Subsurface: conception d'un AUV lowcost

Équipe 1 : Nérée
Thomas AGUIRRE, Michel BILAIN, Solène BOUDOT, Nicolas DAMAGEUX,
Ewen MELE, Léon PERRUCHOT-TRIBOULET, Gabin TEINTURIER
Équipe lèves ingénieurs roboticiens promotion FISE 26
ENSTA - Campus Brest

***Résumé*—Cet article présente la conception et les premières expérimentations d'un AUV (Autonomous Underwater Vehicle). Après une phase de conception mécanique et de définition de l'architecture électronique, une attention particulière a été portée à l'équilibrage statique et dynamique du robot afin d'assurer sa stabilité et son bon fonctionnement en plongée. Différentes missions de navigation autonome ont ensuite été testées, d'abord en simulation puis en environnement réel, notamment des tâches de suivi de cap et de points de passage.**

Des expérimentations en conditions réelles ont permis de valider la conception du système dans un environnement contraint. Dans ces conditions, la capacité du robot à réaliser des missions simples a été évaluée. Malgré certaines difficultés rencontrées avec quelques capteurs, les premiers essais expérimentaux se sont révélés concluants.

I. Introduction

Ces dernières années, la robotique marine a connu un réel développement grâce à ses nombreuses applications, que ce soit pour l'exploration, l'inspection d'infrastructures ou pour la défense, en permettant de s'aventurer dans des environnements dangereux et souvent difficiles d'accès pour l'homme. L'ENSTA, avec sa position géographique favorable grâce à son campus brestois, s'inscrit dans cette dynamique et travaille continuellement sur le développement de nouveaux robots sous-marins. Cependant, bien qu'il existe de nombreuses plateformes robotisées aquatiques, la plupart présentent un coût élevé, souvent de l'ordre de plusieurs milliers d'euros, ce qui peut se révéler être un frein pour l'élaboration d'essaims de robots.

C'est dans ce contexte que nous avons entrepris la conception et la réalisation d'un robot sous-marin autonome (AUV) à faible coût, de type torpille (Fig. 1), destiné à l'exécution de missions simples. Dans un premier temps, nous nous sommes focalisés sur l'architecture mécanique avec la conception mécanique du robot, la fabrication de pièces et de l'armature globale du robot, avant de nous tourner vers l'architecture électronique du système et l'intégration des différents capteurs. En parallèle, nous avons travaillé sur l'équilibrage statique et dynamique du robot pour la remontée "à plat" et le bon fonctionnement de la plongée du robot. Grâce à un travail en simulation, nous avons pu tester différentes missions de navigation autonome comme l'asservissement en profondeur.

La finalité de ce travail est la réalisation d'une campagne d'expérimentations sur le plan d'eau de la darse de Quélern, à l'École Navale de Lanvéoc, où nous avons eu l'opportunité de confronter la plateforme à plusieurs missions simples telles

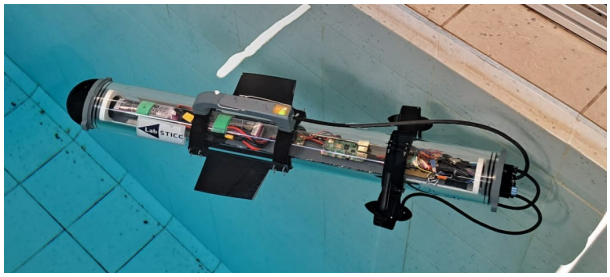


FIGURE 1. Photo du subsurface dans la piscine de l'ENSTA – Campus de Brest

que l'activation des moteurs à une certaine profondeur ou encore la plongée contrôlée du robot.

Les principales contributions de cet article sont les suivantes :

- la conception et la réalisation d'une plateforme AUV à faible coût reposant largement sur des techniques de fabrication par impression 3D
- le développement d'une architecture électronique et logicielle modulaire adaptée à l'intégration de capteurs et d'algorithmes de contrôle
- l'établissement d'un modèle dynamique simplifié du robot permettant la conception de contrôleurs basiques pour la navigation
- la mise en place d'un environnement de simulation permettant le développement et la validation des algorithmes de contrôle avant les tests expérimentaux

Dans cet article, nous retracerons donc l'ensemble de ce travail, en commençant par une description complète de l'architecture du robot (partie II) puis nous établirons un modèle dynamique simple (partie III) et étayerons le développement de la simulation (partie IV) avant de terminer avec un aperçu des tests réels (partie V).

II. Description de la plateforme

La conception du robot suit plusieurs critères, dont le premier est son coût, ce dernier devant être faible. En effet, comme il doit pouvoir être déployé en essaim, une limitation du prix permet l'obtention d'un plus grand nombre de robots. Chaque robot doit ête de petite taille (moins d'un mètre de long) mais suffisamment grand pour avoir une bonne tenue en mer.

Le robot doit aussi permettre une utilisation facilitée pour une mise en service rapide. Ainsi le robot doit avoir non seule-

TABLE I
CARACTÉRISTIQUES DU SUBSURFACE

Paramètre	Valeur
Longueur totale	0.9 m
Diamètre du tube	0.3 m
Masse	8.6 kg
Nombre de propulseurs	2 (T60)
Configuration	Propulsion différentielle
Batteries	2 × LiPo 3S 5000 mAh
Ordinateur embarqué	Raspberry Pi 4
Microcontrôleur	ESP32
Capteurs	Bar30, Artemis IMU, M10Q GNSS
Fabrication	Impression 3D (SLA/FDM)

ment des batteries amovibles mais un système de lancement de mission efficace.

Enfin, le robot doit pouvoir être modulaire. Dans la mesure où la version présentée ici du Subsurface n'est que le prototype initial, il est nécessaire d'anticiper la suite du projet en permettant le rajout de différents capteurs ou systèmes de communication.

Les principales caractéristiques physiques du robot sont résumées dans le tableau I.

A. Architecture mécanique

On retrouve d'abord sur l'extérieur du robot, à l'arrière, deux propulseurs T60 de part et d'autre du robot pour permettre une propulsion différentielle.

Ensuite, est monté un système d'ailes à voilure fixe. Ces ailes servent deux fonctions : la première est de limiter le roulis du subsurface et la deuxième, plus importante, est le contrôle de la force de plongée du robot. La vitesse d'avance du robot étant directement liée à la vitesse de plongée via la portance générée par les ailes, ces dernières constituent un élément essentiel du système. De plus le fait de limiter l'utilisation d'actionneurs exposés à l'eau permet de réduire la complexité mécanique et le coût du robot de manière significative.

Le robot possède aussi une poignée contenant un récepteur GNSS. Cette poignée facilite la mise à l'eau ou la sortie du robot ce qui répond à l'objectif d'aisance d'usage et permet de réhausser le récepteur GNSS afin de pouvoir capter plus aisément ses signaux lors d'une immersion partielle.

Enfin le robot possède à l'avant un amortisseur pour limiter les chocs en cas de contact, et deux capots étanches de part et d'autre du tube pour permettre l'étanchéité de l'intérieur.

À l'intérieur du robot nous retrouvons l'électronique du robot. L'électronique est montée sur une plaque en PVC, elle même soutenue par deux entretoises à l'entrée et la sortie du tube. Ces entretoises peuvent contenir des billes de plomb pour alourdir l'AUV, et s'insèrent par le biais de goupilles dans les capots étanches afin de garantir l'orientation horizontale de l'intérieur du robot.

Pour répondre à l'objectif de coûts minimaux, l'ensemble des pièces du robot (à l'exception du tube en plexiglas et de la plaque de support intérieure) a été fabriqué à l'aide d'imprimantes 3D. Les capots étanches et la poignée ont été fabriqués à l'aide d'une imprimante SLA en utilisant

FIGURE 2. CAO du robot. En orange le tube en acrylique, en gris les pièces par SLA et en noir les pièces par FDM, l'amortisseur n'est pas visible

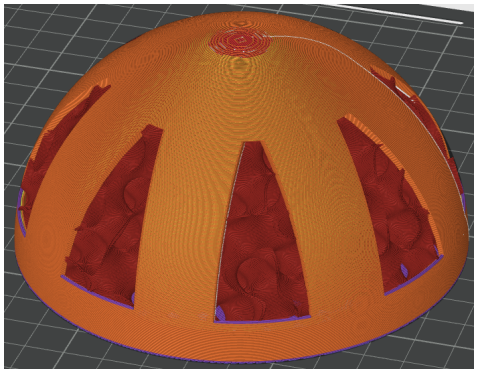


FIGURE 3. Amortisseur avant, imprimé en TPU. Le remplissage romboïde à 10% permet l'élasticité et la tenue mécanique

de la résine durcissant aux rayonnements UV. L'impression résine permet d'obtenir des tolérances de l'ordre du dixième de millimètre, essentiel pour garantir l'étanchéité du robot. Les autres pièces telles que les ailes, les supports pour les autres pièces et l'amortisseur sont-elles réalisées à partir d'une imprimante FDM en PETG pour les pièces extérieures, PLA pour les entretoises intérieurs et TPU pour l'amortisseur. Dans la mesure où l'étanchéité des pièces imprimées par dépôt de filament ne peut être garantie, nous avons percé l'amortisseur pour y laisser entrer l'eau et avons enduit de résine les ailes pour les rendre étanches et essayer d'obtenir un meilleur état de surface.

Enfin pour alourdir le robot afin de le rendre à peine flottant, une entretoise en béton a été coulée entre les deux entretoises de support. La masse ajoutée par le béton est complétée par l'ajout de sachets de billes de plomb insérées dans les compartiments des entretoises plastiques.

B. Architecture électronique

L'architecture électronique du robot doit également répondre à des contraintes de faible coût, de facilité d'utilisation et de modularité. Pour ces différentes raisons, nous utilisons deux cartes électroniques complémentaires : une Raspberry Pi 4 et un ESP32. La Raspberry Pi 4 est un ordinateur monocarte disposant de capacités de calcul importantes et exécutant un système d'exploitation Linux. Elle permet d'implémenter les

fonctions de haut niveau telles que la gestion de la navigation, le traitement des données capteurs ou la communication avec l'utilisateur. L'ESP32 est un microcontrôleur embarqué particulièrement adapté aux tâches temps réel et à l'interfaçage direct avec le matériel. Il est utilisé pour la lecture des capteurs et le contrôle des actionneurs. Cette séparation entre calcul haut niveau et contrôle bas niveau facilite la modularité de la plateforme, notamment en permettant l'ajout futur de capteurs ou de nouvelles fonctionnalités logicielles sans modification majeure de l'architecture existante.

Le robot est également équipé de plusieurs capteurs nécessaires à la navigation. Il intègre un baromètre de pression absolue Blue Robotics Bar30, une centrale inertielle SparkFun Artemis IMU ainsi qu'un récepteur GNSS Matek M10Q-5883. Dans ce prototype, seuls les capteurs strictement nécessaires à la navigation sont utilisés. Toutefois, l'architecture matérielle et logicielle de la plateforme permet l'intégration ultérieure de capteurs supplémentaires.

L'alimentation électrique du robot repose sur deux batteries LiPo 3S de 5000 mAh. La première est dédiée à l'alimentation des cartes électroniques et des capteurs, via un convertisseur de tension THL 20-4815WI. La seconde est utilisée pour alimenter les propulseurs.

Enfin, le contrôle des deux propulseurs, est assuré par deux contrôleurs électroniques de vitesse BESC30-R3, commandés par un signal PWM.

L'ensemble de l'architecture électronique est résumé dans la figure 4.

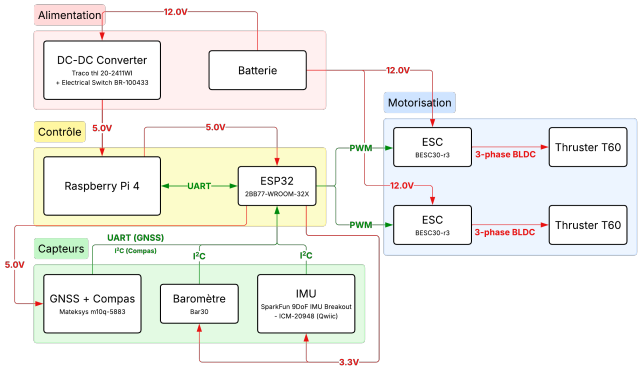


FIGURE 4. Architecture électronique, alimentation électrique et signaux

C. Architecture logicielle

Pour l'architecture logicielle, nous distinguons les codes présents sur la Raspberry Pi de ceux de l'ESP32.

L'ESP32 est chargée de tout l'interfaçage bas-niveau. Ainsi c'est elle qui communique avec les capteurs, envoie les commandes moteurs et peut couper l'alimentation des moteurs pour garantir une sécurité de navigation. La communication avec la Raspberry Pi est assuré via une liaison UART, c'est par elle que le contrôleur haut niveau reçoit les mesures capteurs et envoie les consignes moteur.

La Raspberry Pi, quant à elle, sert d'ordinateur principal et permet un contrôle haut niveau du robot. Elle exécute Raspberry Pi OS au sein duquel une image Docker basée sur Ubuntu 22.04 héberge le middleware ROS2 Humble. Cette solution, que nous avons retenue pour ce prototype initial, a pour avantage de permettre une intégration simple dans l'environnement de simulation (notamment avec des membres de l'équipe travaillant sur des systèmes d'exploitation différents) bien qu'elle présente des complexités d'usage sur le terrain.

En cadre opérationnel le téléversement des codes sur la Raspberry Pi est réalisée par liaison SSH et le lancement des codes se fait simplement en lançant l'image Docker. Une ergonomie améliorée tel que l'autocompilation et envoie du firmware sur l'ESP32 en cas de changement ou un démarrage rapide de l'ensemble du robot sont à envisager. A noter que le robot est déjà muni d'une fonction d'activation de la mission choisie. Lors de la mise à l'eau une immersion à plus de vingt centimètre de profondeur pendant cinq secondes lance la mission.

III. MODÉLISATION PHYSIQUE

Nous allons établir les équations régissant la physique de notre système. Nous plaçons le robot dans un repère monde $\mathcal{M} = (O_\mathcal{M}, \mathbf{i}_x, \mathbf{i}_y, \mathbf{i}_z)$ ayant pour origine la position initiale du robot et orienté suivant la convention NED et le munissons de son propre repère $\mathcal{B} = (O_\mathcal{B}, \mathbf{e}_x, \mathbf{e}_y, \mathbf{e}_z)$ attaché au robot suivant la même convention et illustré en figure 5. Nous négligeons certains effets hydrodynamiques, notamment la masse ajoutée, afin d'obtenir un modèle simplifié dans le lequel le robot n'est soumis qu'aux forces suivantes :

- La force des propulseurs : $\mathbf{F}_p = F_p \mathbf{e}_x$
- La force de traînée : $\mathbf{F}_t = -\frac{1}{2}\rho C_x S|u|u\mathbf{e}_x$
- Le poids du robot : $\mathbf{P} = mg\mathbf{e}_z$
- La poussée d'Archimède : $\mathbf{F}_A = -\rho V g \mathbf{e}_z$
- La force de coulée du robot : $\mathbf{F}_c = \frac{1}{2}\rho C_z S|w|w\mathbf{e}_z$

Avec m la masse du robot, g la constante de pesanteur, ρ la masse volumique de l'eau, V le volume du robot, v la vitesse du robot et u, w ses composantes projetées selon les axes $\mathbf{e}_x$ et $\mathbf{e}_z$, les coefficients de frottement visqueux C_x, C_z, et S la surface apparente du robot selon $\mathbf{e}_x$.

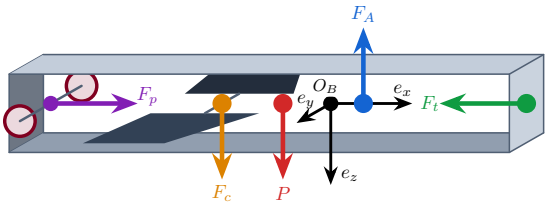


FIGURE 5. Schéma des forces appliquées au robot dans le cas d'un robot déséquilibré

Supposons que les équilibrages statique et dynamique du robot sont réalisés (cf partie V). On peut obtenir d'après le principe fondamental de la dynamique appliqué au centre de

masse du robot dans $\mathcal{M}$ projeté selon l'axe $\mathbf{e}_z$ l'équation suivante :

$$\begin{aligned} m\dot{w} &= P + F_A + F_c \\ &= mg - \rho V g + \frac{1}{2}\rho C_z S|w|w \end{aligned} \tag{1}$$

On identifie alors une vitesse critique permettant de faire plonger le robot.

$$v_{crit} = \sqrt{\frac{2(F_A - P)}{\rho C_z S}} \tag{2}$$

En-deçà de v_{crit} la force hydrostatique domine, et le robot flotte, au-dessus le robot plonge.

Pour contrôler l'altitude du robot nous appliquons un PID sur la profondeur du robot avec en sortie une commande sur la vitesse des propulseurs. Ce contrôleur simple suffit à obtenir une convergence vers la valeur souhaitée.

Pour contrôler l'orientation du robot nous utilisons la propulsion différentielle telle que $F_p = f_1 + f_2$ et $\tau = d(f_1 - f_2)$ avec τ le couple généré par f_1, f_2 les forces de chaque propulseur et d la distance latérale au centre de poussée. À partir de cette modélisation dynamique simple nous pouvons assimiler le robot à une voiture de Dubins et lui appliquer des contrôleurs robustes et éprouvés.

Il convient de noter que le couplage entre la vitesse d'avance du robot et sa vitesse de coulée rajoute une contrainte sur le contrôle du robot qu'il est nécessaire d'étudier afin d'obtenir des commandes optimales et un contrôle fin du robot. Nous n'avons pas effectué ce travail qu'il reste à faire.

IV. Simulation

Afin de tester les capacités du robot et d'implémenter de premiers algorithmes de contrôle, nous avons développé une simulation sur Unity. Nous utilisons les fonctions prédéfinies du moteur physique pour gérer la plupart des forces et n'avons qu'à déterminer leur point d'application.

Nous avons tout de même dû implémenter la poussée d'Archimède et la force de coulée manuellement. Pour la pousée d'Archimède nous établissons un maillage du corps du robot sous la forme de sphères élémentaires qui sont rigidement attaché au robot. Ensuite nous déterminons pour chaque maille élémentaire la poussée d'Archimède ressentie en fonction de la profondeur de celle-ci à l'aide de la fonction suivante :

$$F_A(z) = \begin{cases} 0 & z \leq -R \\ \rho g \dfrac{\pi(R+z)^2(2R-z)}{3} & -R < z < R \\ \rho g \dfrac{4}{3}\pi R^3 & z \geq R \end{cases} \tag{3}$$

avec z la profondeur de la sphère, R le rayon de la sphère, ρ la masse volumique du fluide et g la pesanteur.

Les forces sont ensuite appliquées au centre de chaque maille et le moteur physique du simulateur determine les effets sur le corps du robot. Pour la force de coulée la relation est plus directe, nous appliquons directement la force telle qu'exprimée dans la partie III au centre de masse. La figure 6 représente une vue de debogage de la simulation.

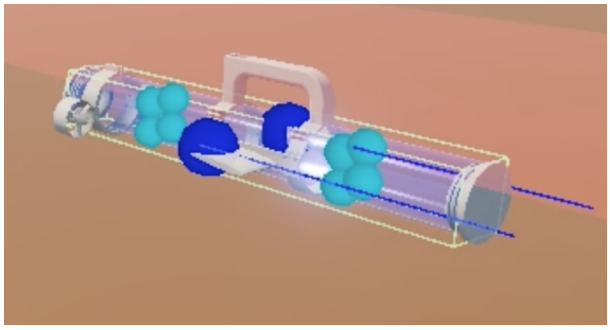


FIGURE 6. Représentation de quelques sphères de flottaison et des forces des propulseurs

La décomposition du corps du robot sous la forme d'une maille de sphères n'est pas une modélisation exacte mais permet déjà d'obtenir un compromis efficace entre précision du modèle et temps de calcul.

À partir de la simulation nous avons pu valider notre architecture logicielle en intégrants nos capteurs simulés dans la boucle de contrôle. Ainsi nous avons réalisé un asservissement en profondeur du robot à l'aide un PID sur l'altitude du robot en envoyant des commandes en vitesse d'avance. Après ajustement des paramètres, les résultats obtenus étaient très satisfaisant avec une convergence en quelques secondes pour des variations de profondeur de l'ordre de cinq mètres.

FIGURE 7. Profondeur du robot en fonction du temps affiché dans l'environnement de simulation

D'autres algorithmes ont commencé à être implémentés mais n'ont pas abouti, comme le suivi de points de contrôle GNSS ou le dead reckoning pour tenter de localiser le robot lors des phases de plongée longue durée. Ces embryons d'algorithme mettent en évidence l'utilité de la simulation pour l'implémentation des contrôleurs.

V. Tests réels

Après avoir construit le robot et développé l'environnement de simulation nous avons réalisé quelques tests pour valider expérimentalement le concept du robot.

Une première étape est l'équilibrage du robot. Le but de l'équilibrage est de confondre au même point le centre de masse et le centre de poussée du robot afin d'obtenir un robot se déplaçant horizontalement sans tangage dans l'eau. Il est

nécessaire de distinguer deux formes d'équilibrage statique et dynamique. Dans le premier cas nous cherchons à anéantir le couple exercé par la poussée d'Archimède sur le robot. Pour ce faire nous faisons varier la masse de quatre sachets de billes de plomb placées à l'avant et à l'arrière du robot dans les entretoises de support de la plaque.

À partir de l'équilibre statique il devient possible de réaliser un équilibrage dynamique qui consiste en l'ajustement de la position des ailes pour faire coïncider le centre de poussée et le centre de masse. Ainsi lors de l'avance du robot et lors de la plongée le robot garde une assiette constante.

Nous avons pu, par la suite, réaliser une campagne d'essai à Lanvéoc dans les bassins de l'Ecole Navale. Ces tests ont permis de mettre au jour de nombreux axes d'amélioration pour consolider les fonctions du robot. Il apparaît désormais nécessaire de fabriquer une carte de circuit imprimé pour rendre les connexions électroniques plus stables. Ces essais permettent néanmoins de valider la preuve de concept du système, avec un robot capable de plonger en avançant et d'être contrôlable avec les premiers algorithmes simples évoqués précédemment.

VI. Conclusion et perspectives

Cet article a présenté la conception et les premières expérimentations d'une plateforme AUV à faible coût destinée à la réalisation de missions de navigation autonome simples pour la recherche sur les essaim de drones sous-marins. Nous avons décrit l'architecture mécanique, électronique et logicielle du robot, ainsi que le développement d'un modèle dynamique simplifié et d'un environnement de simulation permettant la mise au point des premiers algorithmes de contrôle.

Les essais expérimentaux réalisés ont permis de valider la preuve de concept de la plateforme. Le robot est capable de plonger en utilisant la portance générée par ses ailes et d'exécuter des missions de navigation simples à l'aide de contrôleurs basiques développés en simulation puis embarqués sur la plateforme.

Plusieurs axes d'amélioration ont toutefois été identifiés lors des essais. En particulier, la fiabilité des connexions électroniques pourrait être améliorée par la conception d'une carte de circuit imprimé dédiée, et le modèle dynamique du robot pourrait être enrichi afin de mieux prendre en compte certains effets hydrodynamiques négligés dans cette première étude.

Enfin, de nouvelles fonctionnalités pourront être intégrées à la plateforme, telles que l'ajout de caméras RGB, de systèmes de radiocommande HF ou de modems acoustiques. Ces évolutions pourraient permettre le développement de capacités de communication et de coopération entre robots en vue d'applications en essaim. Par ailleurs, un protocole de communication optique robuste est actuellement en cours de développement au sein de l'équipe. Ce travail n'a pas pu être présenté dans cet article par manque de place et fera l'objet d'une publication dédiée.

Design and Experimentation of a Low-Cost Autonomous Underwater Vehicle

Hermann Carrier
hermann.carrier@ensta.fr
FISE 3A ROB 2026

Rami Diab
rami.diab@ensta.fr
FISE 3A ROB 2026

Bruno Huntzinger
bruno.huntzinger@ensta.fr
FISE 3A ROB 2026

Hippolyte Lafrad
hippolyte.lafrad@ensta.fr
FISE 3A ROB 2026

Angelo Provot
angelo.provot@ensta.fr
FISE 3A ROB 2026

Matti Soucaille
matti.soucaille@ensta.fr
FISE 3A ROB 2026

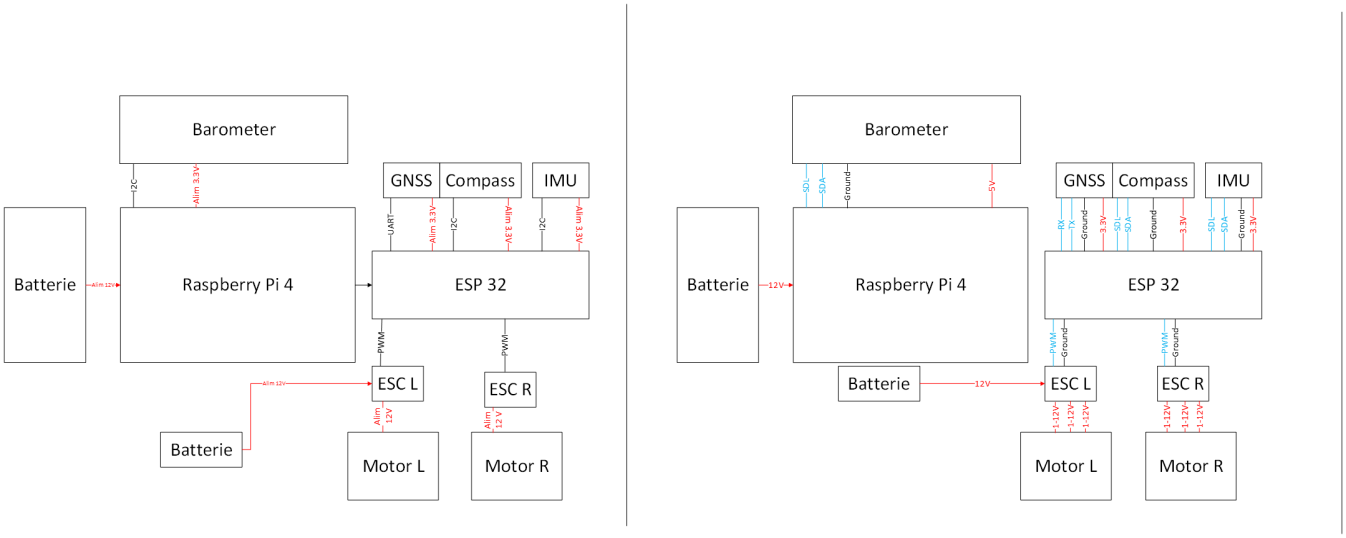


Figure 1: Hardware architecture of the torpedo AUV.

Abstract

This project concerns the design and experimental validation of a low-cost autonomous underwater vehicle (AUV) in a torpedo form factor. The system integrates an embedded electronic architecture based on a Raspberry Pi 4 and an ESP32 microcontroller, and a software architecture relying on ROS2 Humble. The goal is to enable autonomous underwater navigation using onboard sensors, proportional control algorithms, and a Unity-based simulation environment. Four missions were defined; the first two : straight-line underwater propulsion and GPS waypoint following, were successfully validated in real conditions. Underwater ArUco marker detection was also implemented but could not be tested in a pool environment.

Keywords

Autonomous Underwater Vehicle, ROS2, GPS waypoint navigation, depth control, ArUco detection, Unity simulation, torpedo AUV

ACM Reference Format:
Hermann Carrier, Rami Diab, Bruno Huntzinger, Hippolyte Lafrad, Angelo Provot, and Matti Soucaille. 2026. Design and Experimentation of a Low-Cost Autonomous Underwater Vehicle. In *Proceedings of (Research Initiation)*. ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

Research Initiation, ENSTA Bretagne

ACM ISBN 978-x-xxxx-xxxx-x/YYYY/MM
https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 Introduction

Autonomous Underwater Vehicles (AUVs) represent a rapidly growing field of mobile robotics, with applications ranging from oceanographic exploration and environmental monitoring to pipeline inspection and search-and-rescue operations [6]. The design of low-cost AUVs is of particular interest for academic research, as it allows the development and validation

of control and perception algorithms without the financial constraints associated with commercial platforms [].

In this project, we design, build, and test a miniature torpedo-shaped AUV equipped with two rear-mounted thrusters, an inertial measurement unit (IMU), a GNSS receiver, a depth sensor, and a side-facing camera. The embedded software stack is built on ROS2 Humble, running on a Raspberry Pi 4 that communicates with an ESP32 microcontroller via UART. Four autonomous missions were defined to progressively validate the system: (1) straight-line underwater propulsion, (2) heading following, (3) GPS waypoint following with underwater transit, and (4) repeated GPS back-and-forth navigation. This paper describes the full system architecture, the control strategy, the simulation environment, and the results obtained for each mission.

2 Related Work

Low-cost AUV platforms have been the subject of increasing attention in recent years. Zhou et al. [10] describe the development of a low-cost torpedo-shaped AUV based on ROS, demonstrating the faisability of the project while showing the need for simulation. [8] shows how to transform commercial systems such as the BlueROV2 to custom low-cost AUV, highlighting the trade-offs between cost, payload, and autonomy.

For heading estimation, complementary filters fusing gyroscope and magnetometer data are standard in embedded systems using for example the Kalmann filter. [3]. GPS-based waypoint navigation for surface or near-surface AUVs relies on geodesic coordinate transformations to convert global latitude/longitude positions into a local North-East (NE) frame [2]. Depth control using a barometric pressure sensor such as the Blue Robotics Bar30 is a well-established approach for small AUVs [1].

Regarding underwater visual perception, ArUco fiducial markers have been used successfully for AUV localisation in pool environments. Xu et al. [9] demonstrated that placing multiple ArUco markers on the pool floor enables accurate pose estimation after underwater camera calibration. Risholm et al. [7] further addressed the challenge of robust marker detection under varying turbidity and lighting conditions. Image preprocessing techniques such as CLAHE have been shown to significantly improve ArUco detection accuracy in underwater images [11].

Unity has been used as a physics simulation backend for marine robotics [4], and some proposed algorithm in order to obtain a realistic real-time simulation of object floating in the water [5].

3 Methodology

3.1 System Architecture

3.1.1 Mechanical Design. The vehicle adopts a torpedo form with two brushless thrusters that are mounted at the rear of the hull, slightly offset downward from the longitudinal axis, providing both forward thrust and a pitching moment when driven at unequal speeds. A set of two fixed fins is located at mid-body, with a small inclination relative to the horizontal plane. This fin geometry is key to the diving capability of the vehicle: when the thrusters accelerate the vehicle forward, the inclined fins generate a hydrodynamic lift force directed downward, causing the nose to pitch down and the vehicle to dive. The faster the vehicle travels, the greater the downward hydrodynamic force produced by the fins, allowing depth to be controlled primarily through thruster speed rather than active pitch actuators. The hull is sealed for waterproofing, with cable penetrators ensuring watertight electrical connections.

3.1.2 Electronic Architecture. The embedded system is built around two processing units:

- **Raspberry Pi 4** — the main onboard computer running Ubuntu 22.04 and the ROS2 Humble middleware. It executes all high-level control nodes, handles sensor data fusion, and publishes motor commands.
- **ESP32 DevKitC V4** — a low-level microcontroller responsible forinterfacing with the sensors and generating PWM signals for the ESCs. It communicates with the Raspberry Pi via UART.

The main onboard sensors are:

- **IMU**: ICM-20948 (9-axis: accelerometer, gyroscope, magnetometer)
- **GNSS + compass**: M10Q-5883 GPS receiver with integrated magnetometer
- **Depth sensor**: Blue Robotics Bar30 (pressure/depth, 300 m rated) [**?**]
- **Camera**: Logitech Carl Zeiss Tessar 2 MP USB camera

The two thrusters are each driven by an Electronic Speed Controller (ESC) receiving a standard PWM signal (1100–1900 μs) generated by the ESP32. The neutral (zero thrust) value is 1500 μs; values above this correspond to forward thrust, and values below to reverse thrust.

3.1.3 Software Architecture. The software stack is organized as follows:

- **Operating system**: Ubuntu 22.04 LTS
- **Middleware**: ROS2 Humble Hawksbill
- **Workspace**: `/home/ensemble/Documents/IngeSysWS`
- **Main control node**: `ControleTorpille` (C++)

The main ROS2 topics used by the control node are:

- `/imu/data` — IMU measurements (linear acceleration, angular velocity)
- `/gnss/fix` — GNSS position fix (latitude, longitude, altitude)
- `/compass/mag` — magnetometer field vector
- `/bar30/depth` — depth in meters (positive downward)
- `/camera/image_raw` — raw camera frames
- `/cmd_vel/left`, `/cmd_vel/right` — integer PWM commands to the left and right ESCs

All sensor callbacks run at 20 Hz (50 ms timers). A watchdog monitors the freshness of each sensor stream and raises a warning if no valid message is received within 2 seconds, causing the motors to be set to neutral as a safety measure.

3.2 Control Strategy

3.2.1 Heading Estimation. Accurate heading estimation is critical for waypoint navigation. The onboard magnetometer alone is subject to high-frequency noise, while gyroscope integration drifts over time. We therefore employ a simple filter that fuses both sources:

$$\hat{\psi}t = \alpha \left[\hat{\psi}t - 1 \;\; \psi_{\text{gyro}}\,\Delta t\right] \;\; 1 - \alpha\,\psi_{\text{mag}} \tag{1}$$

where $\hat{\psi}$ is the estimated heading (rad), ψ_{gyro} is the gyroscope yaw rate (rad/s), ψ_{mag} is the magnetometer-based heading (rad), Δt is the sampling period (0.05 s), and $\alpha = 0.98$ is the complementary filter coefficient. All angles are normalised to $-\pi, \pi$ using the sawtooth function sawtoothθ = atan2sin $\theta, \cos\theta$.

3.2.2 GPS Coordinate Frame. The GNSS receiver provides global coordinates (latitude, longitude) that must be converted into a local North-East (NE) Cartesian frame centred on the first waypoint [2]. Using the equatorial Earth radius $R = 6\,378\,137$ m:

$$N = \varphi - \varphi_{\text{ref}} \cdot \frac{\pi}{180} \cdot R \tag{2}$$

$$E = \lambda - \lambda_{\text{ref}} \cdot \frac{\pi}{180} \cdot R \cdot \cos\left(\varphi_{\text{ref}} \frac{\pi}{180}\right) \tag{3}$$

where φ and λ are the latitude and longitude of the vehicle, and φ_{ref}, λ_{ref} are those of the reference point.

3.2.3 Heading Control. The desired heading to reach a GPS waypoint is computed as:

$$\psi_d = \text{atan2}E_{\text{wp}} - E,\; N_{\text{wp}} - N \tag{4}$$

A proportional controller calculates a differential thrust command from the heading error e_ψ = sawtooth$\psi_d - \hat{\psi}$:

$$u_R = 1500 \;\; v_{\text{base}} - \text{sat}\left(K_\psi\, e_\psi,\; -\Delta,\; \Delta\right) \tag{5}$$

$$u_L = 1500 \;\; v_{\text{base}} \;\; \text{sat}\left(K_\psi\, e_\psi,\; -\Delta,\; \Delta\right) \tag{6}$$

where u_R and u_L are the PWM commands (μs) for the right and left thrusters, $v_{\text{base}} = 0.4 \times 400 = 160$ is the base forward speed offset, $K_\psi = 400$ is the proportional heading gain, and $\Delta = 200$ is the saturation limit on the differential term. A waypoint is considered reached when the Euclidean distance to the target falls below 5 m.

3.2.4 Depth Control.

Principle of Passive Diving via Hydrodynamic Fins. As described in Section 3.1, the inclined fins produce a downward force proportional to the square of the vehicle's speed. When the thrusters increase speed, the vehicle dives; when they slow down, the restoring buoyancy force brings it back to the surface. This passive diving mechanism means that depth is implicitly coupled to forward speed, which simplifies the control architecture at the cost of independent depth actuation.

Active Depth Control Loop. For finer depth regulation, a proportional depth controller is implemented using the Bar30 sensor readings. The depth error is:

$$e_z = z_d - z_m \tag{7}$$

where z_d is the desired depth (m) and z_m is the measured depth (positive downward). A vertical offset $\delta_z = K_z\, e_z$ is added to both thruster commands, increasing forward speed (and thus hydrodynamic dive force) when the vehicle is shallower than desired:

$$u_{R,L} = 1500 \;\; v_{\text{base}} \;\; \delta_z \pm \text{sat}K_\psi\, e_\psi,\; -\Delta,\; \Delta \tag{8}$$

3.3 Underwater Visual Perception

3.3.1 Camera System. The vehicle is equipped with a forward-facing 2 MP USB camera (Logitech Carl Zeiss Tessar) whose raw frames are published on the `/camera/image_raw` ROS2 topic at the Raspberry Pi level.

3.3.2 ArUco Marker Detection. ArUco fiducial markers are widely used for localisation in structured underwater environments [7, 9]. We implemented an ArUco detection pipeline using OpenCV's `aruco` module that estimates the 6-DoF pose of detected markers relative to the camera. Image preprocessing using CLAHE (Contrast Limited Adaptive Histogram Equalisation) is applied prior to detection, as it has been shown to significantly improve detection accuracy in low-contrast underwater images [11].

The detection pipeline was integrated as a ROS2 node publishing detected marker poses. However, **pool testing of the ArUco pipeline was not possible** within the project timeline, so its performance in real underwater conditions remains to be validated.

3.4 Simulation

3.4.1 Unity Environment. A simulation environment was developed in the Unity game engine to test control algorithms without requiring pool access. The simulation exposes the same ROS2 topics as the real platform (`/imu/data`, `/gnss/fix`, `/bar30/depth`, etc.) via a ROS2-Unity bridge, allowing the full control software to run unmodified.

3.4.2 Buoyancy Modelling. Accurately simulating the buoyancy of the torpedo-shaped hull in Unity is non-trivial because Unity does not natively compute the submerged volume of arbitrary meshes. We adopted a decomposition approach: rectangular boxes of varying sizes are arranged to match the actual hull geometry. The buoyant force applied to each box is proportional to its submerged volume. This approach provides a good approximation of both the total buoyancy force and its centre of application, which determines the vehicle's static stability.

4 Experiments and Results

Two test and then four autonomous missions of increasing complexity were defined to progressively validate the platform.

4.1 Waterproofing and Flotation Tests

Prior to pool trials, the vehicle underwent a series of static tests in a laboratory pool. Waterproofness of all penetrators and end-cap seals was verified before each try in the water and then buoyancy was tuned by adjusting the mass of the UAV. The vehicle was observed to sit stably at the surface with the fins and rear thrusters submerged.

4.2 Remote Control Validation

Before activating autonomous modes, a teleoperator interface was used to manually drive the vehicle and verify that PWM commands were correctly transmitted from the Raspberry Pi to the ESP32 and onward to the ESCs. Thrust symmetry between the two motors was checked, and the heading hold function was tested with manual setpoints.

4.3 Mission 1 — Straight-Line Underwater Propulsion

The first mission validates basic propulsion and waterproofing. Both thrusters are commanded at equal PWM values above 1500 μs, driving the vehicle forward in a straight line. The heading controller maintains the initial heading using the complementary filter estimate. This mission was **successfully completed**: the vehicle moved forward, remained watertight, and maintained its heading using differential thrust.

4.4 Mission 2 — Heading Surface Following

The second mission extends Mission 1 to heading-guided navigation. A heading is loaded as ROS2 parameter. The control node applies differential thrust until the vehicle heading match the desired heading . This mission was also **successfully completed**: the vehicle was able to follow the desired heading.

4.5 Mission 3 — GPS Waypoint Following with Underwater Transit

The second mission extends Mission 1 to GPS-guided navigation. A list of GPS coordinates is loaded as ROS2 parameters. The control node converts each waypoint to the local NE frame, computes the desired heading, and applies differential thrust until the vehicle comes within 5 m of the target before moving to the next waypoint.

Mission 3 uses the GPS waypoint following logic : list of GPS coordinates is loaded as ROS2 then the control node converts each waypoint to the local NE frame, computes the desired heading, and applies differential thrust until the vehicle comes within 5 m of the target before moving to the next waypoint. But this mission comes with the addition of depth control during transit. When the vehicle accelerates toward the target, the inclined fins cause it to dive. Depth is regulated by the speed command so the depth controller is activated during transit and deactivated after a given time. Due to the coupling between speed and depth, tuning the gains proved challenging, and **this mission was not fully validated** in pool conditions.

4.6 Mission 4 — Underwater Back-and-Forth Between Two GPS Points

The fourth mission consists of an automated round-trip between two GPS waypoints. The vehicle dives underwater between the two points and surfaces near each target before diving again for the return leg. This mission builds on Mission 3 by cycling through the waypoint list repeatedly. **This mission was not validated** due to the incomplete resolution of the depth control coupling identified in Mission 3.

5 Discussion

Missions 1 and 2 demonstrated that the core hardware and software stack is functional: sensor data flows correctly through the ROS2 pipeline, the complementary filter provides a usable heading estimate, and the proportional heading controller successfully drives the vehicle to a GPS target.

The main limitation encountered is the coupling between forward speed and depth when using inclined fins as the sole diving mechanism. While this passive approach simplifies the mechanical design, it makes independent depth control difficult: increasing thrust to correct a heading error simultaneously changes the dive angle. Decoupling these two degrees of freedom would require either active pitch actuators (e.g., rotating fins) or a dedicated vertical thruster.

The ArUco pipeline module represents promising directions for future work, particularly for positioning in GPS-denied underwater environments. The Unity simulation proved useful for early algorithm validation but requires further refinement of the hydrodynamic model to match real vehicle dynamics.

6 Conclusion

This project resulted in a functional low-cost torpedo AUV integrating onboard navigation sensors, a ROS2 control stack, and a Unity simulation environment. Two of the four planned missions were successfully validated: straight-line underwater propulsion and GPS surface waypoint following.

Future work should focus on:

- Decoupling depth and speed control, either through active fin actuation or a dedicated vertical thruster;
- Pool validation of the ArUco detection pipeline;
- Integration of optical flow for GPS-denied velocity estimation;
- Improved hydrodynamic modelling in the Unity simulation.

References

[1] Engin Can Dursun, Serhat Yilmaz, and Selim Gökcan Karatop. 2025. Depth Control Based on PID Using a Pressure Sensor in an Autonomous Underwater Vehicle. In *2025 16th International Conference on Electrical and Electronics Engineering (ELECO)*. 1–4. doi:10.1109/ELECO69582.2025.11329270

[2] Thor Fossen. 2021. *Handbook of Marine Craft Hydrodynamics and Motion Control*. doi:10.1002/9781119994138

[3] R. E. Kalman. 1960. A New Approach to Linear Filtering and Prediction Problems. *Journal of Basic Engineering* 82, 1 (March 1960), 35–45. doi:10.1115/1.3662552

[4] Pushkal Katara, Mukul Khanna, Harshit Nagar, and Annapurani Panaiyappan. 2019. Open Source Simulator for Unmanned Underwater Vehicles using ROS and Unity3D. In *2019 IEEE Underwater Technology (UT)*. IEEE, 1–7. doi:10.1109/ut.2019.8734309

[5] Binary Lunar. 2022. Learn to Make Floating Objects in Unity Games Easily - Water Buoyancy Tutorial. https://www.youtube.com/watch?v=vzqoLJmpUqU. YouTube video.

[6] Dong Ma, Ye Li, Teng Ma, and António M. Pascoal. 2025. The state of the art in key technologies for autonomous underwater vehicles: a review. *Engineering* (Aug. 2025). doi:10.1016/j.eng.2025.08.002

[7] Petter Risholm, Peter Ørnulf Ivarsen, Karl Henrik Haugholt, and Ahmed Mohammed. 2021. Underwater marker-based pose-estimation with associated uncertainty. In *2021 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*. 3706–3714. doi:10.1109/ICCVW54120.2021.00414

[8] Jonatan Scharff Willners, Ignacio Carlucho, Tomasz Łuczyński, Sean Katagiri, Chandler Lemoine, Joshua Roe, Dylan Stephens, Shida Xu, Yaniel Carreno, Èric Pairet, Corina Barbalata, Yvan Petillot, and Sen Wang. 2021. From market-ready ROVs to low-cost AUVs. arXiv:2108.05792 [cs.RO] https://arxiv.org/abs/2108.05792

[9] Zhizun Xu, Maryam Haroutunian, Alan J. Murphy, Jeff Neasham, and Rose Norman. 2021. An Underwater Visual Navigation Method Based on Multiple ArUco Markers. *Journal of Marine Science and Engineering* 9, 12 (Dec. 2021), 1432. doi:10.3390/jmse9121432

[10] Mingxi Zhou, Emir Cem Gezer, William McConnell, and Chengzhi Yuan. 2022. Acrobatic Low-cost Portable Hybrid AUV (ALPHA): System Design and Preliminary Results. In *OCEANS 2022, Hampton Roads*. 1–5. doi:10.1109/OCEANS47191.2022.9977113

[11] Jan Čejka, Fabio Bruno, Dimitrios Skarlatos, and Fotis Liarokapis. 2019. Detecting Square Markers in Underwater Environments. *Remote Sensing* 11, 4 (Feb. 2019), 459. doi:10.3390/rs11040459

Subsurface

Conception d'un AUV (Autonomous Underwater Vehicle)

EWAN CHAPUZET
KHALED CHATAH
MAELYS CORMONT
FADI HELOU
BRUNELLE JOSSE
MATHIS LE SCOEZEC
SAMUEL MAGNI

ENSTA
Année 2025–2026

Table des matières

Abstract

Nous présentons la conception et l'intégration de *Subsurface*, un AUV *low-cost* au format torpille, développé dans le cadre d'un projet de cours dont les missions s'inspirent du concours SAUC-E. Le système repose sur une architecture distribuée (Raspberry Pi 4B + ESP32), deux propulseurs, une caméra et une instrumentation minimale (IMU, baromètre, GNSS en surface) avec une pile ROS 2 sur Raspberry Pi. Après une phase centrée sur l'étanchéité et l'équilibrage (centre de masse / centre de flottabilité), nous proposons un modèle dynamique générique et une commande différentielle simple pour le suivi de cap, validée en simulation puis en piscine (mode manuel RC et boucle de cap). Les limites concernent principalement la maîtrise de la profondeur avec des ailes fixes, la portée de la liaison radio sous l'eau et l'absence d'identification fine des paramètres hydrodynamiques.

1 Introduction

1.1 Contexte et cadre du projet

Le concours SAUC-E propose des missions d'inspiration *low-cost* pour des robots sous-marins autonomes. Dans le cadre de deux cours de troisième année à l'ENSTA, chaque groupe disposait d'une base matérielle commune et devait concevoir un véhicule capable d'exécuter un ensemble de missions, en adaptant la mécanique, l'électronique et la couche logicielle.

Notre prototype, nommé *Subsurface*, reprend l'idée d'un véhicule au format torpille, compact et robuste. Le projet nous a confronté à des contraintes de conception sous-marine (étanchéité, gestion des interfaces traversantes, répartition des masses, stabilité hydrostatique) et à l'intégration d'une architecture embarquée permettant à la fois un mode manuel (radio-commande) et des briques d'autonomie (ROS 2, perception caméra).

1.2 Missions du système

Les missions visées (inspirées du cadre SAUC-E) sont les suivantes :

— Suivi de cap en surface
— Réalisation d'un aller-retour sous l'eau sans retour en surface, en passant sous une corde
— Réalisation d'un cycle stable
— Mission d'originalité

2 État de l'art et bibliographie

La conception d'AUV au format torpille à propulsion axiale et gouvernes fixes a fait l'objet de plusieurs études (par ex. Delphin2) [5, 4]. Des architectures de commande PID à double boucle tangage–profondeur sont également documentées [6, 3]. Wang et al. [7] proposent une approche proche de la nôtre : un AUV à flottabilité positive plongeant sans ballast variable, par portance hydrodynamique (ailes) et poussée. Enfin, ce travail s'appuie sur les travaux préliminaires de Brateau [1] (plans mécaniques et premiers essais).

3 Problématique et contributions

La problématique de notre travail est donc la suivante : **Comment concevoir et automatiser un AUV *low-cost* capable d'exécuter des missions sous-marines, tout en respectant des contraintes fortes d'étanchéité, d'équilibrage et de simplicité d'actionnement ?**

Nos contributions principales sont :

— une intégration matérielle complète (mécanique et électronique) au format torpille sur un tube de 80 cm ;
— une architecture logicielle distribuée compatible ROS 2, permettant commande manuelle (RC) et exécution de briques d'autonomie ;
— une méthodologie d'équilibrage (centre de masse et centre de flottabilité) exploitable et réglable rapidement ;
— une brique de perception embarquée (détection couleur et QR code) utilisable comme déclencheur de comportements.

4 Résumé des travaux réalisés

Les travaux ont été organisés autour de trois pôles (mécanique, électronique et logiciel/simulation) afin d'aboutir à un prototype fonctionnel testable en bassin.

— *Mécanique* : étanchéité, protection frontale, ailes fixes et anneaux de lestage ; mise au point d'une procédure d'équilibrage (centre de masse / centre de flottabilité).
— *Électronique/logiciel* : architecture ESP32 (bas niveau) + Raspberry Pi (ROS 2), mode manuel RC et briques d'autonomie ; simulation Unity pour itérer rapidement.
— *Commande/perception* : suivi de cap validé (simulation + piscine) ; profondeur/waypoints/vision développés mais partiellement évalués faute de temps.

5 Hypothèses, définitions et choix de conception

5.1 Hypothèses

Le véhicule est modélisé comme un solide rigide évoluant à faible vitesse en bassin, dans un fluide incompressible et au voisinage de la surface. Les effets de vagues et de courant sont négligés lors des essais en piscine. Les paramètres hydrodynamiques sont considérés inconnus a priori et regroupés dans des coefficients à identifier ultérieurement.

5.2 Forces considérées et conventions

On considère les forces suivantes : poussée d'Archimède (Farch), poids (P), poussées des moteurs (FpropL, FpropR) et traînée globale (Fdrag).

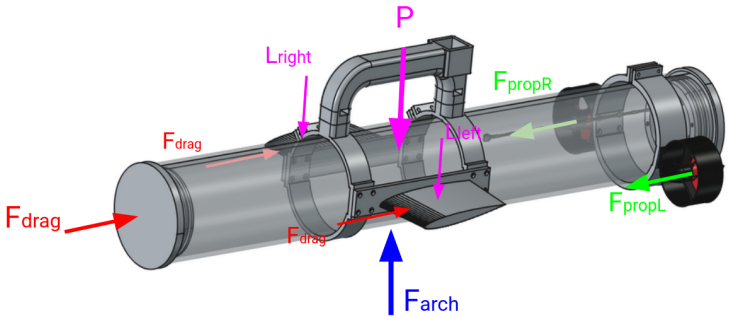


FIGURE 1 – Schéma des forces s'appliquant sur le véhicule Subsurface.

5.3 Structure mécanique et étanchéité

Le véhicule est construit autour d'un tube en plexiglas de longueur 80 cm. Deux bouchons imprimés en résine ferment le tube et assurent l'étanchéité via des joints toriques. Le bouchon avant intègre un hublot (dôme) afin d'obtenir une image propre pour la caméra placée à l'avant. Pour la protection aux chocs, un amortisseur frontal en TPU est ajouté et maintenu par un support imprimé en PLA.

Le bouchon arrière regroupe les interfaces traversantes étanches (pénétrateurs / passe-cloisons) pour les câbles moteurs et certains capteurs, ainsi qu'un interrupteur et un accès de dépressurisation. Sur le dessus du véhicule, une poignée intègre un logement pour un GPS (utilisé en surface) ; le module a été encapsulé dans de la résine afin de garantir l'étanchéité malgré sa position externe.

La stabilité en plongée est obtenue grâce à deux ailes fixes imprimées en résine et montées avec un angle d'incidence constant orienté vers le bas, permettant une plongée naturelle à vitesse suffisante sans actionneur supplémentaire. Deux anneaux externes imprimés en PLA, facilement amovibles, servent d'une part au réglage fin du lest (masses ajoutables et coulissement longitudinal) et d'autre part à l'ajout rapide d'une masse importante lors d'un changement de milieu (eau douce / eau salée).

À l'intérieur du tube, l'électronique est supportée par une plaque en PVC usinée. Un système d'anneaux internes en PLA guide et bloque la plaque afin d'éviter tout glissement ; le démontage reste possible par desserrage d'un écrou.

5.4 Équilibrage : centre de masse et centre de flottabilité

Deux équilibres sont critiques : (i) l'équilibrage des masses, afin d'obtenir une assiette neutre et une dynamique de rotation maîtrisée ; (ii) l'équilibrage hydrostatique, afin d'assurer une remontée stable et sans rotation parasite. Nous appelons *centre de flottabilité* (ou *centre de carène*) le point d'application de la poussée d'Archimède.

Le réglage grossier du centre de masse est réalisé au moyen de bacs imprimés en PLA, remplis de plomb et déplacés sous la plaque interne. Une mousse est ajoutée afin d'éliminer tout déplacement parasite des masses. Un réglage fin est obtenu via un anneau externe coulissant sur le tube, permettant l'ajout de petites masses et un léger déplacement longitudinal du centre de masse.

La compensation d'écart de densité (eau douce vs eau salée) est rendue simple par un second anneau permettant l'ajout rapide d'une masse d'environ 500 g, sans ouverture du tube.

Le centre de flottabilité est ajusté en déplaçant longitudinalement les ailes fixes. La protection frontale (anneau amortisseur) induisant une asymétrie, les ailes doivent être légèrement avancées vers l'avant pour retrouver une stabilité de remontée.

5.5 Architecture électronique

L'architecture électronique du système, dont le schéma complet est présenté en Figure 2, repose sur une alimentation unique par batterie. La distribution de puissance est assurée par un convertisseur DC/DC Traco Power fournissant une tension régulée de 5 V à la Raspberry Pi 4B.

Deux unités de calcul assurent la gestion du véhicule selon une répartition hiérarchique :

— un microcontrôleur ESP32-WROOM-32U, dédié à la commande bas niveau (lecture capteurs, génération PWM, communication série) ;
— une Raspberry Pi 4B, responsable de la couche logicielle ROS 2, de la planification et de l'exécution des missions.

La communication entre ces deux unités est assurée par liaison USB en protocole série.

L'ESP32 pilote deux propulseurs brushless via des contrôleurs ESC dédiés, en générant des signaux PWM. Le système embarque une centrale inertielle (ICM-20948 en SPI), un baromètre (I2C) pour la profondeur, un GNSS utilisé en surface, et une caméra connectée à la Raspberry Pi via l'interface CSI. Les données capteurs sont centralisées sur la Raspberry Pi, qui les intègre dans la boucle de contrôle et la planification des missions.

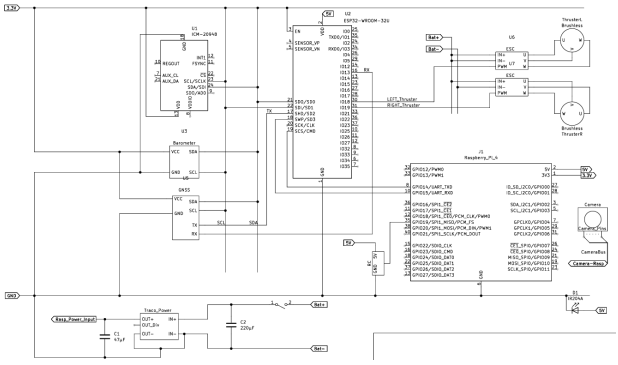


Figure 2 – Schéma électronique du véhicule Subsurface.

5.6 Architecture logicielle et modes de fonctionnement

La logique de commande est distribuée : l'ESP32 assure la boucle bas niveau (lecture capteurs, génération PWM vers ESC, sérialisation), tandis que la Raspberry Pi exécute la pile ROS 2 (en SSH) et calcule les commandes moteurs à partir des mesures reçues. Un récepteur RC (Crossfire) autorise une commande manuelle ; lors des essais en piscine, une commande à environ 1.5 m de profondeur a été observée comme réalisable lorsque l'opérateur reste proche.

Deux modes sont utilisés :

— **Mode manuel** : la Raspberry Pi lit les ordres RC et les relaie à l'ESP32 ;
— **Mode autonome** : un nœud ROS 2 produit des consignes (cap, profondeur, waypoint) et pilote les moteurs via l'ESP32.

6 Modélisation mathématique

6.1 Modèle général 6-DDL

On note $\eta \in \mathbb{R}^6$ la pose (position + orientation) dans le repère inertiel et $\nu \in \mathbb{R}^6$ la vitesse (linéaire + angulaire) dans le repère corps. La cinématique est :

$$\dot{\eta} = J(\eta)\,\nu.$$

La dynamique est écrite sous une forme générique :

$$M\,\dot{\nu} + C(\nu)\,\nu + D(\nu)\,\nu + g(\eta) = \tau,$$

où M regroupe inertie rigide et masse ajoutée, $C(\nu)$ les termes de Coriolis/centripètes, $D(\nu)$ l'amortissement hydrodynamique et $g(\eta)$ les effets poids / flottabilité.

Cette formulation est inspirée des modèles classiques en robotique sous-marine [2] et conserve les coefficients hydrodynamiques sous forme paramétrique (identification ultérieure).

6.2 Actionnement (deux propulseurs)

Le véhicule est équipé de deux propulseurs axiaux. En première approximation, la poussée totale T et le moment de lacet N sont obtenus par combinaison différentielle :

$$T = T_L + T_R, \qquad N = \ell\,(T_L - T_R),$$

où ℓ est un bras de levier effectif. Cette structure motive une loi de commande simple du cap par action différentielle.

7 Théorie et commande

7.1 Suivi de cap par commande différentielle

Nous détaillons ici un correcteur PID sur le cap (la profondeur suit un principe similaire mais agit de manière indirecte sur ce véhicule à ailes fixes).

Soit $\psi(t)$ le cap réel de l'AUV et $\psi_d(t)$ le cap désiré. L'erreur de cap est définie par :

$$e_\psi(t) = \psi_d(t) - \psi(t)$$

La loi de commande PID est donnée par :

$$u_\psi(t) = K_p\,e_\psi(t) + K_i \int_0^t e_\psi(\tau)\,d\tau + K_d \frac{de_\psi(t)}{dt}$$

La commande différentielle sur les propulseurs s'écrit :

$$T_L = T_0 + u_\psi \qquad T_R = T_0 - u_\psi$$

7.2 Régulation de profondeur (développée, non validée)

Une stratégie de régulation de profondeur a été développée sur le même principe PID, à partir de la profondeur estimée par baromètre. Dans notre architecture à ailes fixes et sans gouvernes mobiles, la profondeur dépend fortement de la vitesse et de l'assiette ; la commande agit donc de manière indirecte (principalement via la poussée moyenne T_0). Cette partie n'a pas pu être validée expérimentalement avant la fin du projet.

7.3 Navigation vers waypoint (développée, non validée)

Une brique de navigation a été implémentée pour rejoindre un waypoint : en surface, le GNSS fournit la position, et une consigne de cap est générée vers la cible. La boucle de suivi de cap produit ensuite la commande

différentielle sur les propulseurs. Les essais en bassin n'ont pas permis de valider cette fonctionnalité en conditions représentatives.

7.4 Perception embarquée (caméra)

Une brique de perception (Raspberry Pi + caméra) a été implémentée pour servir de déclencheur de comportements en mission :

— **Détection de feuilles colorées** (rouge/vert/bleu) : segmentation par couleur en espace HSV, puis estimation de la présence de la zone ;
— **Détection QR code** : décodage et estimation d'une orientation relative grossière, utile pour des scénarios d'alignement.

Le résultat est publié sur un topic ROS 2 sous forme d'un identifiant (classe détectée) et, le cas échéant, d'un indicateur d'orientation. Par exemple, une feuille rouge peut déclencher l'arrêt des moteurs.

8 Simulation et expérimentation

8.1 Simulation

Une simulation a été développée sous Unity afin de tester rapidement les briques logicielles (suivi de cap, enchaînement de comportements) avant essais réels. Le modèle y est volontairement simplifié : il reproduit les effets dominants (poussée, traînée, flottabilité) et permet d'observer qualitativement l'impact des gains PID et des seuils de détection vision.

8.2 Dispositif expérimental

Les expérimentations ont été réalisées en piscine (eau douce), dans un environnement contrôlé. Avant la mise à l'eau, un test d'étanchéité a été réalisé. Les paramètres pertinents sont : alimentation batterie LiPo 3S (3000 mAh), deux ESC, Raspberry Pi 4B + ESP32, baromètre de profondeur et IMU.

8.3 Protocoles de tests

Trois tests ont été conduits :

— **Flottabilité statique** : vérification d'une remontée lente et contrôlée ;
— **Assiette en remontée** : le véhicule est immergé puis relâché ; la remontée doit rester parallèle à la surface pour valider l'équilibrage ;
— **Contrôlabilité en translation** : commande manuelle sur quelques mètres ; observation de la stabilité en profondeur et de l'assiette.

8.4 Résultats et discussion

Les essais en piscine montrent que, sur une longueur de bassin de 3 m à 4 m, le véhicule peut maintenir une assiette stable et une profondeur quasi constante à vitesse élevée lorsque l'équilibrage est correctement réglé. La commande manuelle via RC est fonctionnelle sous environ 1.5 m d'eau (opérateur proche). La boucle de suivi de cap a pu être validée en conditions réelles.

Les limites observées et/ou anticipées sont : (i) une sensibilité forte de la profondeur à la vitesse en présence d'ailes fixes, rendant la régulation délicate sans actionneur de profondeur ; (ii) une dépendance aux conditions de liaison radio ; (iii) l'absence d'identification précise des coefficients hydrodynamiques.

9 Conclusion

Nous avons conçu et intégré un AUV *low-cost* au format torpille, en portant l'effort sur l'étanchéité, la robustesse mécanique et l'équilibrage, ainsi que sur une architecture logicielle distribuée compatible ROS 2. Un modèle mathématique général a été posé et une loi de commande différentielle a permis de valider un suivi de cap. Les essais en piscine confirment la contrôlabilité du prototype en mode manuel et la stabilité globale en translation lorsque l'équilibrage est correctement réglé.

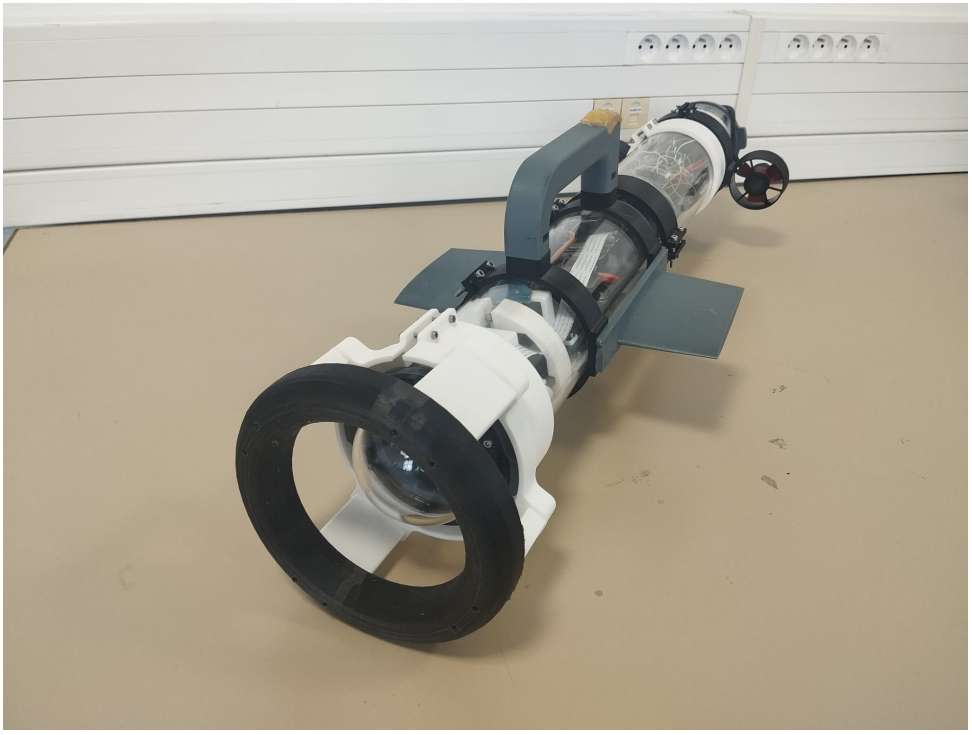


FIGURE 3 – Prototype Subsurface lors des essais (piscine).

Les perspectives d'amélioration les plus directes concernent : (i) l'ajout d'un actionnement dédié à la profondeur (gouvernes mobiles ou ballast) pour réduire la dépendance à la vitesse ; (ii) l'identification des paramètres hydrodynamiques (traînée, portance) afin d'améliorer le dimensionnement et la commande ; (iii) une campagne d'essais plus longue avec acquisition systématique (cap, profondeur, commandes, vidéo) pour quantifier les performances.

Bibliographie

[1] Quentin Brateau. *Conception mécanique et expérimentation d'un AUV low-cost*. Travaux préliminaires, ENSTA.

[2] Thor I. Fossen. *Handbook of Marine Craft Hydrodynamics and Motion Control*. Wiley, 2011.

[3] M. Khalil et al. « Depth control of an autonomous underwater vehicle, STARFISH ». In : *Proceedings of the International Conference on Robotics, Automation and Mechatronics*. 2004.

[4] Alexander B. Phillips et al. « Delphin2 : An over actuated autonomous underwater vehicle for manoeuvring research ». In : *IET Science, Measurement & Technology* (2013).

[5] Leo V. Steenson et al. « Control of an AUV from Thruster Actuated Hover to Control Surface Actuated Flight ». In : *Proceedings of the IFAC World Congress*. 2011.

[6] Leo V. Steenson et al. « Steering and Diving Control of a Small-Sized AUV ». In : *Proceedings of the IFAC Workshop on Navigation, Guidance and Control of Underwater Vehicles*. 2012.

[7] Zheng Wang et al. « Design, Implementation, and Characterization of a Novel Positive Buoyancy Autonomous Vehicle ». In : *Journal of Intelligent & Robotic Systems* (2022).

Titre de l'article : Téléopération et contrôle d'un ROV sous-marin sous-surface

Kilian Barantal, Yasmine Barkoudah-Raoux, Toméo Bourin, Clément Dunot, Rémi Genoux-Lubain

***Résumé*— Les véhicules sous-surfaces permettent la navigation à l'abri des effets marins qui pourraient perturber leur navigation. De plus ils ont l'avantage d'être discret car non-visibles de la surface. Cet article présente un modèle expérimental et une architecture de téléopération pour un ROV sous-surface, combinant estimation d'état, guidage assisté et réalisation de missions de démonstration. Les performances sont évaluées sur des scénarios de navigation et d'immersion, en analysant la précision de suivi, la réalisation des objectifs et la robustesse du système.**

***Index Terms*— robotique sous-marine, positionnemnent, deadreckoning, estimation d'état, contrôle, navigation.**

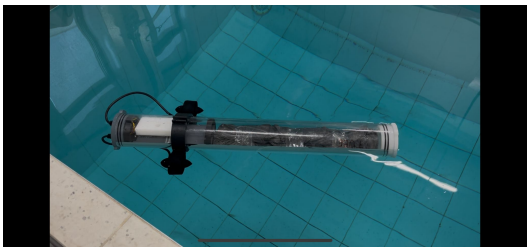


FIG. 1 – Illustration du ROV dans son bassin d'expérimentation.

I. INTRODUCTION

La mise en place de robots sous-marins pouvant naviguer en conditions de mer difficiles et de manière fiable présentent un défi technologique pour de nombreux secteurs tels que l'industrie offshore, la recherche océanographique, les opérations de sécurité et de défense. Dans ce contexte, les véhicules sous-marins jouent un rôle crucial. En particulier, les véhicules opérant sous la surface, à l'abri des perturbations liées à l'état de la mer (houle, vent, trafic de surface), offrent un avantage déterminant : ils bénéficient d'un environnement plus stable pour la navigation tout en assurant une discrétion opérationnelle, étant difficilement détectables depuis la surface.

Cependant, la navigation sous-marine reste une problématique complexe. Elle est soumise à des contraintes sévères qui limitent l'utilisation des technologies de positionnement conventionnelles. La propagation des ondes électromagnétiques, dont le GPS, est en effet impossible dans l'eau, privant les véhicules d'une référence de position absolue en temps réel. De plus, la visibilité souvent réduite complique la perception de l'environnement, et les capteurs embarqués (centrale inertielle, magnétomètre, pression) sont sujets à des dérives et des incertitudes.

Face à ces défis, l'état de l'art propose diverses stratégies, allant du dead-reckoning (navigation à l'estime) basé sur l'intégration de données inertielles, jusqu'à l'utilisation de balises acoustiques (LBL, SBL, USBL) pour un recalage absolu, en passant par des méthodes de fusion de données. Néanmoins, la mise en œuvre de systèmes complets et robustes, alliant estimation d'état fiable et interfaces de téléopération intuitives pour des missions longues, reste un sujet de recherche actif.

Cet article s'inscrit dans cette problématique et présente une contribution au développement d'un système complet pour un robot sous-marin évoluant sous la surface. Nous détaillons la conception d'une architecture robotique à actionnement réduit pour garantir furtivité et robustesse. Ce système cocntient un module d'estimation d'état par fusion de capteurs (centrale inertielle, magnétomètre, sonde de pression) et un mode de guidage assisté. L'objectif est d'évaluer les performances de cette architecture à travers des missions de démonstration en environnement réel. L'analyse porte sur trois aspects clés : la précision du suivi de trajectoire, la capacité à atteindre et maintenir des objectifs de navigation (temps d'immersion, maintien de profondeur), et la robustesse globale du système face aux perturbations et aux limitations capteurs.

Les résultats obtenus démontrent la pertinence de l'approche proposée pour des missions types, tout en mettant en lumière les points de vigilance pour des opérations nécessitant une plus grande précision.

Décrire le contexte (inspection, offshore, recherche, sécurité), les contraintes (courants, visibilité, capteurs), et la contribution de l'article. Situer l'état de l'art et préciser les apports. Exemple de citations : [1], [2].

II. CONTEXTE ET HYPOTHÈSES

Cette section présente le cadre expérimental de l'étude, incluant une description détaillée de la plateforme robotique utilisée, les hypothèses simplificatrices retenues pour la modélisation, ainsi que les notations employées dans la suite de l'article.

A. Plateforme ROV

Le système expérimental est un véhicule sous-marin de petite taille, conçu des missions de navigations furtives. Sa conception privilégie la simplicité mécanique et l'intégration de capteurs à bas coût pour évaluer des architectures de navigation embarquées.

1) Architecture mécanique et propulsion: Le châssis du véhicule est de forme tubulaire, assurant une hydrodynamique simplifiée pour la modélisation. Il est propulsé par deux moteurs DC sans balais (BLDC) montés à l'arrière, configurés en bi-propulsion. Cette configuration permet le contrôle de la

vitesse d'avance par la poussée combinée des deux hélices, et le contrôle du lacet (rotation) par la différence de poussée entre les deux moteurs. Les principales caractéristiques physiques du véhicule sont les suivantes :

— **Masse totale : 8-9 kg**
— **Diamètre du tube : 0.1 m**
— **Longueur hors-tout : 0.80 m**
— **Profondeur opérationnelle maximale : 15 m**
— **Alimentation :** Batterie Lithium-Polymer (LiPo) 3S

2) Architecture électronique et capteurs embarqués: L'ensemble des traitements embarqués est géré par un calculateur central de type Raspberry Pi3 ET UN ESP32. Il est interfacé avec les actionneurs via deux contrôleurs de vitesse électroniques (ESC) et assure l'acquisition des données capteurs. L'alimentation électrique est régulée par un convertisseur DC-DC de type Traco Power pour fournir les tensions nécessaires aux différents composants électroniques.

La suite capteurs, choisie pour estimer l'état du véhicule, est composée de :

— **Centrale Inertielle (IMU) :** Sparkfun, fournissant les accélérations linéaires (3 axes) et les vitesses angulaires (3 axes). Les biais et bruits associés sont caractérisés expérimentalement.
— **Capteur de profondeur :** Bar32. La profondeur est déduite de la mesure de pression.
— **Récepteur GPS :** Un module GPS est intégré dans la poignée située sur le dessus du véhicule. Il est utilisé exclusivement en surface, entre les immersions, pour initialiser la position du véhicule et pour .

B. Hypothèses de travail

Pour le développement du modèle et des lois de commande, les hypothèses simplificatrices suivantes sont adoptées :

1) Le mouvement est considéré dans le plan vertical (immersion) et le plan horizontal (cap et avance). Les mouvements de roulis et de tangage sont supposés négligeables de par la stabilité mécanique du châssis tubulaire.
2) Le véhicile évolue dans un environnement sans courant, ou dont l'effet est considéré comme une perturbation non mesurée rejetée par le correcteur.
3) La poussée des propulseurs est considérée comme proportionnelle au carré du régime moteur, la relation étant identifiée en statique.

C. Cadres et notations

III. Modélisation dynamique et hydrodynamique

A. Cinématique

La cinématique du véhicule établit le lien entre les vitesses exprimées dans le repère corps B et la dérivée de la pose dans le repère inertiel monde I. En notant $\boldsymbol{\eta}_1 = [x, y, z]^T$ la position et $\boldsymbol{\eta}_2 = [\phi, \theta, \psi]^T$ l'orientation (angles d'Euler), et $\boldsymbol{\nu}_1 = [u, v, w]^T$ les vitesses linéaires et $\boldsymbol{\nu}_2 = [p, q, r]^T$ les vitesses angulaires dans le repère corps, la transformation s'écrit :

$$\begin{bmatrix}\dot{\boldsymbol{\eta}}_1 & \dot{\boldsymbol{\eta}}_2\end{bmatrix} = \begin{bmatrix}\mathbf{R}I^B(\boldsymbol{\eta}2) & \mathbf{0}3\times 3 & \mathbf{0}3\times 3 & \mathbf{T}(\boldsymbol{\eta}_2)\end{bmatrix} \begin{bmatrix}\boldsymbol{\nu}_1 & \boldsymbol{\nu}_2\end{bmatrix} \tag{1}$$

où $\mathbf{R}_I^B$ est la matrice de rotation du repère corps vers le repère monde, paramétrée par les angles d'Euler, et $\mathbf{T}$ est la matrice de transformation des vitesses angulaires. Conformément aux hypothèses de travail (section B), les mouvements de roulis (ϕ, p) et de tangage (θ, q) sont supposés négligeables ($\phi \approx 0$, $\theta \approx 0$), ce qui simplifie considérablement ces matrices de transformation.

Pour respecter la garantie de ces contraintes lors de l'expérimentation réelles, un équilibrage cinématique du robot a été réalisé au préalable afin de garantir la stabilité du corps sur le plan horizontal.

B. Dynamique

La dynamique du véhicule est dérivée de l'équation standard de la mécanique des corps rigides en milieu fluide [**?**]. Sous les hypothèses de petits angles et de mouvement découplé dans les plans horizontal et vertical, le modèle 6-DOF complet peut être simplifié. L'équation de la dynamique s'écrit sous la forme :

$$\mathbf{M}\dot{\boldsymbol{\nu}} + \mathbf{C}(\boldsymbol{\nu})\boldsymbol{\nu} + \mathbf{D}(\boldsymbol{\nu})\boldsymbol{\nu} + \mathbf{g}(\boldsymbol{\eta}) = \boldsymbol{\tau} \tag{2}$$

avec $\boldsymbol{\nu} = [u, v, w, r]^T$ le vecteur d'état réduit (la dynamique de roulis p et de tangage q étant négligée), et $\boldsymbol{\tau} = [X, Y, Z, N]^T$ le vecteur des forces et moments appliqués.

La particularité de notre plateforme réside dans le terme de flottabilité et de portance. La poussée d'Archimède est incluse dans $\mathbf{g}(\boldsymbol{\eta})$, où la flottabilité nette est légèrement positive pour assurer une remontée passive en l'absence de vitesse. L'effet des ailes fixes est modélisé au travers des matrices de masse ajoutée $\mathbf{M}A$ (incluse dans $\mathbf{M}$) et d'amortissement $\mathbf{D}(\boldsymbol{\nu})$. Plus spécifiquement, la force de portance L générée par les ailes est modélisée par :

$$L = \frac{1}{2}\rho S C_L(\alpha) u|u| \tag{3}$$

où ρ est la masse volumique de l'eau de mer, S la surface portante des ailes, et C_L le coefficient de portance. Étant donné que les ailes sont fixes et montées au centre de masse avec un angle d'attaque α constant et optimisé pour la plongée, ce coefficient est constant ($C_L(\alpha) = CL0$). La portance est donc directement proportionnelle au carré de la vitesse d'avance u. Les coefficients hydrodynamiques utilisés (C_L, coefficients de traînée) sont issus de la littérature pour des profils adaptés à l'eau de mer. Les efforts dans les autres directions (Y, Z, N) sont principalement dus aux propulseurs et à la traînée du corps.

De même que pour l'équilibre cinématique, un équilibrage dynamique est nécessaire pour éviter les boucles de rétroaction positives qui pourraient rendre le système profondément instable.

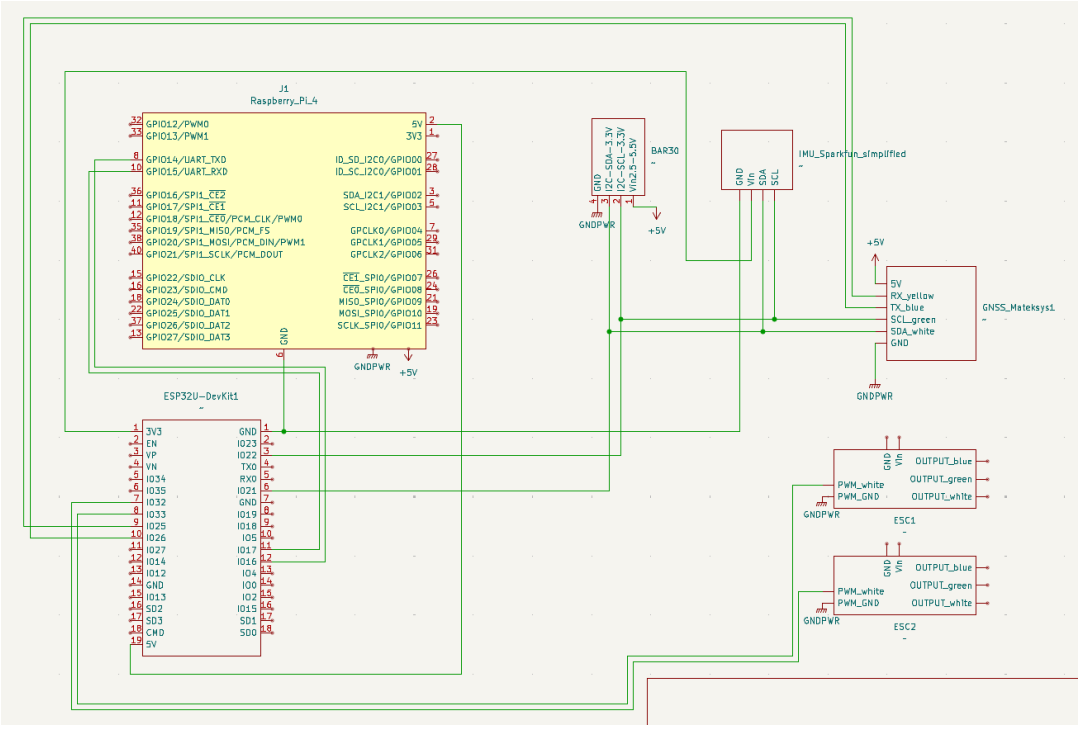


FIG. 2 – Architecture électronique et de contrôle embarquée.

IV. COMMANDE ET SUPERVISION

A. *Architecture de contrôle*

L'architecture de contrôle est hiérarchisée pour assurer une gestion efficace des différentes tâches. La figure 2 illustre cette architecture.

Contrôle de profondeur original : La profondeur est régulée via une boucle unique agissant sur la consigne de vitesse. Un correcteur PID (Proportionnel-Intégral-Dérivé) calcule la vitesse u_{ref} nécessaire pour atteindre et maintenir la profondeur désirée z_{ref} :

$$u_{ref}(t) = K_p e_z(t) + K_i \int_0^t e_z(\tau)d\tau + K_d \frac{de_z(t)}{dt} \qquad (4)$$

avec $e_z = z_{ref} - z$. Ainsi, une erreur de profondeur positive (trop haut) génère une augmentation de la consigne de vitesse, ce qui accroît la portance des ailes et fait plonger le véhicule. Inversement, une réduction de la vitesse diminue la portance, permettant à la flottabilité positive de faire remonter naturellement l'AUV. Cette approche assure un contrôle stable et énergétiquement efficace même s'il limite la vitesse atteinte en surface.

Contrôle de cap et allocation de poussée : Le cap ψ est régulé indépendamment par un second correcteur PID qui détermine un moment de lacet N_{ref} à appliquer. La vitesse d'avance est quant à elle asservie pour suivre la consigne u_{ref} issue du contrôleur de profondeur. La plateforme est équipée de deux propulseurs arrière. L'allocation de poussée consiste donc à convertir les consignes de force de poussée totale T (pour la vitesse) et de moment de lacet N_{ref} en vitesses de rotation ω_d et ω_g pour les propulseurs droit et gauche :

$$T = T_d + T_g \; N \qquad = d \cdot (T_d - T_g) \qquad (5)$$

où d est le bras de levier. La poussée de chaque propulseur est ensuite convertie en régime moteur via la relation quadratique identifiée en statique : $T_i = k_T \omega_i |\omega_i|$. La saturation des actionneurs est gérée en priorisant la commande de vitesse sur celle de lacet en cas de dépassement des limites physiques des moteurs.

B. *Téléopération et modes assistés*

Plusieurs modes de fonctionnement sont voué à être implémentés pour faciliter la téléopération et les missions autonomes :

Mode manuel : L'opérateur commande directement la vitesse de rotation des deux moteurs (différentiel pour le lacet).

Maintien de cap : L'asservissement de cap est activé, l'opérateur ne commande que la vitesse pour établir la vitesse ainsi que la profondeur.

Maintien de profondeur : Le PID de profondeur est actif. L'opérateur commande le cap et la vitesse est asservie pour maintenir la profondeur. Le véhicule peut ainsi "survoler" une isobare.

Suivi de trajectoire (prédéfini) : Mode autonome où le véhicule suit une séquence de waypoints en gérant de manière couplée cap, vitesse et profondeur.

Un retour systématique à un mode neutre est implémenté pour faire remonter le robot en utilisant la flottabilité légèrement positive du système.

TABLE I – Résultats des performances de contrôle pour les missions de suivi de cap et de profondeur.

Métrique	**Mission 1**	**Mission 2**	**Mission 3**
RMSE Profondeur (m)	0.88	0.12	0.15
RMSE Cap (°)	2.5	3.1	4.2
Dépassement max (m)	0.22	0.30	–
Temps de stabilisation (s)	8.5	12.1	–

V. Estimation d'état et fusion de capteurs

L'estimation robuste de l'état du véhicule est cruciale pour la commande. En l'absence de DVL (Doppler Velocity Log), nous avons développé une approche de fusion capteur basée sur un Filtre de Kalman Étendu (EKF). L'état estimé $\hat{\mathbf{x}} = [\hat{x}, \hat{y}, \hat{z}, \hat{\phi}, \hat{\theta}, \hat{\psi}, \hat{u}, \hat{v}, \hat{w}, \hat{b}_a, \hat{b}_g]^T$ comprend la pose, les vitesses et les biais des accéléromètres ($\hat{b}_a$) et gyroscopes ($\hat{b}_g$).

Entrées du filtre :

IMU (Sparkfun) : Fournit les accélérations linéaires ($\mathbf{a}_m$) et vitesses angulaires ($\boldsymbol{\omega}_m$) à haute fréquence. Ces mesures sont utilisées dans l'étape de prédiction de l'EKF via le modèle cinématique.

Capteur de pression (Bar32) : La mesure de pression P est convertie en profondeur z_{mes}, fournissant une mise à jour directe et précise sur la composante verticale de l'état.

Modèle d'évolution et commandes moteur : Les consignes de vitesse moteur sont utilisées dans le modèle de prédiction pour estimer la vitesse d'avance u, en s'appuyant sur le modèle de propulsion.

Modèle de portance : Le modèle de portance des ailes est intégré dans le modèle dynamique utilisé par l'EKF, créant un lien entre la vitesse u estimée et l'évolution de la profondeur z, ce qui améliore la robustesse de l'estimation, même sans mesure directe de vitesse.

GPS : En surface, entre les immersions, la mesure GPS (x_{gps}, y_{gps}) est utilisée pour réinitialiser la position et borner la dérive de l'estimation inertielle.

Sorties du filtre : L'EKF fournit une estimation filtrée et synchronisée de la pose complète (x, y, z, ψ) et des vitesses (u, v, w, r), essentielles pour le bon fonctionnement des boucles de contrôle décrites en section IV.

VI. Expérimentations

A. Protocole

Pour valider l'approche de contrôle de profondeur par les ailes, un protocole expérimental a été défini dans un bassin d'essai (FIG. 1) et réalisé dans un bassin en eau de mer.

Tout d'abord, comme dit précédemment, un équilibrage cinématique et dynamique est nécessaire avan le déploiement de l'UAV.

Trois types de missions ont ensuite été réalisées :

Suivi de cap : Le véhicule, en surface, doit atteindre une cible $z_{cible} = 0, m$ en établissant et suivant un cap à suivre. Une fois la cible atteinte, il s'arrête à proximité.

Aller-retour : L'AUV doit suivre une trajectoire entre deux points, puis faire demi-tour et revenir à son point de départ. L'AUV doit pouvoir recommencer ce cycle à l'infini.

Plongée : Le véhicule doit exécuter une séquence de plongée pour rejoindre sa cible. Une corde flottante est installée et l'UAV doit pouvoir passer en dessous de cette corde pour atteindre cet objectif.

Les métriques de performance évaluées sont l'Erreur Quadratique Moyenne (RMSE) sur la profondeur et le cap, le dépassement maximal (D_{max}), et le temps de stabilisation à 000 de la valeur de consigne.

B. Résultats

Les résultats expérimentaux confirment l'efficacité de la stratégie de contrôle proposée.

IMAGE BASE NAVALE

FIG. 3 – Photos prise pendant les essais dans le bassin de la base navale de Brest

Les faibles RMSE sur la profondeur démontrent la précision du PID couplé au modèle de portance. Le dépassement, bien que modéré, est principalement dû à l'inertie du système et au temps de réponse des propulseurs. Le temps de stabilisation, plus élevé pour la mission 2, s'explique par la complexité de gérer simultanément le cap et la profondeur sur une plus longue distance.

VII. Discussion

Les résultats obtenus valident l'approche de contrôle de profondeur par modulation de vitesse sur un AUV à ailes fixes. Cependant, plusieurs limites et axes d'amélioration ont été identifiés. Tout d'abord, la plage de vitesse contrôlable est bornée : en dessous d'une vitesse minimale u_{min}, la portance devient insuffisante pour vaincre la flottabilité, rendant le contrôle de descente inefficace. À l'inverse, une vitesse trop élevée peut générer une portance excessive et des non-linéarités hydrodynamiques.

Le modèle ne prend pas en compte l'effet des courants, qui constituent une perturbation majeure, en particulier les courants verticaux. La robustesse du correcteur face à ces perturbations non mesurées, bien que partiellement rejetée par l'action intégrale, reste à évaluer en milieu naturel. Comparée

aux approches de référence utilisant des propulseurs verticaux, notre méthode offre un avantage certain en termes de simplicité mécanique et de consommation énergétique pour des missions de longue durée, mais elle est moins adaptée pour des manœuvres verticales rapides ou précises à faible vitesse.

VIII. Conclusion

Nous avons présenté la conception, la modélisation et la validation expérimentale d'un AUV original dont le contrôle de profondeur est assuré par des ailes fixes, la consigne de profondeur étant convertie en une consigne de vitesse d'avance par un correcteur PID. Les résultats en bassin confirment la faisabilité et la précision de cette approche pour des missions de suivi de trajectoire sous-marine. La principale contribution de ce travail réside dans l'intégration simple et efficace d'un modèle de portance dans la boucle de contrôle et le filtre d'estimation, en utilisant une instrumentation minimale.

Les perspectives de ce travail sont multiples :

Autonomie partielle : Intégration d'une planification de trajectoire énergétiquement optimale, exploitant la capacité naturelle du véhicule à planer.

SLAM sous-marin : Couplage de l'estimation d'état actuelle avec un système de vision pour la navigation relative et la cartographie.

Allocation de poussée : Optimisation de la loi d'allocation pour mieux gérer les saturations et le couplage entre vitesse et lacet.

Remerciements

Financement, partenaires, accès aux bassins d'essai, etc.

References

[1] J. E. Colgate, W. Wannasuphoprasit, and M. A. Peshkin, "Cobots : Robots for collaboration with human operators," in *Proceedings of the ASME 1996 International Mechanical Engineering Congress and Exposition*. Atlanta, Georgia, USA : ASME, 1996, 433–439, paper No : IMECE1996-0367.

[2] M. Peshkin and J. E. Colgate, "Cobots," *Industrial Robot : An International Journal*, vol. 26, no. 5, 335–341, 1999.

Robot sub-surface pour la compétition SAUC'E

Ghita Tazeroualt Tlamcani* Maxim Lefèvre* Robin Vidal* Aurèle Planchard*
Thomas Bourgeois* Hussein Rammal* Noé Lemaistre*

6 mars 2026

Résumé

Ce article présente la conception et le développement d'un robot sub-surface autonome destiné à la compétition SAUC'E. Le véhicule est conçu pour naviguer légèrement sous la surface libre grâce à des ailes portantes inclinées contrôlant la profondeur par la vitesse d'avancement, sans système de ballast actif. L'architecture repose sur une électronique hiérarchisée combinant une Raspberry Pi 4 et un microcontrôleur ESP32, ainsi qu'une pile logicielle ROS 2 (Humble) assurant la fusion de données inertielles, la localisation GNSS et la navigation autonome par waypoints. Les essais réalisés à l'École Navale ont validé la stabilité mécanique, l'étanchéité et les algorithmes de navigation, démontrant la viabilité de cette approche pour des missions de surveillance discrète en environnement côtier.

1 Introduction

La robotique marine autonome constitue aujourd'hui un domaine stratégique pour les applications de surveillance, de reconnaissance et de collecte de données en environnement maritime. Dans ce contexte, la discrétion plateformes robotisées représente un enjeu majeur, en particulier pour les missions nécessitant une présence prolongée dans des zones surveillées ou contestées. Les véhicules sous-marins autonomes conventionnels, tels que **Bluefin-21**, sont conçus pour évoluer en immersion complète avec une stabilité hydrodynamique élevée et des capacités de navigation précises [1]. Toutefois, leur profondeur opérationnelle et leur signature énergétique ou acoustique peuvent limiter leur emploi pour des missions nécessitant une discrétion à proximité de la surface.

Une alternative est offerte par les planeurs sous-marins, notamment **Seaglider** et **Liberdade class underwater glider**, qui convertissent des variations de flottabilité en propulsion horizontale grâce à des ailes hydrodynamiques [2], [3]. Cette stratégie de locomotion, fondée sur des cycles de plongée et de remontée, permet d'atteindre des autonomies opérationnelles de plusieurs semaines voire mois [4]. Néanmoins, ces systèmes reposent généralement sur des stratégies de variation de flottabilité (pompes, masses mobiles, réservoirs), ce qui introduit une complexité mécanique et impose des déplacements lents [5].

Des robots hybrides surface-subsurface ont également émergé afin de concilier communication, observation et navigation discrète. Le planeur **SeaExplorer** illustre l'intérêt d'une navigation

*Ensta Bretagne, *emails: {ghita.tazeroualt, maxime.lefevre, robin.vidal, aurele.planchard, thomas.bourgeois, hussein.rammal2, noe.lemaistre}@ensta.fr*

proche de la surface pour la transmission de données et la localisation GNSS tout en conservant les avantages du gliding sous-marin [6]. Toutefois, ces architectures reposent toujours principalement sur la gestion de la flottabilité pour contrôler la profondeur.

Dans ce travail, nous proposons une architecture de robot sub-surface dédiée aux missions de surveillance discrète en milieu marin. Le véhicule est conçu pour évoluer légèrement sous la surface libre, seule une structure émergée minimale supportant le GNSS restant visible. Cette configuration réduit la probabilité de détection par caméra aérienne ou satellitaire, tout en maintenant une immersion insuffisante pour correspondre aux profils typiquement recherchés par les systèmes de détection sous-marins conventionnels.

La contribution principale de cette étude réside dans l'utilisation d'ailes portantes inclinées permettant de contrôler la profondeur uniquement par la vitesse d'avancement. Contrairement aux planeurs sous-marins traditionnels, la transition entre navigation de surface et immersion ne repose pas sur un système de ballast actif mais sur l'équilibre dynamique entre portance hydrodynamique et poids apparent. Cette approche présente plusieurs avantages parmis lesquels on peut citer : (i) une réduction significative de la masse embarquée et de la complexité mécanique, et (ii) une réponse dynamique continue permettant une adaptation instantanée de la profondeur.

Ainsi, le robot proposé se positionne à l'interface entre les véhicules autonomes sous-marins classiques et les planeurs hydrodynamiques. Cette configuration ouvre de nouvelles perspectives pour la conception de robots marin furtivf, légèrs et adaptées aux missions de surveillance discrète en environnement côtier.

2 Architecture mécanique

Afin d'assurer l'assiette et la stabilité du robot, trois types de solutions de lestage ont été développés :

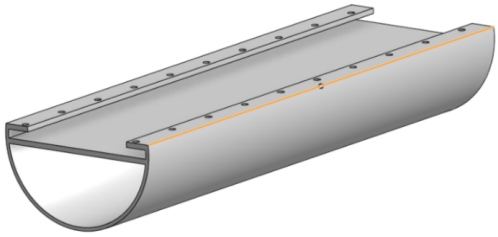
— **Le béton :** utilisé pour le lestage principal.
— **Les billes de plomb :** pour un ajustement précis de la masse.
— **Les masses en aluminium :** dédiées au réglage fin en conditions réelles.

Pour intégrer ces éléments, différents modules ont été conçus en CAO et réalisés par impression 3D, comme illustré sur la Figure 1.

Le dispositif de lestage se décompose ainsi :

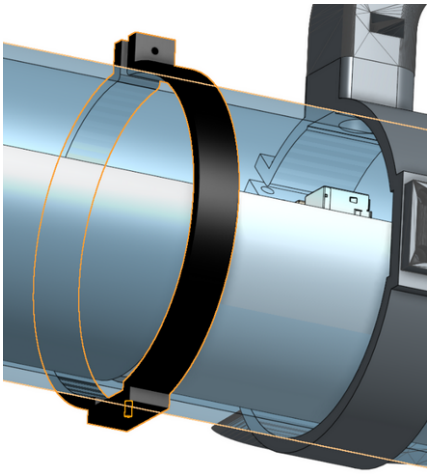
1. **Module central en béton :** Cette pièce a servi de moule lors du coulage et constitue la majeure partie de la masse ajoutée au robot. Elle est positionnée au centre du châssis tubulaire.
2. **Modules de plomb et batteries :** Situé à l'avant, un module spécifique accueille les billes de plomb pour compenser le déport de masse vers l'arrière dû aux batteries. Par souci de modularité et de symétrie, un module identique est utilisé à l'arrière pour fixer l'une des deux batteries.
3. **Bagues de fixation externes :** Ces bagues permettent l'ajout de briques en aluminium. Initialement conçues pour compenser la différence de flottabilité entre l'eau douce et l'eau salée, les tests ont démontré que leur utilisation n'était pas nécessaire.

Au total, le robot affiche une masse de **8 kg**.



(a) Conception du module de lestage en béton

(b) Conception du module pour billes de plomb



(c) Bague de fixation pour masses externes

FIGURE 1 – Modélisations CAO des différents systèmes de lestage

L'organisation interne du robot, illustrée par la Figure 2, repose sur une platine en PVC supportant l'ensemble des composants électroniques. Cette structure est maintenue en position par deux bagues situées à chaque extrémité. Ces bagues sont extensibles (diamètre ajustable) afin d'assurer un maintien en pression optimal contre les parois internes du tube, empêchant ainsi tout mouvement de la structure lors des déplacements du robot.

Au total, le robot affiche une masse de 8 kg.

Pour protéger l'intégrité de la structure lors d'éventuels impacts, une solution d'amortissement a été conçue (voir Figure 3). Ce module se compose de trois éléments distincts :

— Une butée cylindrique à face sphérique imprimée en PLA, destinée à absorber le contact direct.

— Un fourreau cylindrique creux en TPU (matériau flexible) assurant la liaison élastique entre la butée et le tube.

— Une bague de serrage permettant de solidariser l'ensemble au châssis du robot.

L'assemblage final de ces différents modules permet d'aboutir à la configuration complète présentée sur la Figure 4.

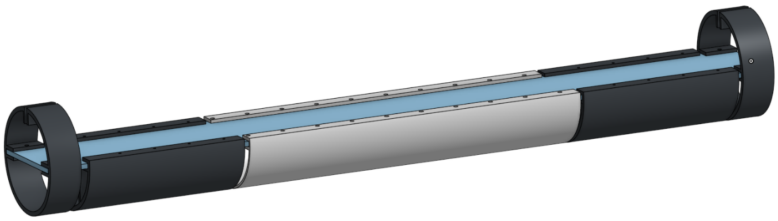


FIGURE 2 – Organisation interne et système de maintien de l'électronique

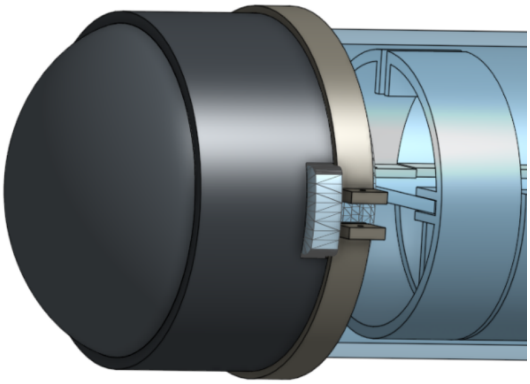


FIGURE 3 – Détail de la solution d'amortissement (PLA et TPU)

3 Architecture électronique

L'architecture électronique du robot est conçue selon une structure hiérarchisée, séparant les tâches de haut niveau (traitement de données, navigation) des tâches de bas niveau (contrôle moteur, lecture de capteurs). Cette séparation est assurée par deux unités de calcul complémentaires.

3.1 Unités de contrôle

Le système repose sur la collaboration entre une **Raspberry Pi 4** et un microcontrôleur **ESP32**.

- **Raspberry Pi 4 (Cerveau central) :** Elle assure la puissance de calcul nécessaire pour faire tourner l'environnement **ROS** (Robot Operating System). Elle permet un accès distant via **SSH** pour la maintenance et le monitoring, et transmet les ordres de mission à l'ESP32.
- **ESP32 (Contrôleur bas niveau) :** Ce microcontrôleur permet de décharger le CPU de la Raspberry Pi des tâches temps réel. Sa compacité, son faible coût et sa facilité d'alimentation en font une interface idéale. Il assure une communication rapide avec les périphériques.

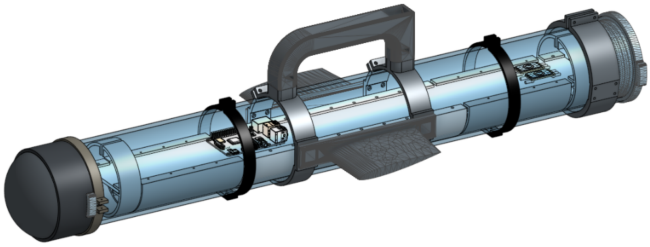


FIGURE 4 – Vue d'ensemble du robot finalisé

La communication entre ces deux unités est réalisée via une liaison série sur **câble USB**, utilisant un protocole d'échange de données au format **ASCII**.

3.2 Gestion des capteurs et actionneurs

L'ensemble des périphériques est centralisé sur l'ESP32. Le robot est équipé des capteurs suivants pour la navigation et le suivi d'état :

— **IMU :** Pour l'orientation et la stabilisation.

— **GPS :** Pour la localisation globale en surface.

— **Baromètre :** Pour la mesure de la profondeur.

Les contrôleurs de vitesse (**ESC**) pilotant les moteurs sont également connectés directement aux sorties PWM de l'ESP32.

3.3 Alimentation et Distribution

L'énergie est fournie par deux batteries **LiPo 3S**, séparées pour isoler la puissance de la commande et éviter les perturbations électromagnétiques :

1. **Batterie de commande :** Alimente la Raspberry Pi 4, qui distribue ensuite l'énergie à l'ESP32 et au reste des capteurs du circuit.
2. **Batterie de puissance :** Dédiée exclusivement à l'alimentation des deux ESCs pour répondre aux appels de courant des moteurs.

3.4 Conception du PCB et câblage

Afin d'optimiser l'organisation interne et de garantir un *cable management* fiable, nous avons conçu un PCB fait main. Ce dernier permet de fixer solidement l'ESP32 et de centraliser les connexions des capteurs et des alimentations, évitant ainsi les déconnexions intempestives dues aux vibrations ou aux manipulations (voir Figure 5).

Gemini a dit

Voici votre section Architecture logicielle corrigée avec l'application stricte des règles LaTeX pour les expressions mathématiques et les symboles techniques.

J'ai également inséré une balise de diagramme pour illustrer visuellement la structure de votre système ROS 2, ce qui est très apprécié dans les rapports techniques.

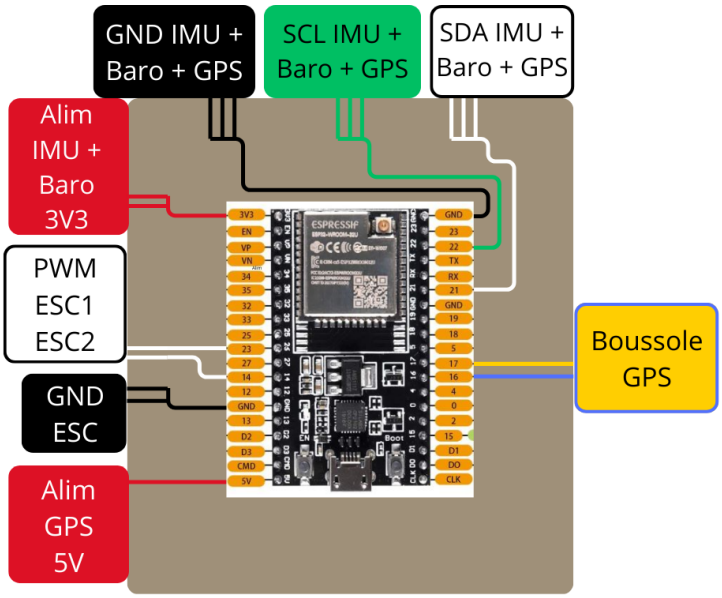


FIGURE 5 – Vue de la carte d'interface (PCB) pour la centralisation des composants

4 Architecture logicielle

L'intelligence du robot repose sur l'écosystème **ROS 2 (Humble)**, permettant une architecture modulaire et distribuée. L'intégralité du code source et des scripts de déploiement est accessible sur le dépôt GitLab du projet : Dépôt Projet - ENSTA Bretagne. L'architecture se décompose en trois couches fonctionnelles : le pont matériel, le traitement de pose et la gestion des missions.

4.1 Couche d'interface matérielle (Bridge)

Le nœud `bridge_node.py` fait office de passerelle entre le monde physique et l'environnement logiciel. Il assure la réception des trames série provenant de l'ESP32 via USB, réalise le *parsing* des données au format ASCII et les publie sur les topics suivants :

- `/groupe5/raw_imu` : Accéléromètre, gyroscope et magnétomètre bruts.
- `/groupe5/pressure` : Données du baromètre pour le suivi de profondeur.
- `/groupe5/gps` : Coordonnées GNSS (WGS84).

4.2 Estimation d'état et filtrage

Le calcul de la position et de l'orientation précise est critique pour la navigation. Deux nœuds principaux gèrent cette transformation :

- **Filtrage d'orientation :** Le nœud `imu_madgwick_node.py` implémente un filtre de Madgwick pour fusionner les données inertielles. Il produit une orientation stable sous forme de quaternion sur le topic `/groupe5/imu`.
- **Changement de repère :** Le nœud `pose_node.py` convertit les données GPS géocentriques en coordonnées cartésiennes locales (**Lambert 93**). Il définit également une position de référence (`pose_ref`) lors de l'initialisation pour permettre une navigation relative précise.

4.3 Couche Navigation et Commandes

Cette couche traduit les objectifs de haut niveau en consignes moteurs (normalisées entre $[-1, 1]$) envoyées à l'ESP32. Plusieurs modes de mission ont été développés :

- **Suivi de cap (*Follow Heading*) :** Asservissement de l'angle lacet (*yaw*) basé sur l'IMU pour maintenir une direction constante.
- **Navigation GPS (*Follow GPS*) :** Ralliement de points de passage (*Waypoints*) avec correction d'orientation dynamique.
- **Mission temporelle (*Corde*) :** Séquenceur de vitesse en trois phases, utilisé pour passer sous la corde.

4.4 Synthèse des échanges de données

Le tableau ci-dessous résume les principaux flux de données circulant sur le bus ROS 2 :

Topic	Type	Description
`/groupe5/pose`	`PoseStamped`	Position et orientation estimées (Lambert 93).
`/groupe5/imu`	`ImuComplete`	Données inertielles filtrées et calibrées.
`/groupe5/command`	`Float32MultiArray`	Consignes finales envoyées aux moteurs $[-1, 1]$.
`/groupe5/radar`	`PointCloud2`	Données sonar (nuage de points obstacles).
`/cmd_vel`	`Twist`	Commande manuelle (clavier ou joystick).

TABLE 1 – Synthèse des principaux topics ROS 2

4.5 Déploiement et Configuration

La flexibilité du système est assurée par l'utilisation de fichiers de configuration **YAML** (situés dans `src/mission/config/`). Ces fichiers permettent d'ajuster les paramètres d'asservissement (k_h, k_v), les coordonnées cibles ou les durées de mission sans recompiler le code. Le lancement des différentes briques logicielles est automatisé par des fichiers *Launch* ROS 2.

5 Résultats et Expérimentations

Les essais en conditions réelles du robot se sont déroulés à l'École Navale. Ces tests ont permis de valider l'architecture matérielle et les algorithmes de navigation développés.

5.1 Validation de la flottabilité et du lestage

Bien que des dispositifs de lestage externe aient été prévus pour compenser la différence de densité entre l'eau douce et l'eau salée, les essais ont démontré que la configuration interne actuelle était suffisante. Le robot a présenté une assiette stable sans nécessiter l'ajout de masses supplémentaires en aluminium.

5.2 Bilan des missions de navigation

Les performances logicielles ont été évaluées à travers trois missions de complexité croissante :

— **Mission 0 (Stabilisation) :** Succès total. Le robot a validé sa stabilité statique ainsi que sa résistance dynamique lors d'une poussée moteur à 100% de puissance pendant 3s. La structure mécanique a parfaitement encaissé les contraintes.

— **Mission 1 (Suivi de cap) :** Succès. Une recalibration de l'IMU a été nécessaire sur site, les paramètres magnétiques de l'École Navale différant de ceux relevés lors des tests préliminaires à l'ENSTA Bretagne. Après cette étape, l'asservissement en cap s'est montré robuste.

— **Mission 2 (Navigation GPS) :** Résultats mitigés. Bien que les *logs* internes du robot indiquent un comportement cohérent des algorithmes de ralliement, la fiabilité n'a pas atteint 100%. L'absence de balises physiques dans l'eau a rendu la vérification visuelle de la précision difficile.

Ces résultats ont été validés par Monsieur Luc Jaulin et Madame Natacha Caouren, présents lors des démonstrations.

5.3 Analyse des limites et points critiques

Les tests de puissance ont permis d'identifier le seuil d'immersion du robot : au-delà de 75% de poussée, le robot commence à plonger sous la surface. Ce comportement, attendu compte tenu de la conception du châssis, définit la limite opérationnelle à partir de laquelle la navigation subaquatique s'engage.

Mission	**Statut**	**Observations**
Stabilisation (M0)	Succès	Structure et étanchéité validées à 100% de puissance.
Suivi de cap (M1)	Succès	Nécessite une calibration locale de l'IMU.
Navigation GPS (M2)	Partiel	Algorithme cohérent mais fiabilité à optimiser.

TABLE 2 – Synthèse des résultats expérimentaux à l'École Navale.

6 Conclusion

Ce travail a permis de concevoir, réaliser et valider un robot sub-surface autonome répondant aux exigences de la compétition SAUC'E. L'ensemble des choix architecturaux, tant mécaniques qu'électroniques et logiciels, ont été guidés par les contraintes de discrétion, de robustesse et de simplicité de déploiement.

Sur le plan mécanique, la stratégie de contrôle de profondeur par ailes portantes s'est révélée efficace, permettant de s'affranchir de tout système de ballast actif. Le lestage interne a suffi à garantir une assiette stable en eau salée, confirmant la pertinence des choix de conception. Sur le plan logiciel, l'architecture ROS 2 modulaire a démontré sa flexibilité, notamment grâce à la reconfiguration rapide des paramètres d'asservissement par fichiers YAML lors des essais sur site. Les trois missions de navigation ont été menées avec succès ou ont fourni des résultats exploitables, validant le pipeline complet allant de la fusion inertielle jusqu'au ralliement de waypoints GPS.

Les principales limites identifiées concernent la précision de la navigation GPS en l'absence de balises de référence physiques dans l'eau, ainsi que le seuil d'immersion à 75% de poussée qui contraint la vitesse opérationnelle maximale en mode surface. Ces points constituent les axes d'amélioration prioritaires pour les itérations futures, notamment par l'intégration d'un filtre de navigation plus robuste et une optimisation hydrodynamique du châssis.

Références

[1] Junku Yuh. "Design and control of autonomous underwater robots : A survey". In : *Autonomous Robots* 8.1 (2000), p. 7-24.

[2] Charles C Eriksen et al. "Seaglider : A long-range autonomous underwater vehicle for oceanographic research". In : *IEEE Journal of oceanic Engineering* 26.4 (2001), p. 424-436.

[3] Naomi Ehrich Leonard et Joshua G Graver. "Model-based feedback control of autonomous underwater gliders". In : *IEEE Journal of oceanic engineering* 26.4 (2002), p. 633-645.

[4] Enrico Petritoli et Fabio Leccese. "Autonomous underwater glider : a comprehensive review". In : *Drones* 9.1 (2024), p. 21.

[5] Joshua Grady Graver. "Underwater gliders : Dynamics, control and design". In : (2005).

[6] ALSEAMAR. *SeaExplorer Underwater Glider : System Overview and Applications*. Technical Documentation. Rousset, France : ALSEAMAR, 2020. URL : `https://www.alseamar-alcen.com/en/seaexplorer/`.