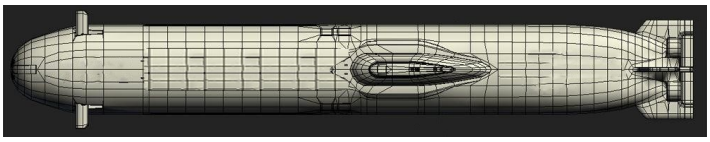

FIGURE 3.6 – Vue de dessus

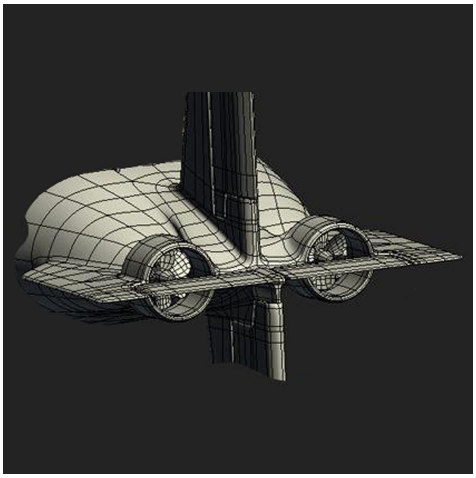

FIGURE 3.7 – Vue de derrière

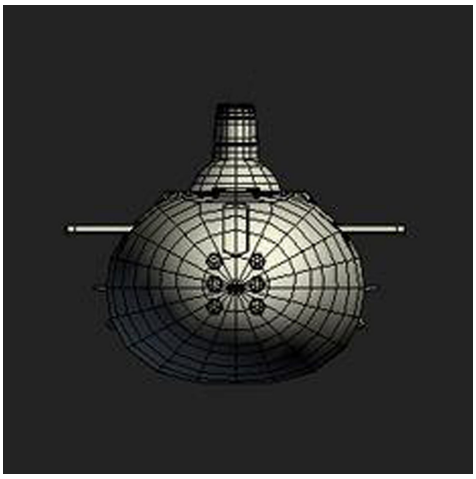

FIGURE 3.8 – Vue de face

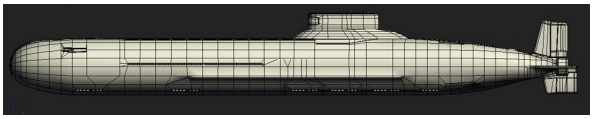

FIGURE 3.9 – Vue de coté

Le modèle 2

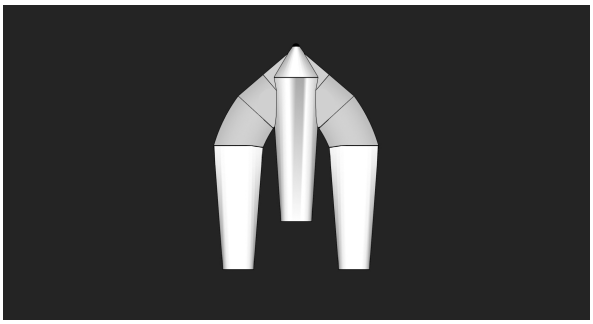

FIGURE 3.10 – Vue de dessus

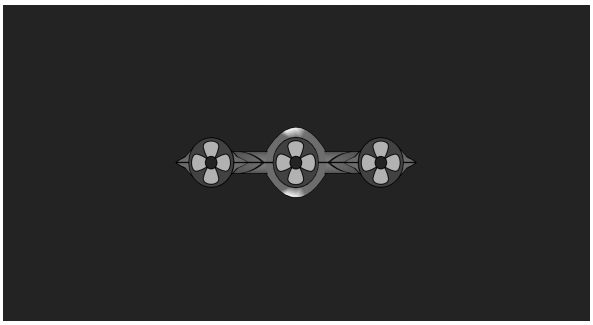

FIGURE 3.11 – Vue de derrière

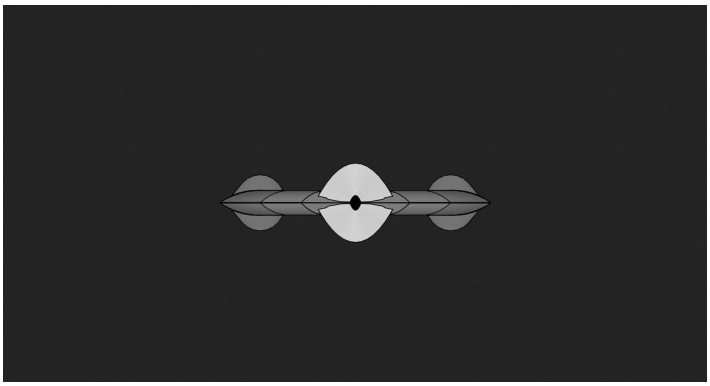

FIGURE 3.12 – Vue de face

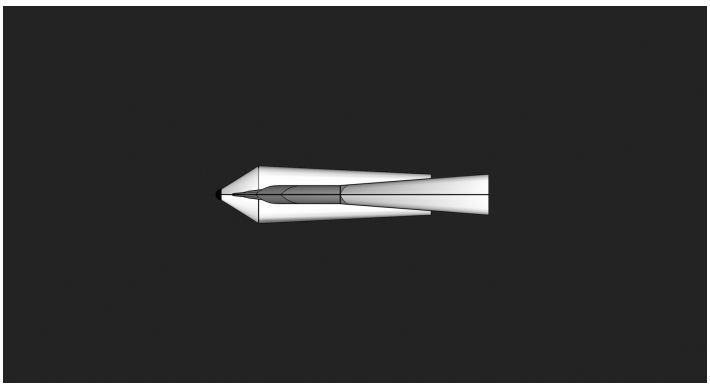

FIGURE 3.13 – Vue de coté

3.5 Conclusion

Dans ce chapitre, nous avons expliqué étape par étape notre code écrit sur l'IDE d'Arduino afin d'initialiser nos composants, permettant au microcontrôleur ATmega328 de cette carte, de connaître le nombre, et l'enchainement des tours exacte des moteurs utilisés à chaque déplacement dont en a besoin, ce qui nous a mené par la suite a avoir les résultats de notre exécution sur la simulation du circuit établi sur le logiciel Proteus à n'importe quel moment donné, et voir le fonctionnement de ces moteurs. Enfin nous avons exposé deux modèles automatisés sur le plan réel avec le même code, même circuit tel que nous avons bien détaillé leurs fonctionnements, dans le but d'éclaircir notre idée d'un drone sous-marin semi-autonome en utilisant une fibre optique dans le monde réel.

Chapitre 4

Commande non linéaire par les modes glissants Modélisation et Simulation sur Matlab

Introduction

Dans ce chapitre, la partie commandée est soumise à de fortes non linéarités et à des variations temporelles, dans ses conditions il faut concevoir des algorithmes de commande assurant la robustesse du comportement du processus vis-à-vis des incertitudes sur les paramètres et leurs variations. Nous commençons par une présentation des concepts généraux de la commande des systèmes à structure variable, ainsi que des notions générales sur la technique des modes glissants, et une modélisation des 6 degrés de liberté de notre robot. Nous concluons par l'analyse des résultats graphiques obtenus après la simulation sur Matlab.

4.1 Définition du mode glissant

Le phénomène de glissement consiste à amener la trajectoire d'état d'un système vers la surface de glissement, et de la faire commuter à l'aide d'une logique de commutation appropriée autour de celle-ci jusqu'au point d'équilibre. La dynamique de ce mode est caractérisée par la détermination de la surface de glissement $S(x) = 0$ [26].

4.2 Système à structure variable

Un système à structure variable en anglais "variable structure system VSS" est un système qui change de structure pendant le fonctionnement, il permet de choisir la fonction et la logique de commutation, par conséquent le système commute d'une structure à une autre à tout moment, ce qui induit l'apparition des différentes propriétés dans quelques structures. La commande VSS par mode glissement, sa trajectoire d'état est dirigée vers la surface de glissement appelant le mouvement par mouvement de glissement, et grâce à loi de commu-

tation la commande reste au voisinage de cette surface [26].

4.3 Objectif de la commande par mode glissant

La commande par mode glissant consiste à ramener la trajectoire d'état vers la surface de glissement, et de le faire évoluer dessus avec une certaine dynamique jusqu'au point d'équilibre. Cette commande simplifie la mise en œuvre, et la rend plus robuste par rapport aux incertitudes du système, et des perturbations externes entachant le processus [26].

4.4 Les notions de base de la commande par mode glissant

4.4.1 La modélisation mathématique de la commande à structure variable

Les équations différentielles de cette commande [25] :

$$x(t) = f(x,t) + B(x,t).V \tag{4.1}$$

x est le vecteur d'état écrit sous la forme $[x_1, x_2, x_3 \dots x_n]^T$.
f est le vecteur en fonction de x et t.
B est la matrice en fonctions de x et t (une matrice $m \times n$).
V est le vecteur commande de dimension m, dont chaque composante V_i subit une discontinuité sur une surface $S_i(x) = 0$.

La surface S peut s'écrire comme : $S(x) = 0$, On a $i = 1, 2 \dots m$ les composantes du vecteur de commande.
⋆ Si $S_i > 0$ alors $V_i(x)$ est positif.
⋆ Si $S_i < 0$ alors $V_i(x)$ est négatif.
Dans le cas multivariables le régime glissant s'effectue sur une variété $S(x) = 0$ de dimension $(n \times m)$.

4.4.2 La condition d'existence par mode glissant

Le mode de glissement existe lorsque les commutations ont lieu d'une façon continue entre les deux grandeurs U_{max} et U_{min}. Pour un système du deuxième ordre avec les grandeurs d'état x_{s_1} et x_{s_2} ce phénomène est démontré comme suit :

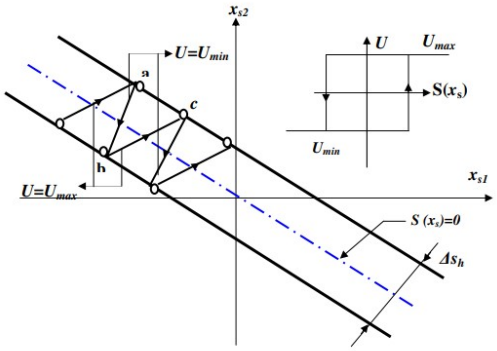

FIGURE 4.1 – Démonstration du mode de glissement

On considère d'abord une hystérésis sur la loi de commutation $S(x) = 0$, les commutations ont lieu sur les droites décalées parallèlement de $\pm\Delta S$. Une trajectoire avec $U = U_{max}$ touche au point "a" le seuil de basculement intérieur. Si avec $U = U_{min}$, la trajectoire est orientée vers l'intérieur de la zone de l'hystérésis, elle touche au point "b" le seuil de basculement supérieur où à lieu de commutation sur $U = U_{max}$. Si la trajectoire est de nouveau orientée vers l'intérieur, elle touchera le point "c" le seuil de basculement inferieur et ainsi de suite. Il y'a donc un mouvement continu à l'intérieur de la zone de l'hystérésis. Par conséquent la loi de commutation fait un mouvement infiniment petit autour de $S(x) = 0$ et le vecteur x suit une trajectoire qui respecte cette condition [26].

4.4.3 La surface du glissement

Si tout point de S est tel qu'il existe des trajectoires d'état hors de S le contenant alors la surface de commutation S est appelée surface de glissement. Le but d'un système de commande à structure variable est d'amener asymptotiquement l'état du système à partir d'une condition initiale quelconque $x(0) = x_0$ vers l'origine de l'espace d'état quand $t \longrightarrow \infty$ en respectant la condition de convergence.

Pour choisir la surface du glissement, commençons par considérer le système par la représentation d'état non linéaire suivante :

$$\begin{aligned} \dot{x}(t) &= f(x) + g(x).u \\ y(t) &= h(x) \end{aligned} \tag{4.2}$$

Le principal objectif de la commande par le mode du glissement est de maintenir la surface à zéro. Tel que cette surface est une équation différentielle linéaire avec $e(x) = 0$ sa solution unique.

Pour faire un choix correct des paramètres, ceci dépend à un problème de poursuite de trajectoire qui est équivalant à une linéarisation exacte de l'écart, en respectant la condition de convergence d'une variable vers sa valeur désirée.

L'équation générale pour la surface de glissement, qui assure la convergence d'une variable vers sa valeur désirée est donnée par [26] :

$$S(x) = (\frac{\partial}{\partial t} + \lambda_x)^{r-1} e(x) \tag{4.3}$$

⋆ $e(x)$ est l'ecart de la variable à régler, $e(x) = x_{ref} - x$.
⋆ λ_x est une constante positive qui intérpréte la bande passante du contrôle désiré.
⋆ r est le degré relatif, égal au nombre de fois qu'il fait dériver la sortie pour faire apparaitre la commande.
⋆ $S(x) = 0$ est une équation différentielle non linéaire dont l'unique solution est $e(x) = 0$.

Pour $r = 1$,

$$S(x) = e(x)$$

Pour $r = 2$,

$$S(x) = \dot{e}(x) + \lambda_x e(x)$$

4.5 Condition de convergence

Les conditions de convergence permettent aux dynamiques du système de converger vers les surfaces de glissement. Nous retenons de la littérature deux conditions, celles-ci correspondent au mode de convergence de l'état du système.

4.6 Fonction directe de la commutation

Il s'agit de donner à la surface une dynamique convergente vers zéro. Elle est donnée par :

$$\dot{S}(x).S(x) < 0 \tag{4.4}$$

4.7 Définition et expression analytique de la commande

Après le choix de surface de glissement, ainsi que le critère de convergence est prouvé, on va déterminer la commande pour ramener la variable à contrôler vers la surface et ensuite vers son point d'équilibre en maintenant la condition d'existence des modes glissants. Une des hypothèses essentielles dans la conception des systèmes à structure variable contrôlés par les modes glissants, est que la commande doit commuter entre V_{max} et V_{min} instantanément avec fréquence infinie, en fonction du signe de la surface de glissement .Dans ce cas, des oscillations de très haute fréquence appelées « broutement » ou « Chattering » apparaissent dans le mode de glissement [26].

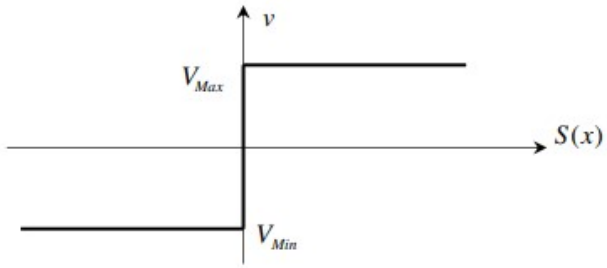

FIGURE 4.2 – Commande appliquée aux systèmes à structure variable

La surface de glissement se détermine en fonction du système et des performances désirées, indépendamment de la commande, et l'obtention du régime glissant supposerait la commande discontinue. De ce fait, si cette commande est indispensable, elle n'empêche nullement, au contraire, qu'une partie continue lui soit adjointe pour diminuer l'amplitude de la discontinuité.

Alors, la structure d'un contrôleur comporte deux parties, une première concernant la linéarisation exacte et une deuxième stabilisante qui est utilisée pour rejeter les perturbations extérieures dans la commande par mode glissant. On prend :

$$V(t) = V_{eq}(t) + V_N \tag{4.5}$$

$V_{eq}(t)$ est calculé en reconnaissant que le comportement du système durant le mode de glissement est décrit par : $\dot{S}(x) = \dot{e}(x) = 0$
$V_N(t)$ est un terme introduit pour satisfaire la condition de convergence décrite dans (4.4). Il détermine ainsi le comportement dynamique du système durant le mode de convergence. Donc cette commande est garantie l'attractivité de la variable à contrôler vers la surface de glissement. Le terme V_N est donné par :

$$V_N = \dot{S}(x) = -K.sign(S(x)) \tag{4.6}$$

Où $K > 0$ et $sign(S(x))$ est la fonction définie par :

$$sign(S(x)) = \begin{cases} +1 & \text{si } S > 0 \\ -1 & \text{si } S < 0 \end{cases} \tag{4.7}$$

Reprenons l'équation $\dot{x}(t) = f(x,t) + g(x,t).V(t)$, Le vecteur V est composé de deux grandeurs : V_{eq} et V_n , soit :

$$V(t) = V_{eq} + V_N \tag{4.8}$$

On obtient la dérivé de surface comme suit :

$$\dot{S}(x) = \frac{dS}{dt} = \frac{\partial S}{\partial x}\frac{\partial x}{\partial t} = \frac{\partial S}{\partial x}\{f(x,t) + g(x,t).V_{eq}(t)\} + \frac{\partial S}{\partial x}\{g(x,t).V_N\} \tag{4.9}$$

En mode de glissement et en régime permanent, la dérivée de la surface est nulle. Ainsi, nous obtenons :

$$V_{eq}(t) = -\{\frac{\partial S}{\partial x}g(x,t)\}^{-1}\{\frac{\partial S}{\partial x}f(x,t)\} \quad V_N = 0 \tag{4.10}$$

Durant le mode de convergence, nous obtenons une nouvelle expression de la dérivée de la surface :

$$\dot{S}(x) = \frac{\partial S}{\partial x}\{g(x,t)V_N\} \tag{4.11}$$

Et Le problème revient à trouver V_N tel que :

$$S(x)\dot{S}(x) = S(x)\frac{\partial S}{\partial x}\{g(x,t)V_N\} < 0 \tag{4.12}$$

La solution la plus simple est de choisir V_N sous la forme :

$$V(t) = V_{eq} + V_N \tag{4.13}$$

Dans ce cas, la commande s'écrit comme suit :

$$V_N = K.sign(S(x)) \tag{4.14}$$

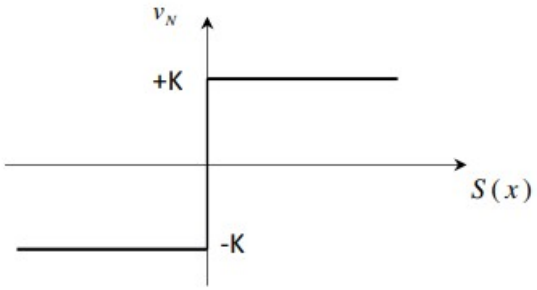

FIGURE 4.3 – Représentation de la fonction Sign

Alors on obtient :

$$S(x)\dot{S}(x) = \frac{\partial S}{\partial x}g(x,t)K|S(x)| < 0 \tag{4.15}$$

Tel que $\frac{\partial S}{\partial x}g(x,t)$ toujours négatif pour la classe des systèmes que nous considérons. Le gain $K > 0$, le choix du gain est très influent car, s'il est très petit le temps de réponse sera très long et s'il est très grand, nous aurons de fortes oscillations au niveau de l'organe de la commande. Ces oscillations peuvent exciter les dynamiques négligées (phénomène de

Chattering).

4.8 Phénomène de Chattering

Un mode glissant idéal n'existe pas étant donné qu'il nécessite une commande qui commute avec une fréquence infinie. Dans un cas réel la commutation se fait pendant un temps de commande, dans lequel sa discontinuité produit un comportement dynamique particulier autours de la surface de glissement, et ajoute au spectre de la commande des composantes de haute fréquence, qui peut endommager les actionneurs, ce qu'on appelle le phénomène de réticence ou bien Chattering . Ce phénomène peut être réduit en remplaçant la fonction «*sign*» par une fonction de saturation adéquate qui filtre les hautes fréquences [28].

4.8.1 La fonction sat

La fonction *sat* est donnée par :

$$sat(S) = \begin{cases} +1 & \text{si } S > \mu \\ -1 & \text{si } S < \mu \\ \frac{S}{\mu} & \text{si } |S| < \mu \end{cases} \tag{4.16}$$

Où μ est un paramètre petit et positif, la représentation de la fonction SAT est la suivante [28] :

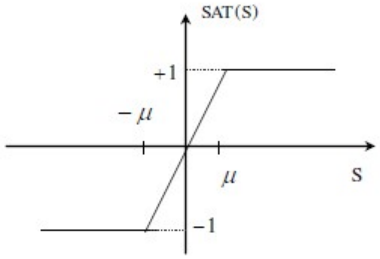

FIGURE 4.4 – Représentation de la fonction SAT

4.8.2 La fonction adoucie $SMOOTH$

La fonction $SMOOTH$ est donnée par [28] :

$$SMOOTH(S) = \frac{S}{|S| + \mu} \tag{4.17}$$

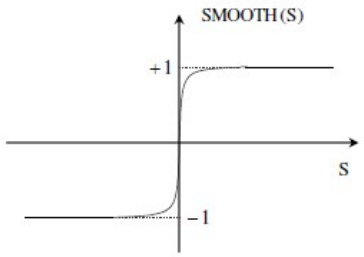

FIGURE 4.5 – Représentation de la fonction SMOOTH

4.9 Domaines d'application de la commande par mode glissant

Le réglage par mode glissant a connu des développements importants dans tous les domaines de la commande. Commençant par le domaine des commandes hydrauliques ou pneumatiques, il existe des valves fonctionnant par tout ou rien qui ne possèdent que deux états stables, ou complètement ouvertes ou fermées. Aussi les entraînements électriques pour les machines électriques, ou robots qui nécessitent, soit un réglage de la vitesse de rotation, soit un réglage de position. Enfin dans certains domaines tels que, les processus chimique et métallurgiques, où il y'a des réglages qui font appel à des régulateurs à deux positions[28].

4.10 Modélisation cinématique du sous-marin

Généralement, le mouvement d'un sous-marin autonome à six degrés de libertés (6ddl) peut être représenté en équation différentielles. Ces équations sont développées en utilisant deux cadres de coordonnées comme l'illustre la figure ci-dessous :
Six composantes de vitesse $[u, v, w, p, q, r]$ (surtension, balancement, vitesse de soulèvement, roulis, taux de tangage, taux de lacet) sont définies dans le cadre fixe à la carrosserie, tandis que le cadre définit les attitudes correspondantes et positions $[x, y, z, \Phi, \Theta$ et $\Psi]$.
Ainsi, le mouvement de l'engin est décrit par les paramètres suivants :
− On note $\eta = [x, y, z, \Phi, \Theta, \Psi]^T$ le vecteur d'état représentant la position et l'orientation du véhicule dans le repère terrestre R_0.
− Les coordonnées x, y et z, exprimées en mètres, représentent la position du centre du repère R_v dans le repère R_0. On note $\eta_1 = [x_0, y_0, z_0]^T$ le vecteur de position du centre de gravité par rapport au repère fixe R_0 exprimé dans le repère fixe R_0.
− Les angles Φ, Θ et Ψ, exprimés en radians, sont les angles qui indiquent l'orientation du repère R_v par rapport au repère R_0, communément appelés angles d'Euler.
Ces angles (Φ, Θ, Ψ) sont nommés respectivement roulis (ou gîte), tangage (ou assiette) et lacet (ou cap). On note $\eta_2 = [\Phi, \Theta, \Psi]^T$ le vecteur altitude par rapport au repère fixe R_0 exprimé dans le repère fixe R_0.

– On note $v^T = [u, v, w, p, q, r]^T$ le vecteur qui rassemble les vitesses linéaires et angulaires dans le repère R_v lié au véhicule.
– Les vitesses u, v et w, exprimées en m/s, correspondent aux vitesses selon les axes O_x, O_y et O_z du véhicule et sont appelées respectivement vitesse longitudinale (surge), vitesse ou dérive transversale (sway) et vitesse de pilonnement (heave). On note $v_1 = [u, v, w]^T$ le vecteur vitesse local de translation par rapport à R_0 exprimé dans le repère local R_v.
– Les vitesses de rotation p, q et r, exprimées en rad/s, correspondent aux vitesses de rotation autour des axes O_x, O_y et O_z, c'est-à-dire les vitesses de roulis, tangage et lacet du véhicule respectivement. On note $v_2 = [p, q, r]^T$ le vecteur vitesse local de rotation par rapport à R_0 exprimé dans le repère local R_v [27].

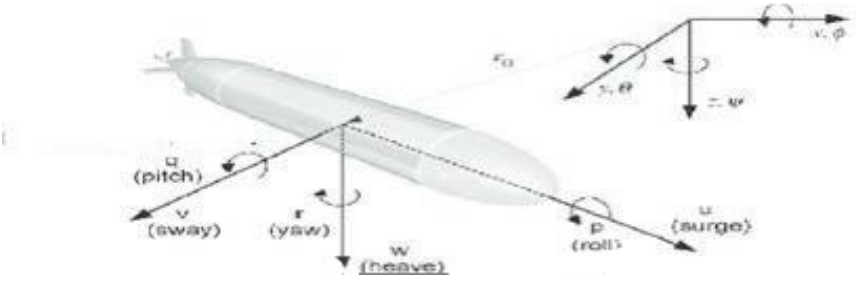

FIGURE 4.6 – Repère 3D

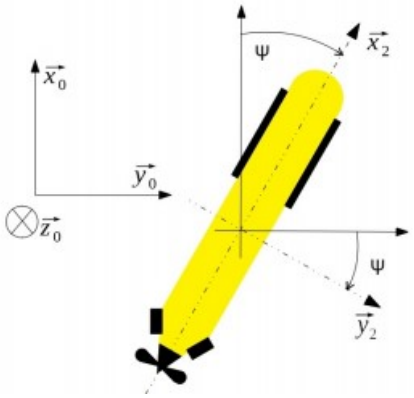

FIGURE 4.7 – Définition du lacet - Vue de dessus

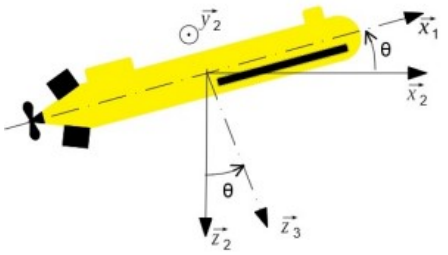

FIGURE 4.8 – Définition du tangage - Vue de côté

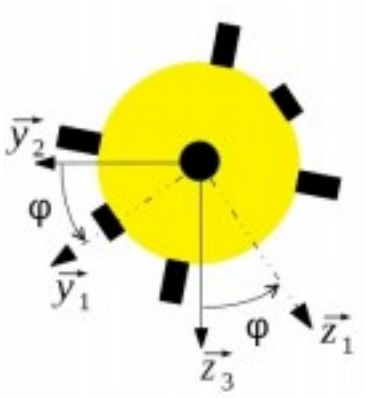

FIGURE 4.9 – Définition du roulis - Vue de face

DDL	Mouvement	Forces Moments	Rapidité	Positions angles d'Euler
1	vitesse longitudinale	X	μ	x
2	vitesse transversale	Y	υ	y
3	vitesse de pilonnement	Z	w	z
4	roulis	K	P	Φ
5	tangage	M	q	Θ
6	lacet	N	r	Ψ

TABLE 4.1 – Symboles utilisés pour décrire le 6 DDL

4.10.1 Définition des matrices de passage

La configuration de l'engin est décrite au moyen de trois rotations élémentaires définies par trois angles d'orientation à savoir le lacet Ψ, le tangage Θ et le roulis Φ :

La matrice de rotation $J_1(\eta_2)$ qui décrit la transformation entre le repère global R_0 et le repère local R_v est donnée par :

$$\begin{bmatrix} cos(\Theta)cos(\Psi) & sin(\Phi)sin(\Theta)cos(\Psi) - cos(\Phi)sin(\Psi) & cos(\Phi)sin(\Theta)cos(\Psi) + sin(\Phi)sin(\Psi) \\ cos(\Theta)sin(\Psi) & sin(\Phi)sin(\Theta)sin(\Psi) + cos(\Phi)cos(\Psi) & cos(\Phi)sin(\Theta)sin(\Psi) - sin(\Phi)cos(\Psi) \\ -sin(\Theta) & sin(\Phi)cos(\Theta) & cos(\Phi)cos(\Theta) \end{bmatrix}$$

4.10.2 Transformation des vitesses de translation et de rotation

A l'aide de la matrice de passage de changement de base $J_1(\eta_2)$, la transformation des composantes des vitesses de translation, par rapport à R_0 est donnée par :

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = J_1(\eta_2) \begin{bmatrix} u \\ v \\ w \end{bmatrix} \tag{4.18}$$

Inversement, les vitesses du cadre des coordonnées du corps peuvent être déterminées à partir des vitesses du cadre de coordonnées mondiales dans un mode similaire :

$$v_1 = J_1^{-1}(\eta_2)\dot{\eta}_1 \tag{4.19}$$

La vitesse locale de rotation peut être exprimée ainsi :

$$v_2 = \dot{\Psi}Z_0 + \dot{\Theta}Y_1 + \dot{\Phi}X_v \tag{4.20}$$

On peut alors exprimer le vecteur vitesse de rotation dans la base du repère mobile comme suit :

$$v_2 = \begin{bmatrix} p \\ q \\ r \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(\Phi) & sin(\Phi) \\ 0 & -sin(\Phi) & cos(\Phi) \end{bmatrix} \begin{bmatrix} \dot{\Phi} - \dot{\Psi}sin(\Theta) \\ \dot{\Theta} \\ \dot{\Phi}cos(\Theta) \end{bmatrix}$$

Alors :

$$v_2 = \begin{bmatrix} p \\ q \\ r \end{bmatrix} = \begin{bmatrix} \dot{\Phi} - \dot{\Psi}sin(\Theta) \\ \dot{\Theta}cos(\Phi) + \dot{\Psi}sin(\Phi)cos(\Theta) \\ \dot{\Psi}cos(\Theta)cos(\Phi) - \dot{\Theta}sin(\Phi) \end{bmatrix}$$

Où encore :

$$v_2 = \begin{bmatrix} p \\ q \\ r \end{bmatrix} = \begin{bmatrix} 1 & 0 & -sin(\Theta) \\ 0 & cos(\Phi) & sin(\Phi)cos(\Theta) \\ 0 & -sin(\Phi) & cos(\Phi)cos(\Theta) \end{bmatrix} \begin{bmatrix} \dot{\Phi} \\ \dot{\Theta} \\ \dot{\Psi} \end{bmatrix} \tag{4.21}$$

La dernière équation nous permet d'écrire :

$$\begin{aligned} \dot{\Phi} &= p + qsin(\Phi)tan(\Theta) + rcos(\Phi)tan(\Theta) \\ \dot{\Theta} &= qcos(\Phi) - rsin(\Phi) \\ \dot{\Psi} &= \frac{qsin(\Phi) + rcos(\Phi)}{cos(\Theta)} \end{aligned} \tag{4.22}$$

Remarque : Il est à noter que la paramétrisation en tangage fait apparaître une singularité en $\Theta = \Pi 2 + k\pi$. Cette paramétrisation est inaccessible car grâce aux phénomènes hydrostatiques hydrodynamique et à la propulsion, il est impossible pour le R_{ov} d'avoir un angle de tangage 90°.

A l'aide des équations (4.18) et (4.22), la cinématique du robot sous-marin peut s'exprimer de la manière suivante :

$$\dot{\Phi} = p + qsin(\Phi)tan(\Theta) + rcos(\Phi)tan(\Theta) \tag{4.23}$$

$$\dot{\Theta} = qcos(\Phi) + rsin(\Phi) \tag{4.24}$$

$$\dot{\Psi} = \frac{(qsin(\Phi) + r.cos(\Phi))}{cos(\Theta)} \tag{4.25}$$

Aussi :

$$\begin{aligned} \dot{x} = u(cos(\Theta)cos(\Psi)) + v(sin(\Phi)sin(\Theta)cos(\Psi) - cos(\Phi)sin(\Psi)) \\ + w(cos(\Phi)sin(\Theta)cos(\Psi) - sin(\Phi)sin(\Psi)) \end{aligned} \tag{4.26}$$

$$\begin{aligned} \dot{y} = u(cos(\Theta)sin(\Psi)) + v(sin(\Phi)sin(\Theta)sin(\Psi) + cos(\Phi)cos(\Psi)) \\ + w(cos(\Phi)sin(\Theta)sin(\Psi) - sin(\Phi)cos(\Psi)) \end{aligned} \tag{4.27}$$

$$\dot{h} = -\dot{z} = u(sin(\Theta)) + v(-sin(\Phi).cos(\Theta)) + w(-cos(\Phi)cos(\Theta)) \tag{4.28}$$

4.10.3 Modélisation de l'angle de dérape en Lacet

L'angle de dérape en lacet (Θ) peut être modélisé via la fonction de transfert de boucle interne relative à l'angle de poupe d'entrée (f_s) rapport à l'angle de pas de sortie du véhicule

(Θ). Selon les équations cinématiques et dynamiques exprimé, la fonction de transfert de la boucle de profondeur interne est comme suit :

$$G_\Theta(s) = \frac{\Theta(s)}{f_s(s)} = \frac{\frac{M_{f_s}}{I_y - M_q}}{s^2 - \frac{M_q}{I_y - M_q}s - \frac{M_\Theta}{I_y - M_q}} \tag{4.29}$$

D'où l'écriture dans le domaine temporelle :

$$\ddot{\Theta} = a_1 f_s + a_2 \dot{\Theta} + a_3 \Theta \tag{4.30}$$

Avec :

$$\begin{aligned} a_1 &= \frac{M_{f_1}}{I_y - M_q} \\ a_2 &= \frac{M_q}{I_y - M_q} \\ a_3 &= \frac{M_\Theta}{I_y - M_q} \end{aligned} \tag{4.31}$$

Dans des conditions réelles, la réponse de la boucle interne est beaucoup plus rapide que la boucle de profondeur externe, donc on peut être approximé à Θ. Le calcul de la fonction de transfert de la boucle extérieure reliant l'angle désiré Θ_d d'entrée et la profondeur du véhicule (z) donne la relation suivante :

$$G_z(s) = \frac{z(s)}{\Theta(s)} = -\frac{\mu_1}{s} \tag{4.32}$$

Ce qui donne l'équation différentielle suivante :

$$\dot{z} = \mu_1 \Theta \tag{4.33}$$

4.11 Synthèse de la commande par les modes glissants

4.11.1 Commande de la profondeur z par fs

Soit : $e = \Theta - \Theta_d$ l'erreur entre le Θ obtenu et le Θ désiré Θ_d, On définit la surface de glissement suivante :

$$S = \frac{d}{dt}e + \lambda_1 e \tag{4.34}$$

Sa dérivée par rapport au temps est donnée par :

$$\dot{S} = \frac{d^2}{dt^2}e + \lambda_1 \frac{d}{dt}e \tag{4.35}$$

En rappelant (4.30) , l'équation (4.35) permet d'avoir :

$$\dot{S} = a_1 f_s + a_3 \Theta + (\lambda_1 + a_2)\dot{\Theta} \tag{4.36}$$

Soit maintenant la fonction de Lyapunov candidate suivante :

$$V = \frac{1}{2} S^2 \tag{4.37}$$

Donc :

$$\dot{V} = \frac{dV}{dS}\frac{dS}{dt} = S\dot{S} \tag{4.38}$$

Si on impose :

$$\dot{S} = -K sign(S) - QS \text{ avec : } K, Q > 0 \tag{4.39}$$

On peut déduire à partir de (4.38) que $\dot{V} = -K|S| - QS^2 \leq -K|S| \leq 0$.
Selon la théorie de Lyapunov, le système converge asymptotiquement vers sa valeur désirée.
Ainsi, l'angle de renforcement d'entrée sera donnée par :

$$-K sign(S) - QS = a_1 f_s + a_2 \frac{d}{dt}\Theta + a_3 \Theta + \lambda_1 \frac{d}{dt}\Theta \tag{4.40}$$

$$f_s = \frac{-K sign(S) - QS}{a_1} - \frac{a_2}{a_1}\frac{d}{dt}\Theta - \frac{a_3}{a_1}\Theta - \frac{\lambda_1}{a_1}\frac{d}{dt}\Theta \tag{4.41}$$

Donc la commaned f_s qui permet la stabilisation de Θ sur sa valeur désirée est donnée par l'expression suivante :

$$f_s = \frac{-K sign(S) - QS}{a_1} - \left(\frac{a_2 + \lambda_1}{a_3}\right)\frac{d}{dt}\Theta - \frac{a_3}{a_1}\Theta \tag{4.42}$$

4.11.2 Commande de la profondeur z par U

En supposant que le robot effectue une plongeé dans le plan $(z - y)$ ce qui signifie que $v = w = 0$ et à partir de (4.30) , la profondeur z sera donnée par :

$$\dot{h} = -\dot{z} = u \sin(\Theta) \tag{4.43}$$

Pour permettre une poursuite d'une trajectoire variable dans le temps, on considére la trajectoire de référence $z_d(t)$. la fonction $z_d(t)$ doit vérifier les conditions de continuité et de bornitudes de $\dot{z}_d(t)$ et $\ddot{z}_d(t)$.
Alors, l'erreur de poursuite e_p donnée par $e_p = z - z_d(t)$ aura la dynamique suivante :

$$\dot{e}_p = \dot{z} - \dot{z}_d(t) = -u \sin(\Theta) - \dot{z}_d(t) \tag{4.44}$$

par la suite, da deuxième dérivée de $\dot{e}_p$ par rapport au temps sera comme suit :

$$\ddot{e}_p = \ddot{z} - \ddot{z}_d(t) = -\dot{u}\sin(\Theta) - u\dot{\Theta}cos(\Theta) - \ddot{z}_d(t) \tag{4.45}$$

Pour un gain $\lambda_2 > 0$, on définit la surface de glissement suivante :

$$S_2 = \dot{e}_p + \lambda_2 e_p \tag{4.46}$$

Sa dérivé par rapport au temps donne :

$$\dot{S}_2 = \ddot{e}_p + \lambda_2 \dot{e}_p \tag{4.47}$$

En utilisant les équations (4.44) et (4.45), nous obtenons :

$$\dot{S}_2 = \sin(\Theta)(-\dot{u} - \lambda_2 u) - u\dot{\Theta}\cos(\Theta) - \ddot{z}_d(t) - \lambda_2 \dot{z}_d(t) \tag{4.48}$$

Soit la fonction de Lyaponov suivante :

$$V_1 = \frac{1}{2}S_2^2 \tag{4.49}$$

D'où :

$$\dot{V}_1 = S_2 \dot{S}_2 \tag{4.50}$$

Si on choisit :

$$\dot{S}_2 = -K_2 sign(S_2) - Q_2 S_2 \text{ avec : } K_2, Q2 > 0 \tag{4.51}$$

Donc, la dynamique de V_1 devient telle que :

$$\dot{V}_1 = -K_2|S_2| - Q_2 S_2^2 \leq -K_2|S_2| \leq 0 \tag{4.52}$$

Selon la théorie de Lyapunov, le système est stable et converge asymptotiquement vers zéro. Ainsi, le signal de commande sera donné par :

$$\sin(\Theta)(-\dot{u} - \lambda_2 u) - u\dot{\Theta}cos(\Theta) - \ddot{z}_d(t) - \lambda_2 \dot{z}_d(t) = -K_2 sign(S_2) - Q_2 S_2 \tag{4.53}$$

Finalement, et puisque $\sin(\Theta) \neq 0$, le signal de la deuxiéme commande aura la forme différentielle suivante :

$$\dot{u} = \frac{K_2 sign(S_2) + Q_2 S_2}{\sin(\Theta)} + u\left(-\lambda_2 + \frac{\sin(\theta)}{\cos(\theta)}\dot{\Theta}\right) + \left(\frac{\ddot{z}_d(t) + \lambda_2 \dot{z}_d(t)}{\sin(\theta)}\right) \tag{4.54}$$

4.11.3 Commande de la profondeur z avec l'angle de plongé Θ et une surtension u constante

Pour ce cas, on va considérer une nouvelle fonction de Lyapunov définie par :

$$V_2 = \frac{1}{2}e_p^2 \tag{4.55}$$

Sa dérivée par rapport au temps donne :

$$\dot{V}_2 = e_p \dot{e}_p \tag{4.56}$$

En utilisant l'équation (4.44) on obtient :

$$\dot{V}_2 = e_p \left(-u \sin(\Theta) - \dot{z}_d(t)\right) \tag{4.57}$$

On pose :

$$\dot{e}_p = -K_3 sign(e_p) - Q_3 e_p \text{ avec } : K_3, Q_3 > 0 \tag{4.58}$$

donc la forme final de la dynamique de V_1 sera donnée par :

$$\dot{V}_2 = -K_3|e_p| - Q_3 e_p^2 \leq -K_3|e_p| \leq 0 \tag{4.59}$$

Selon la théorie de Lyapunov, le système est stable et converge asymptotiquement vers sa valeur désirée. Donc d'après l'équation (4.57) et (4.58) on obtient le signal de commande suivant :

$$\Theta = \arcsin\left(\frac{K}{u} sign(z) - \dot{z}_d(t)\right) \tag{4.60}$$

Remarque - Pour eliminer l'effet du chattering , nous avons remplacer la fonction standard "sign" par la fonciton tangentehyperbolique définie par :

$$\tanh x = a\left(\frac{e^x - e^{-x}}{e^x + e^{-x}}\right); \text{avec } : a = 1 \tag{4.61}$$

4.12 Simulation sur Matlab et résultats obtenus

4.12.1 Commande 1 et 2

Nous allons effectuer dans cette partie la commande par mode glissant, tel que d'une part nous allons saisir un angle de plongé désiré θ_d avec un angle de renforcement d'entrée f_s convenable dans le but d'atteindre la profondeur désiré z_d en visualisant les changements

d'angle de plongé en fonction du temps. D'autre part nous allons atteindre z_d avec une surtension U convenable.
Notre objectif est d'atteindre profondeur Z = 2m avec l'angle de plongé Tetad = pi/4 en appliquant l'équation sign.

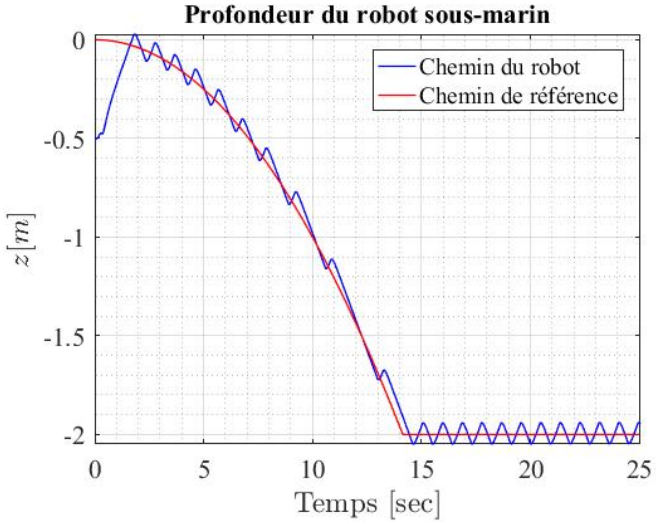

FIGURE 4.10 – Chemin du drone sous-marin par rapport au temps

La figure (4.10) représente le chemin de notre drone sous-marin par rapport au temps pour atteindre la profondeur $2m$, commençant par une profondeur initiale à Temps = $0s$, $z = -0.5m$ suivant le chemin de la trajectoire parabolique (graphe rouge) comme référence attrapé à temps = $2s$ à $z = 0.032m$, puis notre robot converge vers cette référence puis notre robot converge avec des oscillations de petites fréquences afin d'atteindre la profondeur désiré $z = -2m$, à Temps = $14.7s$.

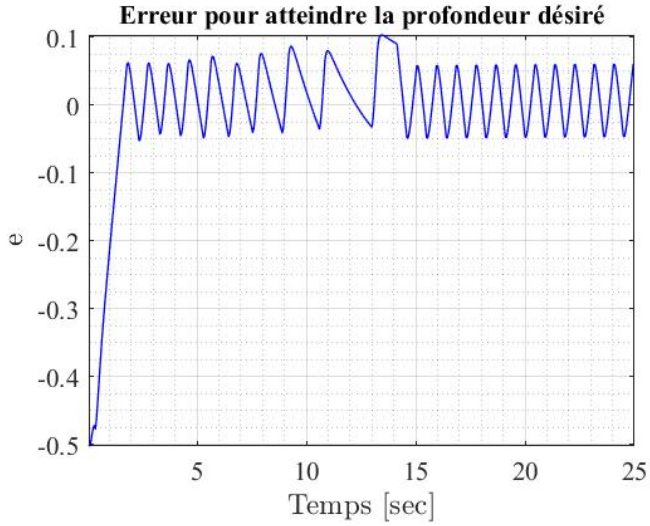

FIGURE 4.11 – Erreur pour atteindre la profondeur désirée

La figure (4.11) représente l'erreur entre chemin du robot et la trajectoire de référence

par rapport au temps, tel que l'erreur initial égal à -0.5 à temps = $0s$ proportionnelle avec le mouvement du sous-marin pour atteindre les $2m$ (figure 4.10), l'erreur maximale à temps= $13.45s$ est à 0.1 dans laquelle le drone flotte sur l'eau, puis termine l'erreur jusqu'à l'infini avec des variations entre ˇ0.05 et +0.05.

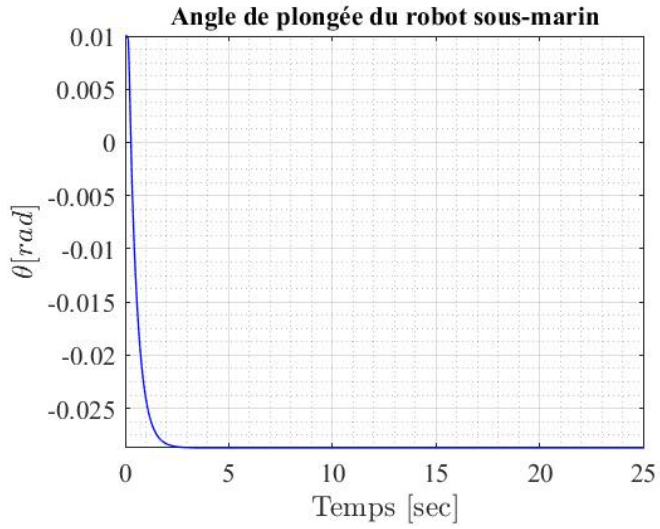

FIGURE 4.12 – Angle de plongé

La figure (4.12) représente l'angle de plongé de notre drone sous-marin par rapport au temps, commençant par un angle initial à Temps = $0s$, $\Theta = 0.01rad$ et l'angle égale à $-0.028rad$ à Temps = $2s$ lorsque le robot commence à suivre la trajectoire du chemin de référence.

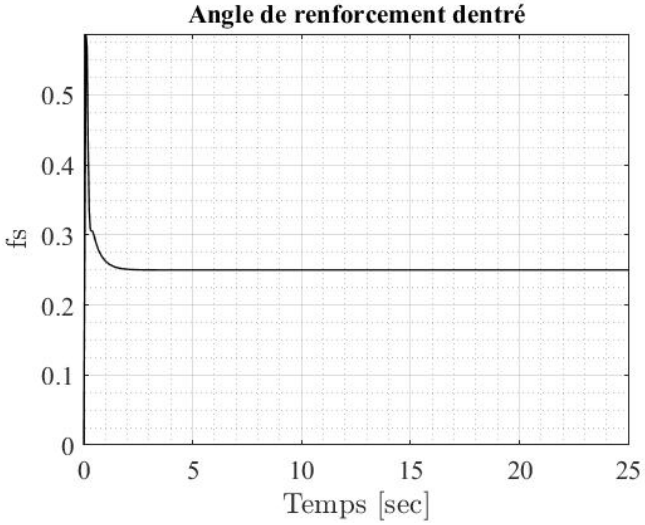

FIGURE 4.13 – Angle de renforcement d'entrée

La figure (4.13) représente l'angle de renforcement d'entrée commandé de notre drone sous-marin par rapport au temps, commençant par une valeur de f_s initiale $1rad$ à Temps = $0s$, et atteindre l'angle maximal à $f_s = 0.25rad$, Temps = $2.05s$, et se stabilise lorsque le

robot commence à suivre la trajectoire du chemin de référence jusqu'à la fin du trajectoire du robot.

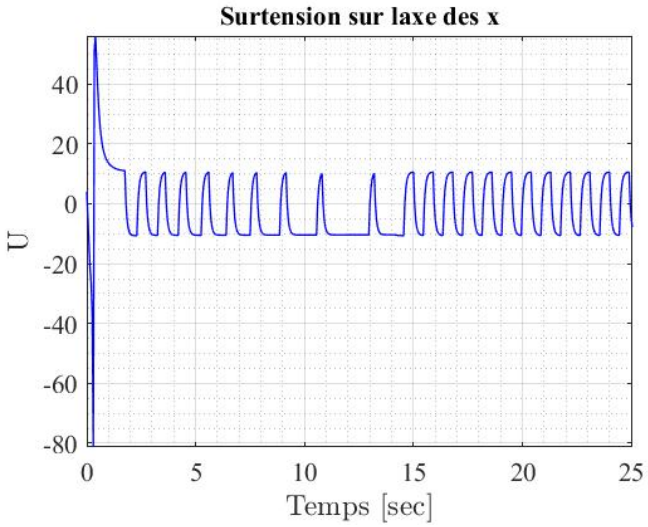

FIGURE 4.14 – Représentation de la commande u

La figure (4.14) représente la surtension sur l'axe des x de notre drone sous-marin par rapport au temps, commençant par une valeur de U initiale est nul à Temps = $0s$, puis après $1.75s$ le $U = -11.02rad/s$ puis termine avec des oscillations entre $U = 10.52rad/s$ et $U = -10.52rad/s$.

Interprétation des résultats obtenus : Nous arrivons à dire que la fonction *sign* dans les deux commandes a un impact sur la surtension, et la trajectoire le temps que notre robot arrive à la profondeur que nous souhaite avec des oscillations de petites fréquences ce qu'on appelle phénomène de Chattering.

Notre objectif est d'atteindre profondeur $z = 2m$ avec l'angle de plongé $\Theta_d = \frac{\pi}{4}$ en ajoutant une fonction hyperbolique pour éliminer le broutement.

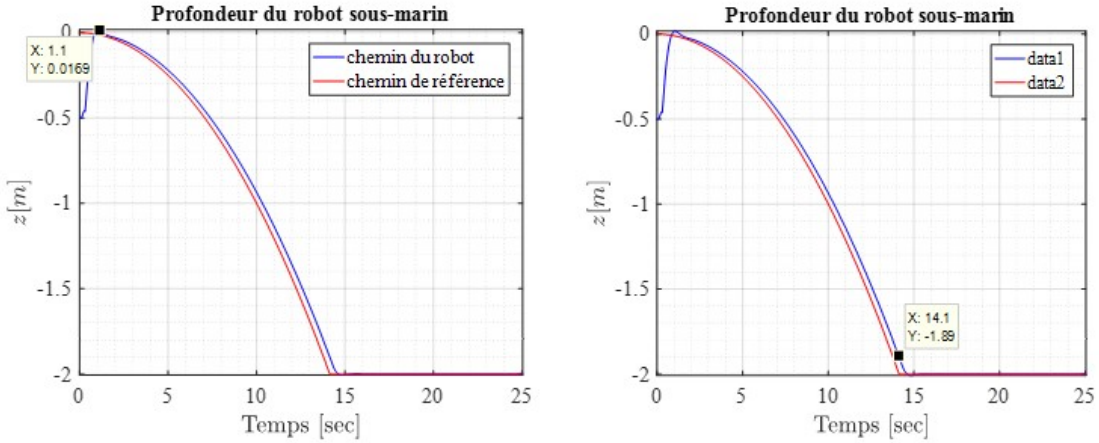

FIGURE 4.15 – Chemin de notre drone sous-marin par rapport au temps

La figure (4.15) représente le chemin de notre drone sous-marin par rapport au temps pour atteindre la profondeur $2m$, commençant par une profondeur initiale à Temps = $0s$, $z = -0.5m$ suivant le chemin de la trajectoire parabolique (graphe rouge) comme référence attrapé à temps = $2s$ à $z = 0.032m$, puis notre robot converge vers cette référence afin d'atteindre la profondeur désiré $z = -2m$, à Temps = $14.7s$.

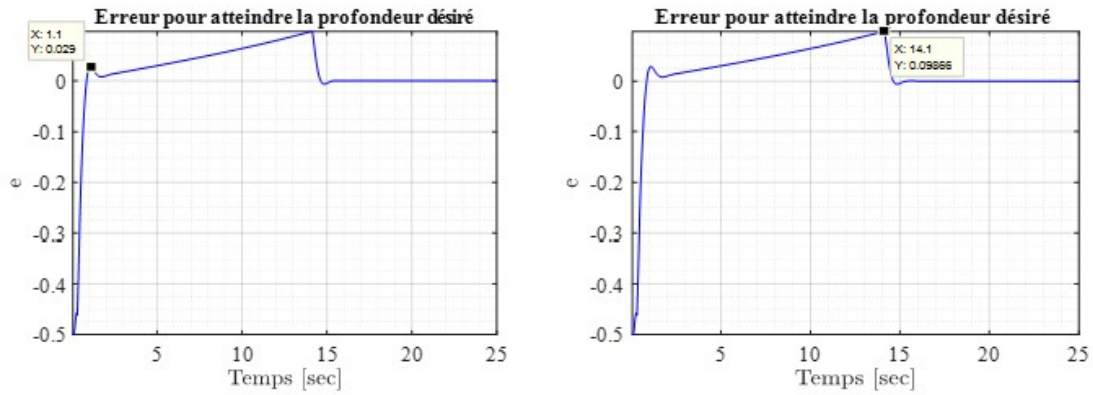

FIGURE 4.16 – Erreur pour atteindre la profondeur désirée

La figure (4.16) représente l'erreur entre chemin du robot et la trajectoire de référence par rapport au temps, tel que l'erreur initial égal à -0.5 à temps = $0s$ proportionnelle avec le mouvement du sous-marin pour atteindre les $2m$ (figure 4.15), l'erreur à temps = $1.1s$ est à 0.029 dans laquelle le drone flotte sur l'eau, puis cette dernière prend la valeur de 0.098 lorsque le robot atteint la profondeur $z = -1.89m$ à temps = $14.1s$ et à temps = $14.7s$ lorsque le drone touche $-2m$ alors l'erreur s'élimine complétement.

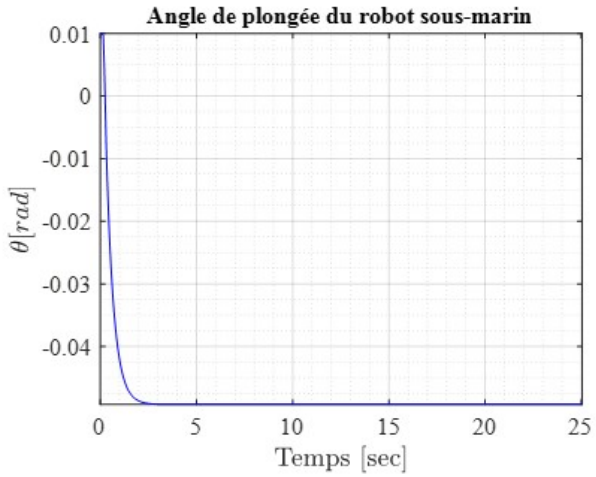

FIGURE 4.17 – Angle de plongé

La figure (4.17) représente l'angle de plongé de notre drone sous-marin par rapport au temps, commençant par un angle initial à Temps = $0s$, $\Theta = 0.01rad$ et l'angle égale à $0.048rad$ à Temps = $2s$ lorsque le robot commence à suivre la trajectoire du chemin de référence.

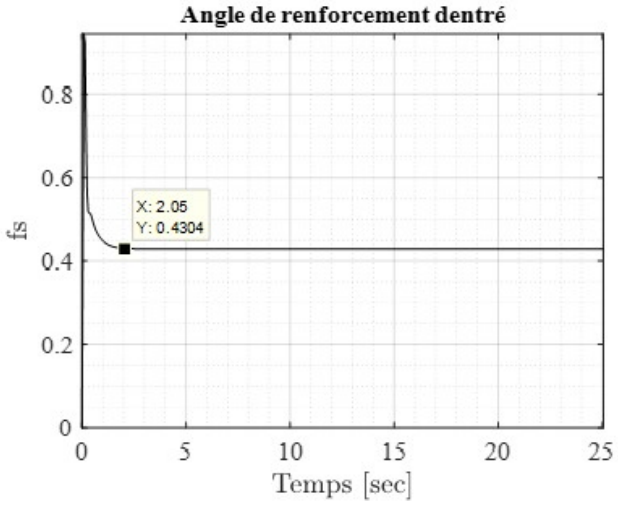

FIGURE 4.18 – Angle de renforcement d'entrée

La figure (4.18) représente l'angle de renforcement d'entrée commandé de notre drone sous-marin par rapport au temps, commençant par une valeur de f_s initiale $1rad$ à Temps = $0s$, et atteindre l'angle maximal à $f_s = 0.43rad$, Temps = $2.05s$, et se stabilise lorsque le robot commence à suivre la trajectoire du chemin de référence.

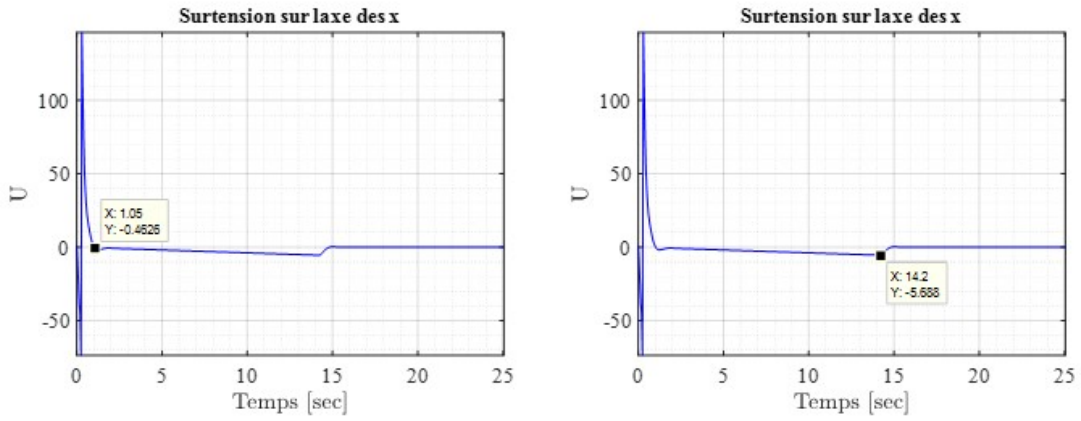

FIGURE 4.19 – Représentation de la commande u

La figure (4.19) représente la surtension sur l'axe des x de notre drone sous-marin par rapport au temps, commençant par une valeur de U initiale est nul à Temps = $0s$, puis après $1.05s$ le $U = -0.48$ lorsque le drone flotte, et atteint U finale = -5.65, Temps = $14.2s$ avec $z = -1.89m$, puis s'annule lorsque le robot est sur la profondeur désiré $zZ = -2m$, à Temps = $14.7s$.

Notre allons garder la même profondeur désiré $z = 2m$ avec l'angle de plongé $\Theta_d = \frac{\pi}{6}$.

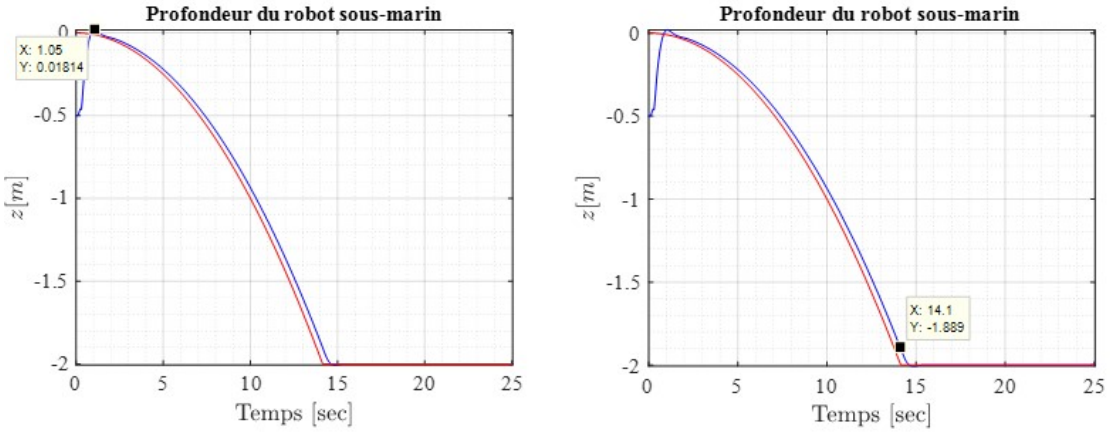

FIGURE 4.20 – Chemin de notre drone sous-marin par rapport au temps

La figure (4.20) représente le chemin de notre drone sous-marin par rapport au temps pour atteindre la profondeur $2m$, commençant par une profondeur initiale à Temps = $0s$, $z = -0.5m$ suivant le chemin de la trajectoire parabolique (graphe rouge) comme référence attrapé à temps = $2s$ à $z = 0.032m$, puis notre robot converge vers cette référence afin d'atteindre la profondeur désiré $z = -2m$, à Temps = $14.7s$.

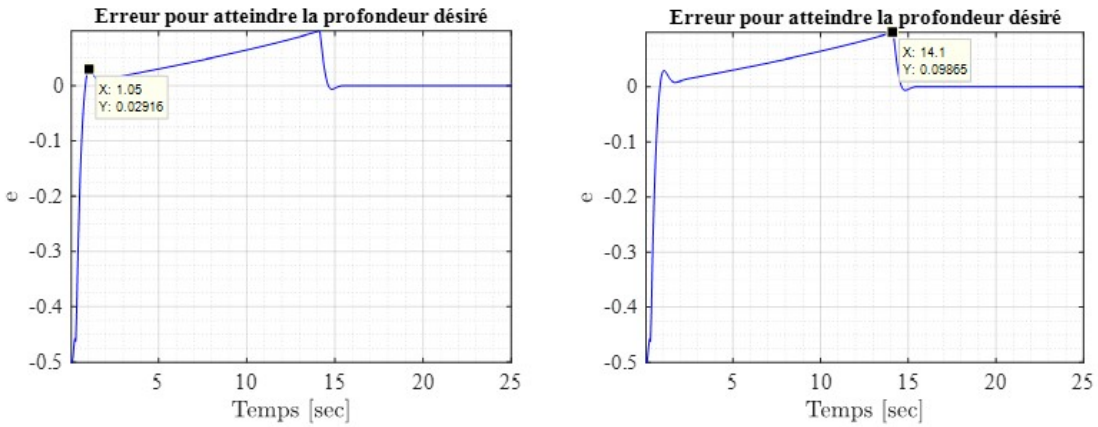

FIGURE 4.21 – erreur pour atteindre la profondeur désirée

La figure (4.21) représente l'erreur entre chemin du robot, et la trajectoire de référence par rapport au temps, tel que l'erreur initial égal à -0.5 à temps $= 0s$ proportionnelle avec le mouvement du sous-marin pour atteindre les $2m$ (figure (4.20)), l'erreur à temps $= 1.05s$ est à 0.029 dans laquelle le drone flotte sur l'eau, puis cette dernière prend la valeur de 0.098 lorsque le robot atteint la profondeur $z = -1.889m$ à temps $= 14.1s$ et à temps $= 14.7s$ lorsque le drone touche $-2m$ alors l'erreur s'élimine complétement.

La figure de mouvement d'angle de plongé est la même que la figure ((4.17)), ce qui nous permet de dire que le changement d'angle désiré n'a aucun impact sur la variation d'angle de plongé du robot.

La figure d'angle de renforcement d'entrée commandé de notre drone sous-marin par rapport au temps, ((4.18)), ce qui nous permet de dire que le changement d'angle désiré n'a aucun impact sur la variation d'angle de renforcement de plongé du robot.

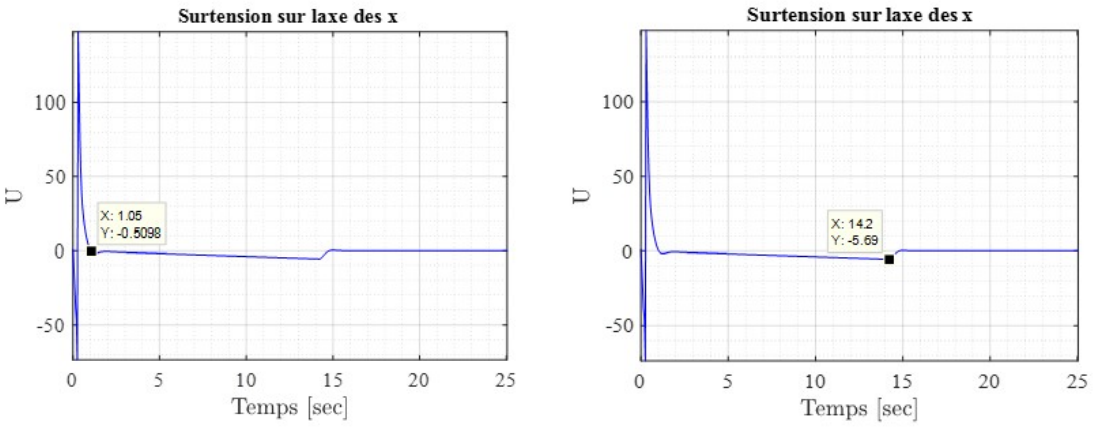

FIGURE 4.22 – surtension sur l'axe x

La figure (4.22) représente la surtension sur l'axe des x de notre drone sous-marin par rapport au temps, commençant par une valeur de U initiale est nul à Temps = $0s$, puis après $1.05s$ le $U = -0.50$ lorsque le drone flotte, et atteint U finale = -5.69, Temps = $14.2s$ avec $z = -1.89m$, puis s'annule lorsque le robot est sur la profondeur désiré $z = -2m$, à Temps = $14.7s$. Ce que nous avons remarqué que le Θ désiré a un impact sur la surtension par rapport aux résultats précédents, et le déplacement du drone sous l'eau est proportionnel avec la vitesse sur l'axe des x.

Interprétation des résultats obtenus : L'angle de renforcement d'entré F_s est proportionnel avec la variation de la surtension U pour permettre le robot de nager jusqu'à la profondeur que nous souhaitons.

Application avec perturbation

Nous allons appliquer un bruit sinusoïdale sur notre système tel que la profondeur qu'on désire $Z = 2m$ et l'angle désiré $\Theta_d = \frac{\pi}{4}$.

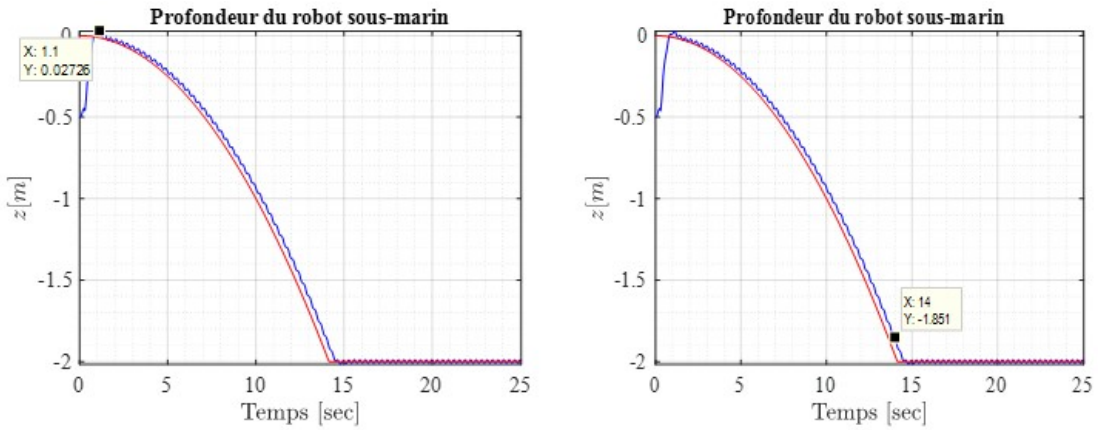

FIGURE 4.23 – Chemin de notre drone sous-marin par rapport au temps sous l'effet du bruit

La figure (4.23) représente le chemin de notre drone sous-marin par rapport au temps pour atteindre la profondeur $2m$ en appliquant un bruit sinusoïdale, commençant par une profondeur initiale à Temps = $0s$, $z = -0.5m$ suivant le chemin de la trajectoire parabolique (graphe rouge) comme référence attrapé à temps = $2s$ à $z = 0.032m$, puis notre robot converge vers cette référence afin d'atteindre la profondeur désiré $z = -2m$, à Temps = $14.7s$, on remarque que le bruit appliqué a causé des oscillations pendant le déplacement.

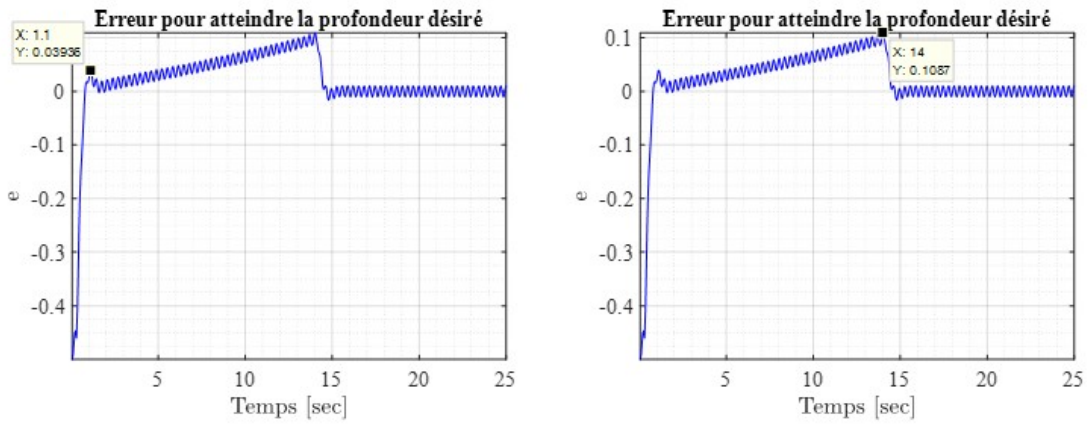

FIGURE 4.24 – erreur pour atteindre la profondeur désirée

La figure (4.24) représente l'erreur entre chemin du robot et la trajectoire de référence par rapport au temps, tel que l'erreur initial égal à -0.5 à temps $= 0s$ proportionnelle avec le mouvement du sous-marin pour atteindre les $2m$ (figure (4.23)) avec des oscillations de petites fréquences, l'erreur à temps $= 1.1s$ est à 0.029 dans laquelle le drone flotte sur l'eau, puis cette dernière prend la valeur de 0.108 lorsque le robot atteint la profondeur $z = -1.85m$ à temps $= 14s$ et à temps $= 14.7s$ lorsque le drone touche $-2m$ alors l'erreur s'élimine complétement en gardant les oscillations à l'infini.

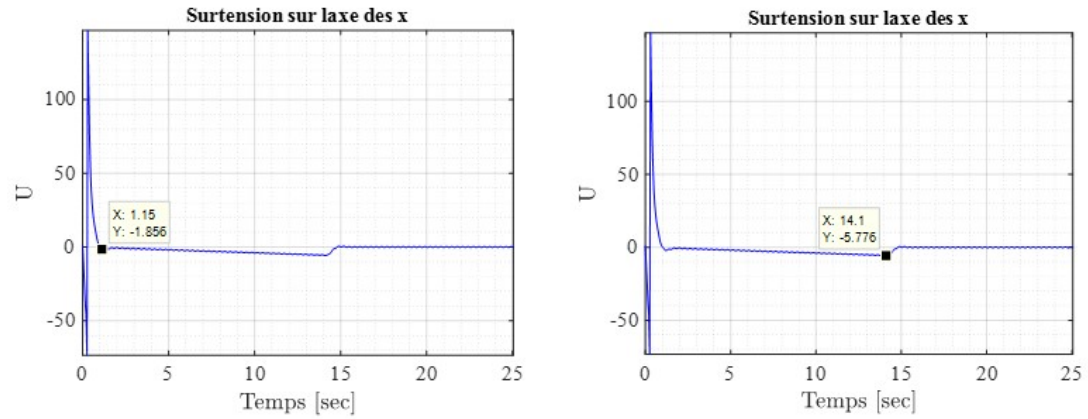

FIGURE 4.25 – surtension sur l'axe x

La figure (4.25) représente la surtension sur l'axe des x de notre drone sous-marin par rapport au temps, commençant par une valeur de U initiale est nul à Temps $= 0s$, puis après $1.15s$ le $U = -1.856$ lorsque le drone flotte, et atteindra U finale $= -5.776$, Temps $= 14.1s$ avec $z = -1.8m$, puis s'annule lorsque le robot est sur la profondeur désiré $z = -2m$, à Temps $= 15s$ avec des oscillations des petites fréquences.

Interprétation des résultats obtenus : Nous constatons que le bruit dans les deux commandes n'affecte pas le f_s non plus la variation de θ, donc il a un impact seulement sur la surtension, et la trajectoire le temps que notre robot arrive à la profondeur que nous souhaite avec des oscillations de petites fréquences ce qu'on appelle phénomène de Chattering.

4.12.2 Commande 3

Nous allons effectuer dans cette dernière partie la commande par mode glissant, tel que la surtension U est fixe $U = 0.9m/s$ dans le but de commander la profondeur que nous souhaitons par l'angle de plongé que nous appliquons.

Nous allons garder même profondeur désiré $z = 2m$ avec l'angle de plongé $\Theta_d = \frac{\pi}{4}$ avec $U = 0.9m/s$.

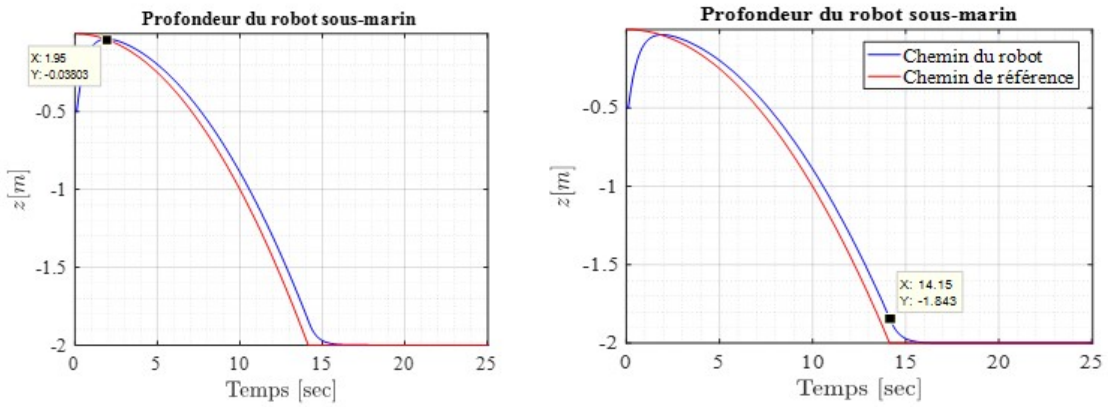

FIGURE 4.26 – Profondeur du robot sous marin

La figure (4.26) représente le chemin de notre drone sous-marin par rapport au temps pour atteindre la profondeur $2m$, commençant par une profondeur initiale à Temps $= 0s$, $z = -0.5m$ suivant le chemin de la trajectoire parabolique (graphe rouge) comme référence attrapé à temps $= 2s$, puis notre robot converge vers cette référence afin d'atteindre la profondeur désiré $z = -2m$, à l'instant $= 15.7s$.

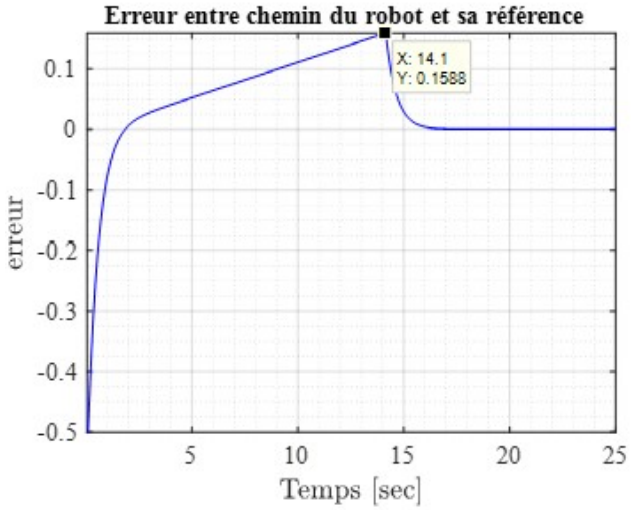

FIGURE 4.27 – erreur pour atteindre la profondeur désirée

La figure (4.27) représente l'erreur entre chemin du robot et la trajectoire de référence par rapport au temps, tel que l'erreur initial égal à -0.5 à temps $= 0s$ proportionnelle avec le mouvement du sous-marin pour atteindre les $5m$ (figure (4.26)), l'erreur à temps $= 14.1s$ augmente jusqu'à 0.15 lorsque le robot atteint la profondeur $z = -1.84m$, et à temps $= 16s$ lorsque le drone touche $-2m$ alors l'erreur s'élimine complétement.

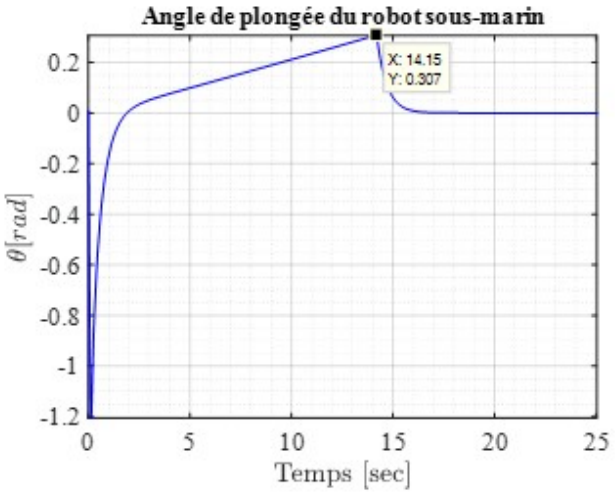

FIGURE 4.28 – Angle de plongé

La figure (4.28) représente l'angle de plongé de notre drone sous-marin par rapport au temps, commençant par un angle initial à Temps $= 0s$, $\Theta = -1.2rad$ et atteindre l'angle maximal à $\Theta = 0.307rad$, Temps $= 14.15s$, puis se stabilise sur $\Theta = 0$ à Temps $= 16s$ lorsque le robot atteint ses $2m$ de profondeur.

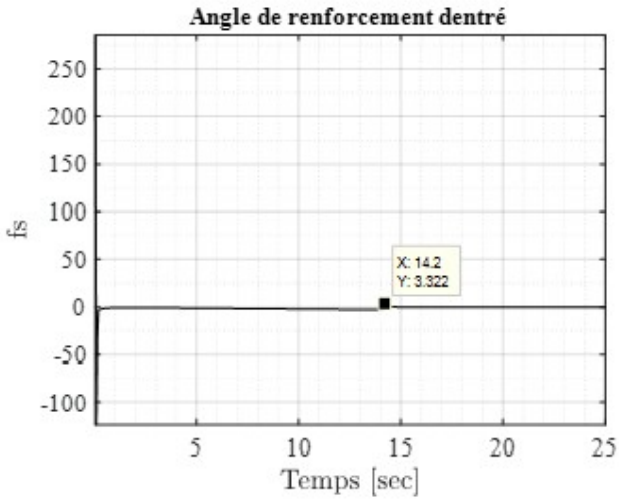

FIGURE 4.29 – Angle de renforcement d'entré

La figure (4.29) représente le signal de commande de notre drone sous-marin par rapport au temps, commençant par une valeur de f_s initiale est nul à Temps = $0s$, et atteindre l'angle maximal à $f_s = 3.322rad$, Temps = $14.2s$, puis se stabilise lorsque le f_s s'annule encore à Temps = $15s$.

Nous allons garder même profondeur désiré $z = 2m$ avec l'angle de plongé $\Theta_d = \frac{\pi}{4}$ avec $U = 4m/s$.

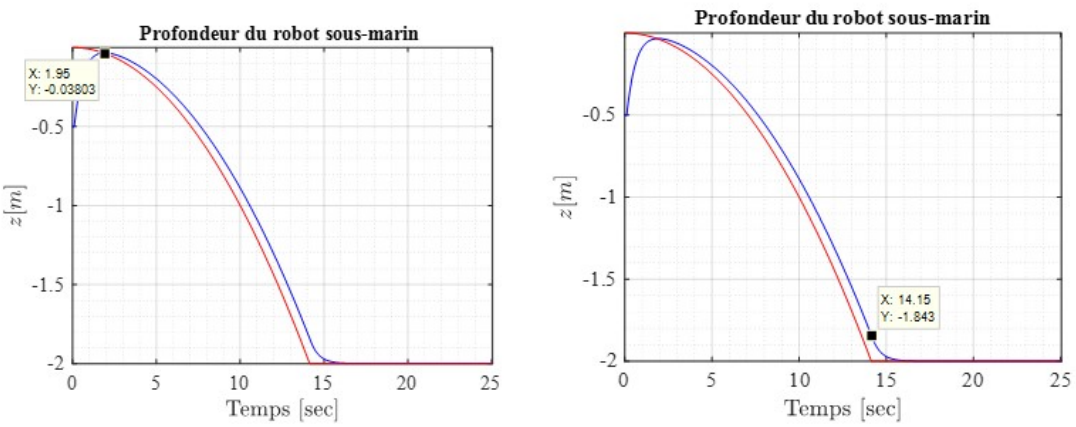

FIGURE 4.30 – Profondeur du robot sous marin

La figure (4.30) représente le chemin de notre drone sous-marin par rapport au temps pour atteindre la profondeur $2m$, commençant par une profondeur initiale à l'instant = $0s$, $z = -0.5m$ suivant le chemin de la trajectoire parabolique (graphe rouge) comme référence attrapé à temps = $2s$, puis notre robot converge vers cette référence afin d'atteindre la profondeur désiré $z = -2m$, à t= 15.7.

Nous avons obtenu même graphe d'erreur que la figure (4.27).

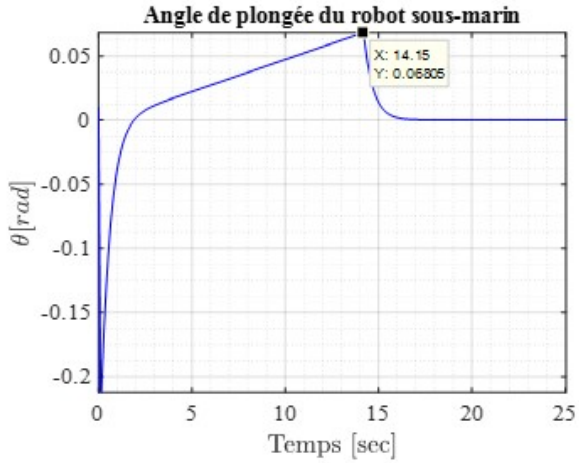

FIGURE 4.31 – Angle de plongé

La figure (4.31) représente l'angle de plongé de notre drone sous-marin par rapport au temps, commençant par un angle initial à Temps= $0s$, $tTheta = -0.22rad$ et atteindre l'angle maximal à $\Theta = 0.068rad$, Temps = $14.15s$, puis se stabilise sur $\Theta = 0$ à Temps = $16s$ lorsque le robot atteint ses $2m$ de profondeur.

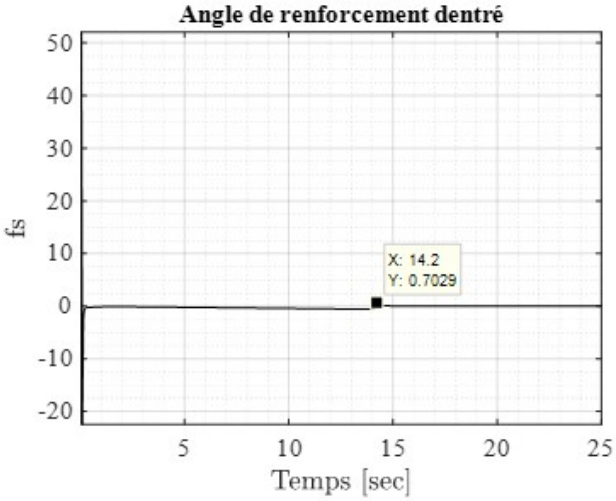

FIGURE 4.32 – Angle de renforcement d'entré

La figure (4.32) représente le signal de commande de notre drone sous-marin par rapport au temps, commençant par une valeur de f_s initiale est nul à Temps = $0s$, et atteindre l'angle maximal à $f_s = 0.7029rad$, t= $14.2s$, puis se stabilise lorsque le f_s s'annule encore à t= $15s$.

Interprétation des résultats obtenus : Nous avons constaté que la surtension commande l'angle de renforcement d'entrée f_s, et l'angle de plongé du robot pour nager jusqu'à la profondeur que nous souhaite z_d.

4.13 Conclusion

Nous constatons que les résultats d'application des commandes par mode glissant sur notre sous-marin confirment que ce réglage présente une réponse qui assure à la fois la robustesse contre le changement des paramètres ou des perturbations, la performance et la simplicité.

Conclusion générale

Dans ce projet de fin d'étude, nous nous sommes menés à une étude détaillée visant à structurer un drone sous-marin automatisé, satisfaisant éventuellement à des applications civiles. En effet, pour réaliser cet objectif, nous avons divisé notre travail en trois parties principales.

En premier lieu nous avons abordé l'étude du matériel électronique nécessaire pour la réalisation d'un tel système, et cela inclus la carte Arduino UNO, les moteurs à courant continu et le moteur pas à pas. Pour la partie mécanique, nous avons modélisé notre robot sous-marin en utilisant Autocad.

Ensuite nous avons effectué l'étude software qui contient les logiciels nécessaires pour la conception de notre robot, notamment l'IDE d'Arduino, Proteus, Autocad, et Matlab.

Vu que l'objectif d'un robot sous-marin est de se positionner dans un espace Euclidien (profondeur, longueur et largeur), nous avons présenté et proposé dans la dernière partie de notre travail une commande nonlinéaire du robot en faisant appel à la commande par les modes glissants. La synthèse de cette commande a été appuyée par une analyse détaillée de la stabilité, et de la convergence en utilisant la théorie de Lyapunov. L'application numérique a donné des résultats satisfaisants ce qui a prouvé l'étude théorique de la synthèse de la commande.

En effet, après plusieurs simulations effectuées pour de différentes situations (différentes profondeurs et angles de plongée), l'application de la commande par les modes glissants a conduit à des bonnes performances telles que la robustesse par rapport aux variations des paramètres du système.

Pour les travaux futurs, nous suggérons de :

1. Mettre en œuvre le modèle proposé avec les modifications nécessaires adaptées à l'environnement choisi.

2. Tester la commande synthétisée en temps réel en incluant les techniques d'observations pour la grandeur de la profondeur.

3. Apporter une amélioration technologique pour la partie matérielle pour adapter le robot sous-marin aux différentes applications : vision, communication sous-marine, ... etc

Bibliographie

[1] Fares Nafa, Labiod Salim, Chekireb Hachemi, *Contribution à la commande d'une classe de systemes mécaniques sous-actionnés*, thèse de doctorat , Alger, École Nationale Polytechnique : 2014.

[2] L. LAPIERRE, *Underwater Robots Part I : Current Systems and Problem Pose* : towards New Applications, Aleksandar Lazinica, ED.London : IntechOpen, pp 310-334, 1 Décembre 2006.

[3] A.KHADHRAOUI, *Modélisation et simulation interactive pour la navigation d'un robot sous-marin de type ROV Observer*, thèse de doctorat en Mathématiques et Informatique, L'école polytechnique de Tunisie et de l'université Paris-Saclay, France, 19 Novembre 2015.

[4] S. BEN SAAD, *Conception d'un algorithme de coordination hybride de groupes de robots sous-marins communicants*, thèse de doctorat, Université de Bretagne Occidentale, Bretagne, 2016.

[5] *Sous-marin de commerce*, https://fr.wikipedia.org/wiki/Sous-marin_de_commerce, 2 Février 2020 consulté le 15 Mars 2020.

[6] *Océanographie*, https://fr.wikipedia.org/wiki/Océanographie, 14 Janvier 2020 consulté le 16 Mars 2020.

[7] *Sous-marin de Sauvetage*, https://fr.wikipedia.org/wiki/Sousmarin_de_sauvetage, 23 Novembre 2019 consulté le 16 Mars 2020.

[8] *Narco sous-marin*, https://fr.wikipedia.org/wiki/Narco_sous-marin, 20 Décembre 2019 consulté le 16 Mars 2020.

[9] *Sous-marin*, https://fr.wikipedia.org/wiki/Sous-marin, 13 Mars 2020 consulté le 16 Mars 2020.

[10] *Les sous-marin nucléaire lanceur d'engins*, https://www.defense.gouv.fr/english/marine/intervention/sous-marins/sous-marins-nucleaires-lanceurs-d-engins/les-sous-marins-nucleaires-lanceurs-d-engins, 3 Jan 2020 consulté le 17 Mars 2020.

[11] *Véhicule sous-marin téléguidé*, https://fr.wikipedia.org/wiki/Véhicule_sousmarin_téléguidé, 2 Février 2020 consulté le 15 Mars 2020.

[12] *Deutschland sous-marin*, https://fr.wikipedia.org/wiki/Deutschland_(sous-marin)#SM_U-155, 2 Juillet 2019 consulté le 16 Mars 2020.

[13] *DSRV-1 Mystic*, ttps://en.wikipedia.org/wiki/DSRV-1_Mystic, 4 Mars 2020 consulté le 18 Mars 2020.

[14] *OMER sous-marin à propulsion humaine* , http://www.etsmtl.ca/activites-et-services-aux-etudiants/clubs-etudiants/omer, 2017 consulté le 17 Mars 2020.

[15] M.VALIN, *A quoi ressembleront les sous-marins du futur ?*, https://www.science-et-vie.com/questions-reponses/a-quoi-ressembleront-les-sous-marins-du-futur-54592, 18 Février 2020 consulté le 17 Mars 2020.

[16] *Arduino UNO*, https://components101.com/microcontrollers/arduino-uno, 28 Février 2018 consulté le 27 Juin 2020.

[17] *ATmega328*, https://www.microchip.com/wwwproducts/en/ATmega328, consulté le 27 Juin 2020.

[18] *DC Motors*, https://www.electronics-tutorials.ws/io/io_7.html, Mai 2019 consulté le 29 Juin 2020.

[19] DR. FRITZ FAULHABER GMBH et CO. KG, *Motor Calculations for Coreless Brush DC Motors*, https://www.faulhaber.com/en/support/technical-support/motors/tutorials/dc-motor-tutorial-dc-motor-calculation, Mai 2020 consulté le 29 Juin 2020.

[20] *Les Avantages Du Moteur En Courant Continu*, http://www.stipaf.com/fr/products-sep/les-avantages-du-moteur-en-courant-continu, 2014 consulté le 29 Juillet 2020.

[21] C. FIORE, *Stepper Motors Basics : Types, Uses, and Working Principles*, https://www.monolithicpower.com/en/stepper-motors-basics-types-uses, 2020 consulté le 29 Juillet 2020.

[22] *L293x Quadruple Half-H Drivers*, https://www.ti.com/lit/ds/symlink/l293.pdf, Janvier 2016 consulté le 29 Juillet 2020.

[23] A. AQEEL, *Introduction to Arduino IDE*, https://www.theengineeringprojects.com/2018/10/introduction-to-arduino-ide.html, 3 Octobre 2018 consulté le 26 Juillet 2020.

[24] *Proteus PCB Design and Simulation Software – Introduction*, https://www.circuitstoday.com/proteus-software-introduction, 2016 consulté le 9 août 2020.

[25] *What is AutoCad*, https://www.autodesk.com/products/autocad/overview?plc=ACDIST&term=1-YEAR&support=ADVANCED&quantity=1, 2020 consulté le 10 Septembre 2020.

[26] Y.ZAOUAOUI, *Commande par mode glissant des courants statoriques de la machine asynchrone statoriques de la machine asynchrone*, Magister en Automatique, Université FERHAT ABBAS, Setif, 3 Mars 2010.

[27] S. VAHID, K. JAVANMARD, *Modeling and Control of Autonomous Underwater Vehicle (AUV) In Heading and Depth Attitude via PPD Controller with State Feedback*, INTERNATIONAL JOURNAL OF COASTAL AND OFFSHORE ENGINEERING, vol. pp12-14, 15 Décembre 2016.

[28] M. SAHRAOUI, *Commandes non linéaires de la machine asynchrone*, Master en génie électrique, Université de M'SILA, 2013.

Annexe 1

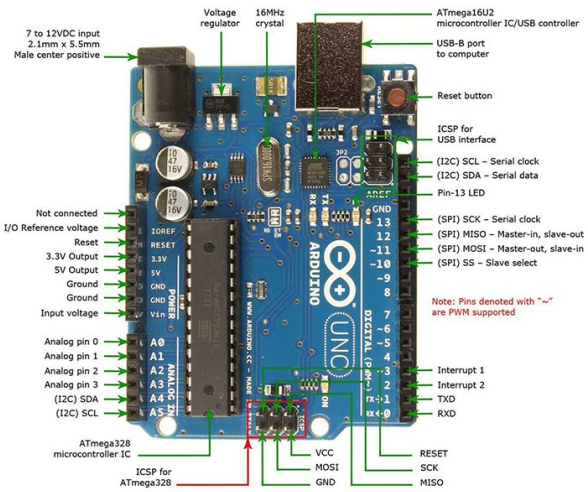

FIGURE 33 – Carte Arduino UNO R3 avec Microcontrôleur ATmega328

On va définir le rôle de chaque broche de la carte Arduino Uno :

⋆ V_{in} : tension d'entrée vers Arduino lors de l'utilisation d'une source d'alimentation externe.

⋆ **5V** : alimentation régulée utilisée pour alimenter le microcontrôleur et d'autres composants de la carte stable grâce au régulateur de tension.

⋆ 3.3**V** : alimentation de 3, 3 V générée par le régulateur de tension embarqué. La consommation maximale de courant est de 50 mA.

⋆ **GND** : broches de masse.

⋆ **Reset** : réinitialise le microcontrôleur.

⋆ **Broches analogiques A**0 **à A**5 : utilisé pour fournir une entrée analogique dans la plage de 0 à 5V tel que :

▹ **Les entrés A**4 **(SDA), A**5 **(SCA)** : utilisé pour la communication TWI.

⋆ **Broches numériques de** 0 **à** 13 : broches d'entrée / sortie, utilisé comme broches d'entrée ou de sortie tel que :

▹ **Les broches** 0 **(Rx),** 1 **(Tx) en série** : utilisé pour recevoir et transmettre des données série TTL. Ils sont connectés à la puce série ATmega328P USB vers TTL correspondante.

▹ **Les broches** 2 **et** 3 : interruptions externes pour déclencher une interruption sur une valeur basse, un front montant ou descendant, ou un changement de valeur.

▹ **Les broches** 3**,** 5**,** 6**,** 9 **et** 11 : fournissent une sortie PWM 8 bits en utilisant la fonction analogWrite ().

▹ **Les broches** 10 **(SS),** 11 **(MOSI),** 12 **(MISO) et** 13 **(SCK)** :Utilisé pour la communication SPI.

⋆ **La** 13**ème broche** : pour allumer la LED intégrée lorsqu'elle est HIGH la LED est allumée lorsqu'elle est LOW, LED est éteinte.

Arduino peut être utilisé pour communiquer avec un ordinateur, une autre carte Arduino ou d'autres microcontrôleurs. Le microcontrôleur ATmega328P fournit une communication série UART TTL (5V) qui peut être effectuée à l'aide de la broche numérique 0 (Rx) et de la broche numérique 1 (Tx). Un ATmega16U2 sur la carte canalise cette communication série via USB et apparaît comme un port de communication virtuel vers le logiciel de l'ordinateur. Le micro logiciel ATmega16U2 utilise les pilotes COM USB standard et aucun pilote externe n'est nécessaire. Cependant, sous Windows, un fichier .inf est requis.

Le logiciel Arduino comprend un moniteur série qui permet d'envoyer des données textuelles simples vers et depuis la carte Arduino. Il y a deux LED RX et TX sur la carte Arduino qui clignotent lorsque les données sont transmises via la puce USB-série et la connexion USB à l'ordinateur (pas pour la communication série sur les broches 0 et 1).

Une bibliothèque SoftwareSerial permet la communication série sur n'importe laquelle des broches numériques de l'Uno. L'ATmega328P prend également en charge la communication I2C (TWI) et SPI. Le logiciel Arduino comprend une bibliothèque Wire pour simplifier l'utilisation du bus I2C.