Task-based design and optimization of reconfigurable propulsion systems for autonomous underwater vehicles

Emanuel Pablo Vega

université de bretagne occidentale

UNIVERSITE BRETAGNE LOIRE

THÈSE / UNIVERSITÉ DE BRETAGNE OCCIDENTALE

sous le sceau de l'Université Bretagne Loire

pour obtenir le titre de

DOCTEUR DE L'UNIVERSITÉ DE BRETAGNE OCCIDENTALE

présentée par

Emanuel Pablo VEGA

Préparée au sein de la FRE CNRS 3744 IRDL

Institut de Recherche Dupuy de Lôme

Task-based design and optimization of reconfigurable propulsion systems for autonomous underwater vehicles

Thèse soutenue le 20 octobre 2016
devant le jury composé de :

Philippe BIDAUD
Professeur, Université Pierre et Marie Curie / *rapporteur*

Frédéric BOYER
Professeur, Ecole des Mines de Nantes / *rapporteur*

Frédéric CHAPELLE
Maître de Conférences, SIGMA Clermont / *examinateur*

Vincent HUGEL
Professeur, Université de Toulon / *examinateur*

Luc JAULIN
Professeur, ENSTA Bretagne / *président*

Janito VAQUEIRO FERREIRA
Professor, University of Campinas, Brazil / *examinateur*

Olivier CHOCRON
Maître de Conférence, ENI Brest / *co-encadrant*

Mohamed BENBOUZID
Professeur, Université de Bretagne Occidentale / *directeur de thèse*

Acknowledgments

I wish to thank my supervisors, *Prof. Mohamed Benbouzid* and *Olivier Chocron*, for their assistance, advice and encouragement throughout this work. Working with them gave me the opportunity to gain from their areas of expertise and allowed me to broaden my knowledge, for which I am very grateful.

I would like to thank as well the pre-examiners *Prof. Philippe Bidaud* from the Université Pierre et Marie Curie and *Prof. Frédéric Boyer* from the Ecole des Mines de Nantes for their valuable corrections and comments. I would also like to thank my PhD thesis defense committee members: *Frédéric Chapelle* from SIGMA Clermont, *Prof. Vincent Hugel* from the Université de Toulon, *Prof. Luc Jaulin* from ENSTA Bretagne and *Prof. Janito Vaqueiro Ferreira* from the University of Campinas, Brazil.

I wish to thank all my co-workers from the IRDL-ENIB laboratory, who helped me during the completion of my PhD thesis via their comments, advise and technical expertise. I would also like to thank those I have shared an office with, their company and friendship made the realization of this work a fantastic experience.

At last, but not least, I would like to thank my *non-workplace friends* and family, for their support and for always believing in me.

Abstract

In this manuscript, the optimization of the propulsion and control of AUVs is developed. The hydrodynamic model of the AUVs is examined. Additionally, AUV propulsion topologies are studied and models for fixed and vectorial technology are developed. The fixed technology model is based on an *off the shelf* device, while the modeled vectorial propulsive system is based on a magnetic coupling thruster prototype developed in IRDL[1] (ENI Brest).

A control method using the hydrodynamic model is studied, its adaptation to two AUV topologies is presented and considerations about its applicability will be discussed.

The optimization is used to find suitable propulsive topologies and control parameters in order to execute given robotic tasks, speeding up the convergence and minimizing the energy consumption. This is done using a genetic algorithm, which is a stochastic optimization method used for *task-based* design. The results of the optimization can be used as a preliminary stage in the design process of an AUV, giving ideas for enhanced propulsive configurations.

The optimization technique is also applied to an IRDL existing robot, modifying only some of the propulsive topology parameters in order to readily adapt it to different tasks, making the AUV dynamically reconfigurable.

[1]Institut de Recherche Dupuy de Lôme CNRS FRE 3744, Brest-Lorient, France

Résumé

Dans ce travail, l'optimisation de la propulsion et de la commande des AUV (Autonomous Underwater Vehicles en anglais) est développé. Le modèle hydrodynamique de l'AUV est examiné. Également, son système de propulsion est étudié et des modèles pour des solutions de propulsion différentes (fixe et vectorielle) sont développés dans le cadre de la mobilité autonome.

Le modèle et l'identification de la technologie de propulsion dite fixe sont basés sur un propulseur disponible commercialement. Le système de propulsion vectoriel est basé sur un prototype de propulseur magneto-couplé reconfigurable (PMCR) développé à l'IRDL-ENIB.

Une méthode de commande non linéaire utilisant le modèle hydrodynamique de l' AUV est développée et son adaptation à deux systèmes de propulsion est présentée. Des analyses portant sur la commandabilité du robot et l'application de cette commande à différents systèmes sont proposées.

L'optimisation globale est utilisée pour trouver des topologies propulsives et des paramètres de commande adaptés à la réalisation de tâches robotiques spécifiques. L'optimisation réalisée permet de trouver des solutions capables d'assurer le suivi de trajectoire et de minimiser la consommation génergétique du robot. L'optimisation utilise un algorithme génétique (algorithme évolutionnaire), une méthode d'optimisation stochastique appliquée ici à la conception orientée tâche de l'AUV. Les résultats de cette optimisation peuvent être utilisées comme une étape préliminaire dans la conception des AUVs, afin de donner des pistes pour améliorer les capacités de la propulsion.

La technique d'optimisation est également appliquée au robot RSM (fabriqué au sein de l'IRDL-ENIB) en modifiant seulement quelques paramètres de sa topologie propulsive. Cela afin d'obtenir des configurations de propulsion adaptées au cours d'une seule et même mission aux spécificités locomotrices des tâches rencontrées : Reconfiguration dynamique de la propulsion de l'AUV.

List of publications

The work on this manuscript has been published in the following international journals and conference:

- E. P. Vega, O. Chocron, and M. Benbouzid, "AUV Propulsion Systems Modeling Analysis," *Int. Rev. Model. Simulations (IREMOS)* , vol. 7, no. 5, pp. 827–837, 2014.
- E. P. Vega, O. Chocron, J. V. Ferreira, M. Benbouzid, and P. S. Meirelles, "Evaluation of AUV fixed and vectorial propulsion systems with dynamic simulation and non-linear control," in *IECON 2015 - 41st Annu. Conf. IEEE Ind. Electron. Soc.* IEEE, nov 2015, pp. 944–949.
- E. P. Vega, O. Chocron, and M. Benbouzid, "A Flat Design and Validated Model for AUV Reconfigurable Magnetic Coupling Thruster", *IEEE/ASME Trans. Mechatronics*, vol. 21, no. 6, pp. 2892-2901, Dec 2016.

Contents

List of Figures

List of Tables

Chapter 1

Résumé

1.1 Introduction

1.1.1 Véhicules sous-marins non habités

L'impact et le potentiel des océans dans le développement humain est indéniable, ils ont été tout au long de l'histoire un élément clé pour les domaines économique, militaire et scientifique. D'un point de vue économique, les mers sont les fondations de l'industrie de la pêche; elles sont également utilisés, depuis l'antiquité, pour établir les routes commerciales. Aujourd'hui, en temps de croissance démographique sans précèdent, les mers sont considérées comme un élément essentiel de l'économie par ses ressources (c.-à.-d nourriture, eau et énergie [1]). Les mers ont aussi un intérêt militaire, puisqu'elles sont considérées depuis toujours comme une composante importante pour la stratégie et la géopolitique. Enfin, les mers sont également importantes pour l'étude du changement climatique. Les variations dans leur composition chimique, donnent aux chercheurs des informations sur la dégradation du climat et de l'environnement [2].

L'humanité a toujours essayé d'exploiter les ressources de l'océan à l'aide de la technologie. D'abord avec des navires et après avec des sous-marins et de la technologie sous-marine. Dans la seconde moitié du XXe siècle, grâce à la maturité de la technologie marine, les robots sous-marins ont commencé à être utilisés dans l'exploration des océans. Des véhicules sous-marins non habités (UUV's, unmanned underwater vehicles en anglais) ont été développés à cette époque, puis leurs caractéristiques ont été progressivement améliorées grâce aux avancées technologiques [3]. On peut diviser les UUV's en deux catégories : robots sous-marins autonomes (AUV) et robots sous-marins télé-opérés (ROV). Actuellement, l'utilisation des ROVs est très répandue dans tous les domaines, alors que les AUV sont plutôt utilisés dans

le domaine militaire pour leur autonomie.

Contexte opérationnel

Après plusieurs décennies d'opération, les robots sous-marins autonomes ont démontré leur capacité à réaliser des missions de plus en plus difficiles. Ces robots étaient utilisés au début comme de simples dispositifs d'appui, puis, au cours des années, ils sont devenus plus fiables grâce aux progrès technologiques et scientifiques. Aujourd'hui ce type de robot peut être utilisé pour réaliser des missions complexes en autonomie ou en mode supervisé. Néanmoins, malgré leur succès, le développement des AUVs n'a pas encore atteint son potentiel maximum. Effectivement, la prochaine étape dans l'évolution des AUVs est l'amélioration de leur autonomie, ce qui leur permettra de réaliser les mêmes mission que les ROVs en autonomie. Le développement d'AUVs plus autonomes nous permettra de réaliser des missions dans lesquelles le robot pourra parcourir de grandes distances, réagir en temps-réel et réaliser des interventions sans supervision humaine.

Les domaines d'activité utilisant déjà les AUVs bénéficieront de ce type de robots sous-marins. Les océanographes, par exemple, pourront utiliser les AUVs pour mesurer sur des grandes distances et prendre des échantillons de façon autonome. L'industrie du pétrole pourrait utiliser ces robots pour inspecter et faire de la maintenance de son équipement sous-marin. La défense pourrait utiliser ces robots pour détecter, identifier et détruire des mines sans mettre en péril la vie humaine.

Même si ces missions sont impossibles à réaliser par les AUVs actuels, elles nous permettent de déterminer les caractéristiques nécessaires pour les futurs robot sous-marins. Afin de réaliser ces missions, nous remarquons qu'une bonne manoeuvrabilité et une bonne autonomie sont nécessaires. La combinaison de ces deux caractéristiques n'est pas disponible dans les AUVs d'aujourd'hui. Même si l'autonomie des AUVs a été améliorée, la manoeuvrabilité de ce type de véhicule reste réduite [27-29]. Une solution à ce problème serait soit d'augmenter l'autonomie des ROVs, soit d'augmenter la manoeuvrabilité des AUVs. Cependant, par leur nature, il est très compliqué de donner des capacités de longue portée aux ROVs. Ces véhicules sont étroitement attachés à leurs navire d'appui car ils y sont reliés par un câble ombilical.

La deuxième option est plus faisable : augmenter la manoeuvrabilité des AUVs. Effectivement, la capacité de calcul embarqué et les capteurs actuels permettent d'implémenter des méthodes de commande avancées. Ces développements, avec les avancements en termes de technologie propulsive,

sont susceptibles d'augmenter la manoeuvrabilité des AUVs et leur autonomie.

1.1.2 Méthodologie de recherche

Dans ce manuscrit on travaillera sur la seconde solution proposée : L'amélioration des capacités des AUVs par le biais d'une manoeuvrabilité augmentée. Effectivement, nous allons nous concentrer sur l'étude des systèmes de propulsion et leur efficacité afin d'augmenter la manoeuvrabilité des AUVs (mobilité et vitesse). La propulsion est un aspect clé dans l'étude de la manoeuvrabilité puisqu'elle intervient directement dans la mobilité et la commandabilité du robot sous-marin.

Selon le type de propulsion, le robot pourrait avoir besoin de plus ou moins de propulseurs, il pourrait être plus ou moins lourd, ou il pourrait même être incapable de réaliser certaines missions. La conception des systèmes de propulsion sera étudiée afin de déterminer son effet sur la performance des AUVs. Différentes solutions technologiques (fixe et vectorielle) seront étudiées. De plus, un propulseur magnéto-couplé réconfigurable (PMCR) sera présenté, modélisé et simulé.

Nous étudierons, également, la topologie de propulsion, c'est à dire le nombre, l'orientation, la position et le type d'actionneur. Nous essaierons de déterminer la configuration propulsive la plus adaptée pour une tâche donnée, ce qui comprend la topologie et la méthode de commande (structure et paramètres).

Le problème de trouver la configuration propulsive la plus adaptée sera traité ici comme un problème d'optimisation. A cause d'infinies possibilités dans les solutions et du manque de formalisme mathématique, il n'est pas possible d'utiliser des techniques conventionnelles d'optimisation (descente de gradient). Afin de résoudre ce problème d'optimisation on utilisera des algorithmes évolutionnistes (algorithme génétique). Ce type d'algorithme utilise l'évolution artificielle afin de trouver des solutions adaptées au problème d'optimisation.

L'AUV étant autonome, il est équipé de batteries. Les solutions proposées ne doivent donc pas conduire à une forte consommation énergétique. Ce compromis (manoeuvrabilité et efficacité énergétique), nous amènera à privilégier des solutions avec un faible nombre de propulseurs idéalement configurés pour maximiser leur potentiel.

1.2 Modelisation de l'AUV

1.2.1 Modélisation des robots sous-marins autonomes

L'étude des robots sous-marins commence par la modélisation mécanique du système, une étape fondamentale pour maîtriser, commander et optimiser n'importe quel robot. Les robots sous-marins nécessitent plusieurs modèles pour être complètement décrits. Les principaux modèles représentent les aspects cinématique et dynamique du système. De plus, on utilisera des modèles décrivant les différents sous-systèmes tels que la topologie ou la technologie propulsive. Ces modèles seront présentés dans cette section. Les premiers seront issus de la littérature [146]. Ils font usage de l'approximation par les masses ajoutées (coefficients hydro-dynamiques), sa précision est suffisante pour notre application.

Cinématique

Ici les équations cinématiques de l'AUV sont présentées. Afin de pouvoir les développer, on doit faire les hypothèses suivantes :

- L'AUV est submergé loin du sol, des parois et de la surface de l'eau. Il est dans un fluide homogène.
- Les courants marins sont négligés (non fondamentales pour la méthode).
- L'AUV est un corps rigide de masse constante.

Deux systèmes de coordonnées orthogonaux sont utilisés : R_0 $(O_0, \mathbf{x}_0\mathbf{y}_0, \mathbf{z}_0)$ le repère fixe (attaché à la terre) et R_b $(O_b, \mathbf{x}_b, \mathbf{y}_b, \mathbf{z}_b)$ le repère mobile, lié a l'AUV.

On peut trouver le vecteur de vitesse absolue de l'AUV exprimé dans R_0 à partir de R_b en utilisant une matrice de transformation cinématique [146].

$$\dot{\boldsymbol{\eta}} = \left.\frac{d\boldsymbol{\eta}}{dt}\right|_{R_0} = \mathbf{J}(\boldsymbol{\eta}_2)\,\boldsymbol{\nu}$$

avec $\boldsymbol{\nu} = [u\ v\ w\ p\ q\ r]$, le vecteur combinant les vitesses absolues linéaire et angulaire du robot dans R_b

Dynamique

Les équations dynamiques non linéaires du robot sous-marins sont données dans le repère R_b et peuvent être développées comme [3] :

$$\mathbf{M}\,\boldsymbol{\nu} + \mathbf{C}\,\boldsymbol{\nu} + \mathbf{D}\,\boldsymbol{\nu} + \mathbf{G} = \boldsymbol{\tau}$$

où $\mathbf{M} \in \mathbb{R}^{6\times6}$, $\mathbf{C} \in \mathbb{R}^{6\times6}$ et $\mathbf{D} \in \mathbb{R}^{6\times6}$ sont les matrices de masse, Coriolis et forces centripètes et forces de dissipation respectivement (incluant les termes de masse ajoutée). $\mathbf{G}$ est le vecteur des forces et moments gravitationnels. Enfin, $\boldsymbol{\tau}$ est le torseur des efforts extérieurs.

$\boldsymbol{\tau}$, représente les forces et moments générés par les propulseurs. Il est construit de la manière suivante :

$$\boldsymbol{\tau} = \mathbf{B}\ \mathbf{u}_p$$

où $\mathbf{B}$ est la matrice de configuration de la propulsion, qui dépend de l'architecture de propulsion (nombre, position et orientation des actionneurs). $\mathbf{u}_p$ est le vecteur des forces d'actionnement (c.-à-d. poussées des propulseurs).

Ces matrices sont créées en utilisant les lois de la physique du solide rigide et hydro-dynamiques. La physique du solide rigide nous permet de modéliser l'AUV comme un solide, donc d'appliquer les équations dynamiques du solide (Newton - Euler). Par ce que l'AUV est submergé, on doit ajouter l'effet de l'eau sur le robot afin de décrire complètement le système dynamique.

1.2.2 Modélisation de la propulsion

La prochaine étape dans la modélisation du robot sous-marin concerne sa propulsion. Nous pouvons diviser la modélisation de la propulsion en deux niveaux. Le premier, le plus global, décrit mathématiquement la topologie (nombre, position et orientation) des propulseurs. Le second niveau, plus spécifique, décrit la technologie utilisé dans chaque propulseur. Dans ce manuscrit on présente deux technologies de propulsion : Fixe et vectorielle.

Propulsion fixe

Un propulseur fixe est un propulseur qui une fois monté dans le robot, n'est pas capable de changer sa configuration (orientation). Ce type de propulseur est très utilisée dans le domaine de la robotique sous-marine. Il existe plusieurs modèles décrivant ce type d'actionneur, dans ce manuscrit on utilisera le modèle le plus utilisé [147][148].

A fin de développer le modèle pour ce type d'actionneur, on prendra comme référence les actionneurs utilisés dans le robot RSM : Le propulseur *Seabotix* BTD150 [1], un propulseur avec un moteur à courant continu.

Propulsion vectorielle

Un propulseur vectoriel, est un propulseur capable de changer la direction de sa poussée. A fin de modéliser ce type d'actionneur, on utilisera le propulseur magnéto-couplé reconfigurable (PMCR) comme base [150].

La figure 1.1 montre un schéma du PMCR. Le système consiste en un moteur attaché à un axe. Un axe attaché à l'hélice fournit la force de propulsion. L'axe de propulsion est orientable par rapport à l'axe du moteur grâce à une fourche, le lien entre le bâti du système et l'axe du propulseur. Le couple généré par le moteur est transmis à l'hélice grâce à un accouplement magnétique.

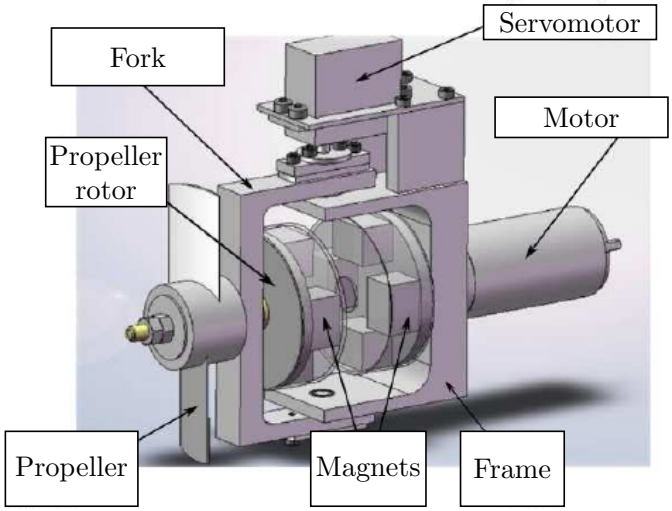

Figure 1.1: Prototype du PMCR.

1.3 Commande basée modèle

Après avoir modélisé le robot, la prochaine étape dans l'étude du robot sous-marin est la définition des méthodes de commande et d'estimation utilisées, deux éléments indispensable pour simuler le robot en mission.

[1] http://www.seabotix.com/products/auv_thrusters.htm

Étant donné qu'on a un modèle assez précis du robot, il semble logique de l'utiliser intensivement. La méthode choisie est le bouclage dynamique linéarisant; Une technique de commande non-linéaire qui utilise complètement les modèles cinématique et dynamique du robot.

Un filtre de Kalman étendu (EKF) est utilisé afin d'estimer les variables d'état non mesurées par des capteurs, mais utilisées par la méthode de commande. L'implantation des méthodes de commande et estimation est faite en utilisant le système d'exploitation robotique ROS (bibliothèque écrite en C++).

1.3.1 Bouclage dynamique linéarisant

Cette méthode de commande est très utilisée pour commander des AUVs [160,161]. La commande utilise directement les modèles cinématique et dynamique du robot. Cette caractéristique nous donne une vision claire du fonctionnement intérieur de la méthode, ce qui nous permet de mieux étudier la commandabilité du système. L'idée principale de cette méthode de commande est de transformer algébriquement le système non-linéaire en un système linéaire, afin de pouvoir appliquer des techniques de commande linéaires (Fig. 1.2).

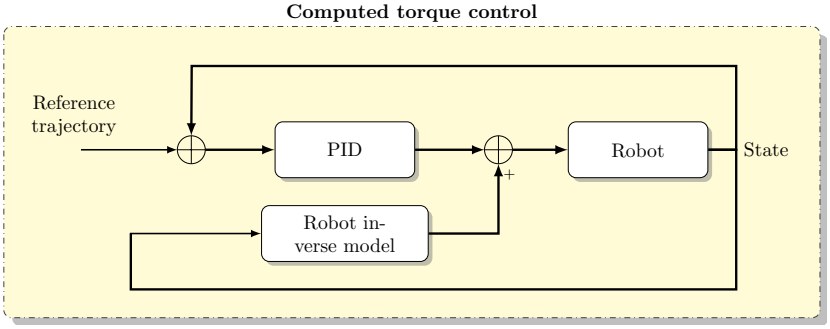

Figure 1.2: Principe de la méthode de commande utilisée.

La méthode du bouclage dynamique linéarisant est différente des méthodes classiques de linéarisation puisque la linéarisation est faite avec une compensation dynamique exacte et non pas par approximation linéaire. Le bouclage dynamique linéarisant est basé sur l'idée de commander le système en utilisant un actionnement subsidiaire capable de compenser les effets non-linéaires du système et de pouvoir appliquer une correction au système ainsi linéarisé.

La méthode est constituée des boucles cinématique et dynamique (Fig. 1.3). La boucle cinématique génère une vitesse de référence $\boldsymbol{\nu} u_{ad}$ qui fait

qu'un point du robot (point traqué) e suive une trajectoire désirée $\boldsymbol{\eta}_{ed}$. Afin de calculer cette vitesse, il utilise la vitesse de la trajectoire désirée $\dot{\boldsymbol{\eta}}_{ed}$ (anticipation) et un terme de correction proportionnel (gain Λ) à l'erreur de position/orientation.

La boucle dynamique calcule le torseur $\boldsymbol{\tau}_a$ à appliquer à l'AUV (compensation) afin qu'il suive ν_{ad}. Elle utilise aussi une anticipation (dérivée de $\boldsymbol{\nu}_{ad}$) et une correction proportionnelle à l'erreur en vitesse (gain K_p). Cette méthode est très efficace si l'architecture de propulsion est capable de générer le torseur calculé $\boldsymbol{\tau}_a$ et si le modèle est très précis (ou la période d'échantillonage très petite). Il est indispensable que le robot soit commandable sur la trajectoire $\boldsymbol{\eta}_{ed}$.

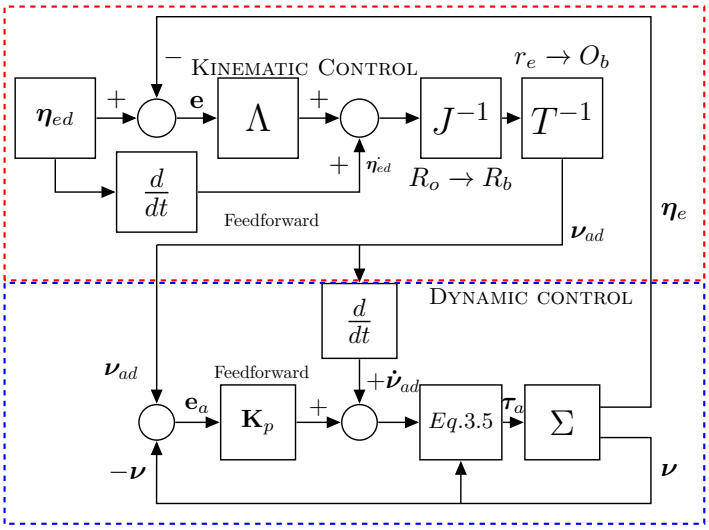

Figure 1.3: Commande par bouclage linéarisant.

1.4 Optimisation génétique de la propulsion

Le modèle et la méthode de commande présentés précédemment montrent les différents paramètres et sous-systèmes auxquels le concepteur doit répondre afin de créer le système de propulsion de l'AUV. Les paramètres et sous-systèmes tels que les gains de la commande et la topologie de propulsion du robot sont essentiels pour créer un robot opérationnel. De plus, même pour un robot sous-marin fonctionnel, un léger changement dans les paramètres de conception peut générer de grandes variations de la performance du robot. Le robot sous-marin doit être alors une combinaison précise d'éléments de conception étroitement couplés.

La grande quantité d'éléments de conception et leur hétérogénéité, font de la conception du robot un processus très complexe. Pour surmonter cette difficulté, le concepteur peut choisir les stratégies suivantes :

- Utiliser une approche descriptive, la méthode classique où "humaine". Le concepteur utilise l'expérience précédente et des nouvelles idées afin de créer le robot. Les idées sont transformées de façon itérative jusqu'à ce qu'on trouve une idée satisfaisante. Ce processus étant semi-empirique, le robot créé avec cette approche dépend fortement de la capacité et de l'expérience du concepteur.

- Utiliser une approche numérique, qui nécessite un ordinateur pour créer le système ou pour analyser un système déjà conçu. Cette approche peut être utilisée pour trouver automatiquement les éléments par optimisation d'un système selon des critères objectifs déterminés.

Le problème de conception dans ce travail peut être formulé comme un problème d'optimisation.

1.4.1 Algorithmes génétiques

Ce problème d'optimisation sera résolu en utilisant des algorithmes génétiques. Étant un type d'algorithme évolutionniste, les algorithmes génétiques (GA) sont basés sur la théorie de l'évolution de Darwin (1859) et les mécanismes de la génétique découverts par Mendel (1866). Les algorithmes génétiques utilisent des individus, regroupés en populations, une fonction objective ou *fitness* et des opérateurs génétiques tels que sélection, croisement et mutation [175].

Un algorithme génétique typique suit les étapes suivantes :

1. Une population d'individus est créée. Cette population peut être créée aléatoirement ou en utilisant des connaissances antérieures.

2. Les individus de la population sont évalués et se voient attribuer une fitness selon leur performance (basée sur une fonction objective d'évaluation de leur gènes (codés en nombres binaires)

3. Les opérateurs génétiques utilisent la *fitness* de chaque individu afin de créer de nouveaux individus :

   (a) Sélection : les meilleurs individus (meilleures fitness) sont sélectionnés. Cette sélection peut être stochastique ou déterministe (les deux souvent).

(b) Croisement : deux individus (parents) combinent leurs gènes afin de créer deux nouveaux individus (enfants).

(c) Mutation: un changement de la valeur d'un gène chez certains individus est réalisé aléatoirement. Ce changement (de 0 à 1 ou inversement) est décidé avec des méthodes stochastiques (chaque gène a une très petite probabilité d'avoir une mutation).

(d) Remplacement de la population: la nouvelle génération créée par les opérateurs remplace complètement la génération précédente.

4. Aller à l'étape 2 si les solutions ne sont pas encore satisfaisantes, autrement, finir l'algorithme.

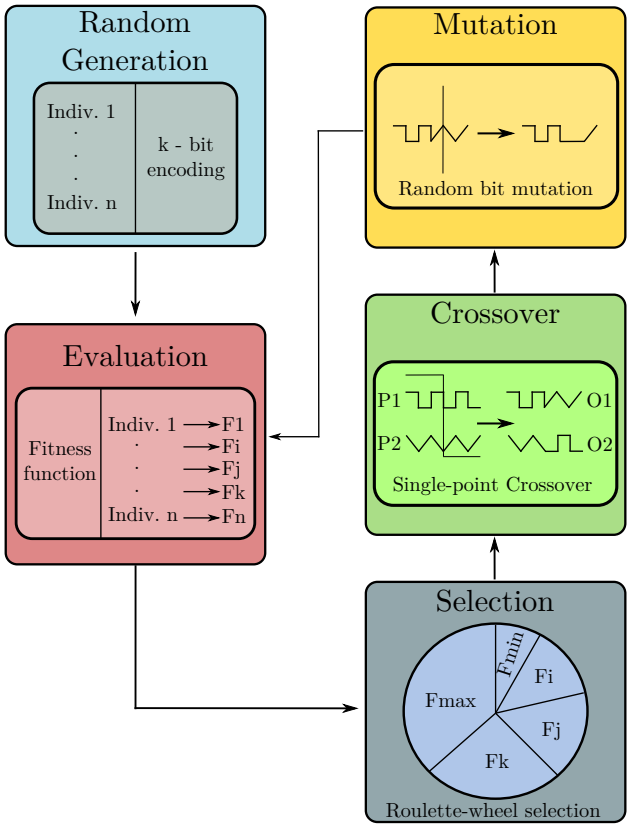

Figure 1.4: Algorithme génétique

Problème d'optimisation

Le problème d'optimisation "créer un système de propulsion capable de réaliser une mission donnée" doit être défini plus précisément. Évidemment, une définition mathématique et concise du problème nous permettra de concevoir et implémenter correctement l'algorithme génétique.

La commande calcule la force (donc la poussée) qui doit être fournie par les propulseurs d'un robot sous-marin afin de qu'il suive une trajectoire donnée. On peut considérer aussi qu'une mission de haut-niveau est une succession de trajectoires enchaînées. Notre méthode de commande nous permet alors de réaliser une mission spécifique tout en suivant des trajectoires successives différentes.

Le robot solution pour le problème d'optimisation doit suivre précisément la trajectoire désirée. Cela signifie que la distance entre le point traqué et la trajectoire (à l'instant correspondant) doit converger vers zéro. Autrement dit, l'intégrale de l'erreur de la trajectoire doit être la plus faible possible et converger. Le problème d'optimisation est, alors, la minimisation de la fonction objective définie comme l'intégrale de l'erreur cinématique ϵ_{kin}. Cela peut être écrit mathématiquement de la façon suivante (cas de la minimisation) :

$$\text{Trouver } \mathbf{x}^\star \in I\!R|_{[-1,1]}^{5n+5} \ /(\forall\, \mathbf{x} \in I\!R|_{[-1,1]}^{5n+5},\ g(\mathbf{x}^\star) \leq g(\mathbf{x})$$

avec :

$\mathbf{x}^\star$	L'individu le plus adapté créé à partir de $5n+5$ gènes binaires (bits). L'espace de recherche est $I\!R|_{[-1,1]}^{5n+5}$
$g(x)$	La fonction objective à minimiser , définit comme l'intégrale de l'erreur cinématique de la méthode de commande.
n	La quantité de propulseurs.

Le nombre de gènes d'un individu peut changer. Chaque propulseur a 5 paramètres : 3 coordonnées de position (x_{pi}, y_{pi}, z_{pi}) et 2 orientations (θ_{pi}, ψ_{pi}), définies par rapport au repère du robot. Étant donné qu'on cherche également la meilleur topologie propulsive, le nombre de propulseurs n est variable.

Le robot a seulement un seul contrôleur, ce qui nous donne 5 paramètres à optimiser : 2 gains (pour les parties cinématiques et dynamiques de la méthode de commande) et 3 coordonnées de position (x_e, y_e, z_e) pour la position du point traqué e.

Paramètres de conception

Une liste exhaustive des paramètres de conception est donnée ici :

- Paramètres de la propulsion (pour le propulseur “i“)
  - Nombre de propulseurs : n
  - Position du propulseur : P_{ix}, P_{iy}, P_{iz}
  - Orientation du propulseur : θ_i, ψ_i
- Paramètres de la méthode de commande
  - Gains (cinématique et dynamique) : Λ, K_p
  - Point traqué : P_{ex}, P_{ey}, P_{ez}

Avec i le numéro du propulseur de la topologie.

1.4.2 Optimisation globale des robot sous-marins autonomes

Dans cette étape de l'optimisation des robot sous-marins, on déterminera la topologie complète du véhicule afin de faire une mission déterminée. On optimisera le nombre de propulseurs, les positions et orientation des propulseurs et les gains du contrôleur.

Mission

La mission qu'on utilisera pour l'optimisation consiste à inspecter une hydrolienne. Cette mission a été choisie à cause du coût élevé de la maintenance des hydroliennes. Ces dispositifs pourraient être inspectés par des robots sous-marins avec une manoeuvrabilité et une autonomie améliorée.

La mission d'inspection (Fig. 1.5) peut être divisée en trois phases :

- **Scann du fond marin :** dans cette étape, le robot suit une trajectoire qui lui permet de scanner le fond marin (trajectoire dans le plan proche de la surface). Le robot sous-marin peut utiliser un scanner latéral pour scanner le fond marin jusqu'à ce qu'il trouve la cible.
- **Plongé vers l'hydrolienne :** une fois la cible trouvée, le robot commence la phase de plongée. Dans cette phase le véhicule essaiera de se rapprocher rapidement et directement de la turbine, tout en essayant de minimiser la consommation énergétique.

- **Tomographie (inspection circulaire) de la turbine:** dans la troisième étape, le robot commence l'inspection de l'hydrolienne. Cette étape est réalisée en utilisant une caméra afin de filmer la turbine. Pour cela, le robot doit faire un cercle de rayon constant autour de la turbine tout en restant radial au cercle (en pointant constamment l'hydrolienne).

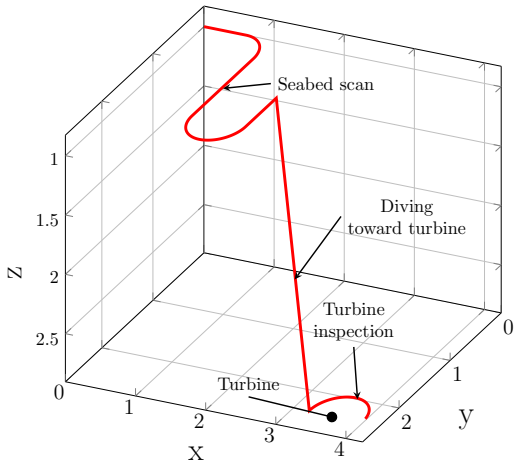

Figure 1.5: Trajectoires des différentes phases de la mission.

On appliquera l'algorithme génétique à chacune de ces trois étapes.

1.4.3 Reconfiguration dynamique de la propulsion

La méthode d'optimisation nous permet d'obtenir des solutions intéressantes en termes de performance pour chaque phase, mais difficilement pour la mission complète. Une solution à ce problème est de réconfigurer dynamiquement le robot, selon la trajectoire suivie. Le robot changera alors sa configuration propulsive pendant la mission, sous l'eau et de manière autonome. Actuellement, la réconfiguration de l'orientation des propulseurs est technologiquement possible (section 3.1.3.2).

Dans cette section on testera l'algorithme dans une application plus "réaliste". On essayera de trouver des solutions pour les trajectoires décrites précédemment, mais avec un nombre fixe propulseurs et avec des positions fixes (configuration du robot RSM). Seulement les orientations des propulseurs et les paramètres de la méthode de commande seront optimisés. Cette solution est réaliste, car elle peut être appliquée en utilisant les propulseurs magnéto-couplés reconfigurables (PMCR) (section 4.2.3).

1.4.4 Discussion

En analysant les résultats de la réconfiguration dynamique et de l'optimisation globale (Tableau 1.1), on voit que l'approche global a de meilleurs résultats. Pour le scan du fond marin, la fitness du meilleur individu trouvé avec l'approche global est 1.35 fois plus grande que celle trouvée avec la réconfiguration dynamique. On obtient un résultat similaire pour la plongée, l'optimisation globale est 1.38 fois plus grande que celle trouvée avec la réconfiguration dynamique. Cependant, pour la tomographie, la réconfiguration dynamique trouve une meilleure solution que l'optimisation globale (1.09 fois plus adaptée). Cela pourrait s'expliquer par l'espace de recherche réduit de la reconfiguration dynamique et par le fait que la configuration de base était proche de celle adaptée à la tâche.

Table 1.1: Résultats obtenus avec reconf. dyn et opt. globale.

Approche	**Tâche**	**Meilleure fitness**
Opt. Globale	Scan du fond marin	47.97
	Plongée	50.2
	Tomographie	32.45
Reconf. dyn	Scan du fond marin	35.52
	Plongée	36.4
	Tomographie	34.17

Concernant les valeurs de la fitness, l'optimisation globale est la plus performante. Cependant, d'un point de vue technologique, la réconfiguration dynamique semble plus intéressante. Effectivement, même si les valeurs de fitness de la réconfiguration dynamique ne sont pas si bonnes, le fait que les configurations soient adaptables dynamiquement, rend ces solutions plus utiles.

Les avantages de la de la reconfiguration dynamique sont claires, elle permet aux AUVs sous-actionnés de réaliser des tâches complexes, grâce à l'amélioration de leurs capacités propulsives. Effectivement, les AUVs équipées de propulseurs réconfigurables, en utilisant les solutions trouvées par la réconfiguration dynamique, sont capables de réaliser des tâches avec une bonne convergence vers la trajectoire désirée et une consommation énergétique réduite. Cela permet d'augmenter l'agilité des AUVs et leur autonomie de déplacement.

1.5 Conclusion

Pour concevoir un véhicule sous-marin autonome on doit définir une grande quantité de paramètres de conception. Ces paramètres doivent être sélectionnés et correctement configurés afin d'obtenir une performance et une autonomie améliorées. Une manière d'améliorer l'autonomie est d'incrémenter les capacités propulsives, est d'adapter le robot aux tâches qu'il doit réaliser.

Actuellement, les AUVs doivent réaliser des tâches et des missions de plus en plus exigeantes. Ces nouvelles missions demandent aux AUVs d'être très efficaces et autonomes.

La majorité des méthodes de conception comptent sur l'expérience précédente, prennent des solutions très connues et adoptent des approches classiques. Les robots conçus avec ces méthodes, ne sont pas optimisés pour réaliser des missions spécifiques. Ces robots ont une manoeuvrabilité et efficacité médiocre.

Cette thèse propose une manière d'améliorer la conception de l'AUV à travers l'optimisation de leur configuration propulsive, ce qui permet d'incrémenter leur performance et leur autonomie. Effectivement, une configuration propulsive adaptée à la mission à réaliser, permet aux AUVs de réaliser des tâches complexes avec plus d'efficience (efficacité à coûts réduits).

La technique d'optimisation utilisée dans cette thèse utilise un simulateur de l'AUV afin d'évaluer les solutions trouvées. Ce simulateur, exploite le modèle hydro-dynamique du robot, avec la dynamique du solide rigide et les effets hydro-dynamiques (masse-ajoutée, force de flottaison et forces de traînée).

Le simulateur inclut également le modèle électromécanique d'un propulseur réel. Un propulseur vectoriel a été étudié aussi dans cette thèse afin d'étudier sa faisabilité. Le propulseur vectoriel permet de changer la direction de la poussée grâce à un accouplement magnétique reconfigurable. Le modèle dynamique de ce dispositif a été développé et validé.

Afin de simuler correctement le comportement du robot, on utilise une méthode de commande non-linéaire appelée bouclage dynamique linéarisant. Cette méthode de commande utilise les modèles cinématique et hydro-dynamique afin de compenser les effets hydro-dynamques et diriger l'AUV vers la trajectoire désirée. La méthode de commande calcule le torseur nécessaire pour suivre la trajectoire désirée. Une étape supplémentaire, appelée répartition de la poussée, a été ajoutée afin de déterminer les poussées nécessaires dans chaque propulseur du robot en fonction de la topologie propulsive.

Un filtre de Kalman étendu et son implantation dans l'architecture de contrôle a été également présenté. Cette étape n'a pas été utilisée dans l'optimisation génétique (car rendue non nécessaire par la simulation) mais elle est indispensable dans le développement du robot expérimental réel (qui

ne dispose pas de capteurs de vitesse).

En utilisant les éléments mentionnés précédemment, l'algorithme génétique a été mis en place. Il nous permet, à partir d'un robot et une tâche donnés, de trouver une topologie propulsive et des paramètres du contrôleur optimales. La solution trouvée sera adaptée pour la tâche demandée et le robot solution sera très performant mais spécifique pour cette tâche. Les solutions trouvées par cette méthode sont difficiles à appliquer dans la réalité, donc une approche alternative a été présentée. Cette approche, appelée reconfiguration dynamique, propose d'appliquer la méthode d'optimisation à un robot avec le nombre de propulseur et leur positions fixés. Ce rétrécissement de l'espace de recherche peut donner des solutions moins efficaces. Cependant, ces solutions sont techniquement faisables aujourd'hui. En appliquant la reconfiguration dynamique au même robot pour différentes tâches, on peut trouver la configuration optimale de la propulsion pour chaque tâche. Cette information nous permettra, à l'aide des propulseurs reconfigurables, de créer des robot sous-marins autonomes reconfigurables dans un futur proche.

Les travaux futurs se concentreront sur l'amélioration de l'algorithme génétique. Cela pourrait être fait en améliorant son efficacité, en utilisant des opérateurs avancés où en réduisant l'espace de recherche. Le nombre de paramètres de conception pourrait être incrémenté sans augmenter le temps de calcul.

Le jeu de paramètres étendu pourrait inclure différents types de commande (PID, sliding mode, commande adaptative, etc). Effectivement, l'algorithme génétique pourrait choisir le type de contrôleur, selon la tâche. Les termes de correction utilisés dans la méthode présentée précédemment pourraient être améliorés également, on pourrait utiliser des correcteurs intégral et dérivé.

L'algorithme génétique pourrait être amélioré avec des modèles plus précis au niveau de la méthode de commande (on pourrait y inclure la dynamique des propulseurs, par exemple). Effectivement, un modèle avec une description hydro-dynamique plus précise et un modèle dynamique des propulseurs pourrait améliorer les solutions trouvées et les rendre plus facilement applicables dans la réalité en augmentant notamment la robustesse du contrôle-commande en situation réelle..

Enfin, des expérimentations en bassin avec les solutions trouvées par l'algorithme génétique doivent être réalisées afin de valider la méthode. Cela nécessitera la conception et la fabrication d'un AUV expérimental à propulsion reconfigurable. Ceci pouvant être fait en développant le PMCR ou en trouvant des solutions alternatives.

Chapter 2

Introduction

2.1 Unmanned underwater vehicles

Oceans impact and potential in human development is undeniable, they have been throughout history a key element in economic, military and, academic activities. From an economic point of view, they are the foundation of the fishing industry and they are used to establish reliable trade routes since ancient times. Nowadays, in times of unprecedented human population growth, they are regarded as a crucial asset, thanks to its large amount of important resources such as food, water and energy [1]. Oceans are also a topic of concern for military reasons, since they have always been considered a strategic aspect in geopolitics. A testimony of the strategic importance of the oceans are the words of Charles De Gaulle, that in his speech in Brest in 1969 foresaw that: "*Men activity will be focused on the research and utilization of the sea. Naturally, states ambitions will seek to dominate the sea to control its resources*". Lastly, oceans also play a key role in the study of environmental and climate change. Variations in measured data such as chemical composition, for example, give scientists informations about the degradation of the environment [2].

The importance of the ocean is well known by mankind since the dawn of civilization. This is why, non surprisingly, humans have always tried to harness its resources. All efforts for mastering oceans have been aided by technology. First with surface ships and afterwards with submarines and deepwater technology. In the second half of the 20^{th} century, thanks to marine technology maturity, marine robotics made its appearance in ocean exploration. Namely, Unmanned Underwater Vehicles (UUVs) started being developed and since then, advancements in technology allowed to have a rapid progression in their capabilities [3]. Nowadays, UUVs have become

increasingly prevalent and are used for scientific, military and industrial purposes. Current UUVs capabilities allow to perform tasks inconceivable fifty years ago [4–6].

2.1.1 Underwater robots classification

Unmanned Underwater Vehicles can be divided in two large groups [3]: Remotely Operated Vehicles (ROVs) and Autonomous Underwater Vehicles (AUVs), each type of underwater robot presents a set of characteristics, which are discussed next.

ROVs

Remotely Operated Vehicles (ROV) are a tethered type of underwater robot. They are controlled by an umbilical cable which is used to transmit orders, energy, and data between the robot and its operators. The operators pilot the ROV from a vessel or platform (or sometimes via a surface unmanned vehicle), controlling most of the robot subsystems [7, 8].

The first ROV, the Poodle, was developed by the Frenchman Dimitri Rebikoff in 1953. The U.S. Navy developed in 1966 the CURV (Cable Controlled Underwater Recovery Vehicle), a ROV that became famous for recovering an atomic bomb from Spanish coasts. After this promising start, the ROV gained popularity in the Oil and Gas industry which rapidly adopted this technology to support offshore interventions.

Remotely Operated Vehicles are typically equipped with at least six thrusters, which actuate the six degrees of freedom of the robot, that makes them fully actuated systems [9]. Thanks to this characteristic, ROVs are very maneuverable, capable of hovering, for stationary interventions (including light work) with good precision.

Maneuverability made of ROVs an important asset, since their applications are very diverse. Remotely Operated Vehicles are used nowadays for commercial, military and scientific purposes. Classical applications include hull inspection [10], drilling [8] and construction support [11], object recovery [12], marine mine destruction [13] and environmental investigation [14].

Despite its many advantages, ROVs have a fundamental drawback: lack of autonomy. Indeed, due to the fact that they are controlled by an operator, this type of robots can only work efficiently if they are assisted by nearby support vessels. Experienced and skilled operators on board of these support vessels receive images from the ROV and use them to pilot the robot. Moreover, deployment and recovery operations are very difficult and dangerous for the robot, they can be used only in calm weather. Furthermore, these

robots use several thrusters in order to be fully maneuverable, making them very dependent of the pilot skills.

AUVs

Autonomous Underwater Vehicles (AUV) are a type of underwater robot that operates with on-board control and power supply. They navigate using a combination of preprogrammed trajectories (or reactive motions), acoustic ranging and detection, GPS technology and inertial sensors. The distinctive property of these underwater vehicles is their autonomy, allowing them to cover large distances to achieve complex missions without assistance.

The first AUV, the SPURV (Special Purpose Underwater Research Vehicle), was developed by the University of Washington in 1957. It was used to support oceanographic research and it had an autonomy of four hours. Since AUVs use an on-board controller, they need powerful processing capabilities. This stalled the development of this type of vehicle until technology was ready to provide sufficient computational power. With technology advances in processing capabilities, power storage and sensors performance, AUV became more reliable and autonomous [15, 16].

Given its autonomous nature, AUVs are mostly used to perform survey tasks [17] and are designed to be energy efficient. Designers aim at reducing the power consumption by using hydrodynamic hulls [18], since this reduces drag forces. Unlike ROVs, AUVs are typically under-actuated, i.e. they have fewer actuators than the Cartesian space degrees of freedom. A common propulsive strategy has one ot two rear propellers plus control surfaces (rudders), this configuration is adequate to perform long range missions. However, AUVs using this configuration need to advance to be able to turn, making them incapable of performing swerves, which are essential in a maneuverable robot. Consequently, energy efficiency and long range capabilities come at expense of maneuverability.

Autonomous Underwater Vehicles are used in tasks where long distances have to be covered. Typical applications for this kind of vehicle are pipeline and cable inspection [19], under-ice exploration [20], port surveillance [21], mine warfare [22], hydro-graphic survey [23], location and study of shipwrecks [24], among others like off-shore platform or airplane accidents.

Even thought AUVs have improved greatly in the last decades, there are still issues that need to be solved. The first one is insufficient power storage capabilities, since technology nowadays does not provide an efficient enough power source [25, 26]. The second issue is AUV autonomy which remains limited up to these days [17]. To solve this problem, efforts are needed in

control methods, sensor technology and propulsion systems. Indeed, better propulsion systems will improve maneuverability which will allow AUVs to perform more complex tasks such as close object inspection, line or surface following despite marine currents or performing sharp turns. The result of a superior maneuverability will be a significant increase in AUVs overall autonomy.

2.1.2 Motivation

Decades of development have proven the ability of AUVs to achieve increasingly complex missions. These robots started as mere support devices, and throughout the years thanks to scientific and technological advancements, have gained reliability. Nowadays these robots can be trusted enough to carry out fairly complicated missions on their own or on supervised mode.

However, despite their success, AUVs development has not yet reached its full potential. Far from being in its maturity, AUVs capabilities can still be greatly improved. Indeed, the next milestone in the evolution of AUVs is the ability to accomplish autonomously the same tasks than ROVs. This would result in underwater robots capable of traveling long distances, react in real-time to unpredicted events and perform immediate intervention without the supervision of a human operator.

Sectors that already use AUVs would evidently benefit from robots of said characteristics. For oceanographers, for example, this would mean having a robot capable of scanning large areas of the sea floor and taking samples autonomously. Gas and oil industry could use these robots not only for inspecting pipelines but also to perform maintenance on their undersea equipment. Renewable Marine Energy industry could use AUVs to inspect and maintain their generators. Military would benefit of robots capable of detecting, identifying and/or destroying mines without jeopardizing human lifes.

Even if these tasks seem nowadays far from AUVs capabilities, they allow us to determine the characteristics that an improved AUV would need. In order to accomplish these tasks, we note that good maneuverability and autonomy are needed. The combination of these properties is not yet available on current AUVs. Even if great advances have been accomplished in terms of autonomy, nowadays AUVs have reduced maneuverability [27–29].

A solution for this problem would be to give more autonomy to ROVs or more maneuverability to AUVs. However, given its nature, it is intricate to provide long range capabilities and autonomy to ROVs. The need of an operator and an umbilical cable makes these vehicles fundamentally tied up to their support vessels.

The second option seems more feasible: to give AUVs more maneuverability. Indeed, current computing processing and sensor technologies allow to implement sophisticated control methods, which added to developments in propulsive strategies and technology, would increase AUVs maneuverability and, therefore, its autonomy.

2.2 Research methodology

In this manuscript we work on the second solution, which is improving AUVs autonomy by means of enhanced maneuverability. More precisely, we will focus on the study of propulsion systems and their efficiency in order to achieve higher maneuverability (mobility and speed) for autonomous vehicles.

Propulsion is a key aspect of maneuverability since it can deeply affect the robot mobility and control strategy. Depending on the propulsion, a robot might need more or less actuators, be more or less heavy, or be incapable of performing some missions. For this reason, propulsion systems design will be studied in order to determine their impact in the performance of AUVs. Technological solutions for AUV propulsion will be analyzed and modeled as well. Additionally, a reconfigurable magnetically coupled thruster (RMCT) is presented, modeled and simulated.

Another important aspect of propulsion that will be covered in this work is the propulsion topology, i.e. the number, orientation, position and type of actuators. We will try to determine the most suited propulsive configuration for a given task, this includes topology and the control system (i.e. control methods and parameters).

Finding the most convenient propulsive configuration will be treated as an optimization problem. Giving the infinite available possibilities and the lack of an unifying mathematical formalism to optimize, we can not use classical optimization techniques (gradient based search). In order to solve this optimization problem, we use evolutionary algorithms, which use the power of artificial evolution to find suitable solutions to our problem.

Given that AUVs have an on-board power storage system, it is important that improvements in maneuverability do not increase power consumption beyond measure. This trade-off (maneuverability vs. power efficiency) will guide us during our studies. In practice, it will lead us to favor solutions consisting in a small number of multipurpose thrusters.

2.3 Organization of this manuscript

This manuscript is organized in four additional chapters prior to conclusions. Chapter 2 presents a state of the art of underwater robot design and establishes the parameters and methods that will be used in order to improve maneuverability in AUVs. Chapter 3 introduces the hydrodynamic model of AUVs and propulsion systems, also it describes the EAUVIVE[1] dynamic simulator. Chapter 4 discusses control and estimation techniques applied to our AUV simulation. Validation of the control system by "*in-the-loop*" methods is presented. Chapter 5 shows the implementation of several evolutionary algorithms for the task-based optimization of AUVs propulsion configuration. Three methods are presented: sequential, global and dynamic configuration optimization. Afterwards, a conclusion discussing results, advantages and disadvantages of the presented methodology will be given. Additionally, prospects opened by this work will be presented.

[1]ENIB AUV In Virtuo Experiment (IRDL-ENIB)

Chapter 3

Underwater robot design: State of the art

3.1 Sub-systems design

The first step in the design process of an underwater vehicle is to determine its sub-systems. These are the systems that define the robot features, and the designer needs to modify in order to create a feasible vehicle. They constitute the building blocks of the final system.

Given that underwater robot systems are object of extensive scientific research since decades, a state of the art of these sub-systems is needed. This is important in order to have a clearer view of the new ideas and features of underwater robots.

In this state of the art, the design and characteristics of the hull will be discussed. Propulsive characteristics of these vehicles, such as actuated DOF and propulsive technologies will be reviewed. Lastly, the more common control strategies will be analyzed.

After reviewing the main design parameters of AUV, this section will focus on the study of design paradigms. Indeed, design paradigms explain the steps necessary to design a system, starting from the idea until the complete definition of said system. Different design methods will be reviewed, ranging from classical experience-based and *ad-hoc* ones to the more globally applicable ones.

3.1.1 Hull design

When designing an AUV, one of the most important components is its hull (sometimes called shell or external structure). This component not only

determines how the robot will look like, but also will have a deep impact in the overall performances of the resulting system. As such, the design of the hull can not be taken lightly.

The characteristics of the hull depend on the application of the AUV, however, when designing a hull some general aspects need to be considered:

- External conditions: working pressure and temperature, impacts etc.
- Size requirements.
- Accessibility to internal components.
- Practicality and versatility to add new components.
- Operational conditions (speed, maneuverability or thruster needs)

As pointed in the list the hull must be able to resist the hydrostatic pressure of the water at working conditions. Additionally, the hull needs to minimize the drag forces of the water. These two characteristics depend on the geometry of the hull, which in turn will also affect the AUV maneuverability and power consumption. Given that pressure resistance and drag reduction play such an important role in AUVs performance, the definition of the shape of the hull is almost an unavoidable step in the design process of these vehicles [30, 31].

Several shapes are used in order to create AUVs [32–34]. From a pressure point of view, an spherically shaped AUV would be suitable, however, this shape at high speeds can lead to instability due to hydrodynamic effects [35]. In the other hand, cylindrical hulls have a series of interesting characteristics such as good pressure resistance and low longitudial drag [36]. Additionally, due to its longitudinal shape, the cylinder is more compatible with the shape of the inner components of an AUV (instruments, batteries, etc). A drawbacks of this shape are the cavitation effect [35] and instability of the robot [37].

Evidently, simple geometrical shapes are only a first approach, since usually these are based on previous experiences and studies. More sophisticated methods exist, in which the determination of the better suited hull for a given application is treated as an optimization problem. Indeed, optimization techniques are also used in order to find shapes to minimize the drag and improve pressure resistance [38]. Some studies use multi-objective optimization approaches in order to find suitable shapes from a hydrodynamic point of view and taking into account design considerations such as component placement [39] and cost [40].

Given that the determination of the drag forces relies heavily on hydrodynamic considerations, many studies are also carried out in order to characterize the hydrodynamic properties of a robot [41, 42]. These studies usually involve the use of finite element methods using specialized software such as Fluent [43, 44]. Hydrodynamic studies prove to be more useful in the design stage of the robot. However, even when used as an evaluation tool, they provide important information for the AUV user.

3.1.2 Propulsive topologies

The degree of actuation, the amount of degrees of freedom actuated in the AUV, defines what kind of movements can be generated by the robot propulsive system (propulsive topology). Evidently, the more an underwater robot is actuated, the more complex tasks it will be able of perform.

At the design stage, the designer needs to imagine the type of tasks that the AUV will perform in order to determine the needed degree of actuation of the vehicle. This task, simple at first sight, is decisive not only for the final capabilities of the robot but also for its overall performance. Indeed, a poorly chosen propulsive configuration can lead to an AUV incapable of performing certain tasks or, at the least, to an under-performant vehicle.

To make sure an AUV is maneuverable enough to perform a great amount of tasks, an immediate approach would be actuate all six degrees of freedom of the Cartesian space, making it *fully actuated*. However, as stated in the introduction of this thesis, another desired characteristic of an AUV is a high degree of autonomy. An AUV actuated in all of its degrees of freedom will certainly need many actuators, which would make the vehicle power-hungry. This could cause a rapid consumption of the, necessarily limited, energetic resources of the robot.

In the process of creating an AUV, at some point the designer must face this dichotomy between actuation capabilities and autonomy (power consumption). This is a key point in the development of the robot, since the solution to this dilemma will be fundamental for the rest of the process.

Several kind of topologies can be found in the literature. This shows the importance of this design domain, as well as the lack of a general solution capable of satisfying most application cases. We can divide these robots in two groups: fully actuated and underactuated robots.

3.1.2.1 Fully actuated robots

This type of underwater robots have six or more actuators actuating the six degrees of freedom of the robot. This property allows these robots to perform

a great variety of tasks, since with their propulsive resources they can move in the space with complete freedom (or holonomy). This freedom comes, however, at the cost of consuming great amounts of energy.

Despite their outstanding maneuverability, there are relatively few fully actuated AUV. This could be because of their power consumption or due to the fact that their potential agility is not yet matched with reliable control and navigation systems. Another reason for the reduced number of fully actuated AUVs can be the lack of appropriate missions. Indeed, for a great number of tasks nowadays there is not a valid reason to actively control all of the degrees of freedom of an AUV. This is due to the fact that AUV operations are mostly meant to be planar and at a certain depth, without many changes in orientation.

The importance of this type of vehicles; however, can not be neglected. A testimony of this, are the research efforts made in order to correctly control this AUV category. Some of these works mostly take a general approach at the control of these vehicles, aiming their developments to a conceptual level and not focusing on the robot particularities nor on a direct application [45–47], while others tackle the control of an specific vehicle [3].

Industrial AUV tend to focus on survey-style tasks [48–50], in which maneuverability is not as important as autonomy and velocity. In such an scenario, fully actuated AUVs are not yet sufficiently attractive. However, for academic proposes, fully actuated AUVs are very appealing. Indeed, due to their maneuverability, these vehicles can be used as a test bench in order to develop new control and localization methods [51–53]. Evidently, given the maneuverability of these robots and thanks to new advances in control, these underwater vehicles are receiving more attention from companies willing to invest in this kind of technology [54].

3.1.2.2 Under actuated robots

The second type of robots reviewed in this section are called *underactuated*. The main characteristic of thee AUV is that they are not able to fully control all of their six degrees of freedom.

The amount of underactuated AUVs found in the literature is greater than the amount of their fully actuated counterparts. This could be due to the reduced power consumption (less actuators means less consumed energy) or to the fact that most of the tasks for these robots only need the control over a few degrees of freedom.

Many examples of underactuated AUVs are available in the literature. Historically, the most common type were the torpedo-shaped underwater robots [55]. These vehicles, designed to travel great distances in order to

perform survey-style missions, were the first AUVs to find their way out of research facilities and get adopted to commercial use [56, 57]. The range of applications of torpedo-shaped AUVs is quite large, they are used to collect data [58], to map the sea floor [59] and to inspect pipelines [60]. Motivated by the need of autonomy and to travel great distances, a type of torpedo-shaped AUV called *glider* has been developed in the last decades [61, 62]. The main advantages of these vehicles are the long range mission capabilities with low energy consumption.

After the first wave of torpedo-shaped AUVs and thanks to advancements in control, localization and path planing, AUVs applications have mutated to other areas of interest. Along with the change of mission, the structure of these AUVs have mutated to adapt to new missions. Nowadays, we can find (mostly underactuated) AUVs acting as diving companions [63], cleaning ship hulls [64] and reaching their application scope to intervention tasks [65]. Even if they seem complex, these tasks do not demand the control of all degrees of freedom of the robot.

Control over this type of vehicle is more challenging than over fully actuated robots. Indeed, for underactuated AUVs, not every trajectory is reachable, due to their propulsive limitations. Great efforts have been made in order to overcome this drawback, in the literature we can find advanced and creative control techniques [66–69] trying to solve this problem.

3.1.3 Propulsion means

Equally important as the propulsive technology, is the way in which the propulsive forces are created. In this section, different propulsive means will be discussed, ranging from classic technologies, such as control and control surface, to the innovative biomimetic approach.

3.1.3.1 Fixed propulsion

The first type of propulsion technology analyzed in this section is the fixed one. These actuators are able to generate the propulsive force along a fixed direction with regard to the robot body. Usually, in order to properly control the degrees of freedom required by a mission, many fixed actuators are needed.

The number, orientation and position of these actuators on the hull of the AUV depend on the type of mission performed by the robot. In virtue of this, no general rules exist to define said design parameters. This step of the design process relies heavily on the previous experience of the designer.

Several ways of creating a fixed propulsive force exist [70]. The two most common being thrusters and water-jet actuators.

Thrusters are nowadays the most used technology, their relatively low cost and the fact that they have been developed for decades, give these actuators great advantages in terms of technological maturity. Given the popularity of these actuators, we can find them mounted o AUVs of very diverse types [71–73].

Also due to their popularity, several mathematical models explaining the behavior of these devices exist [74–76], which encourages their use even more. A sign of popularity of these actuators are the diverse studies we can find in the literature. These go from researching the hydrodynamic interaction between the thruster and the hull [77] to the miniaturization of this technology to be used in small AUVs [78].

The second type of fixed propulsive technology analyzed here is the jet-based propulsion. Typically, these actuators draw water from an inlet into a pump in charge of adding energy to the fluid. The water then passes through a nozzle, which in turn, transforms the pressure of the liquid in kinetic energy [70, 79], creating a net thrust force used to control the robot.

Jet propulsion offers several advantages over typical thrusters [61, 70], some of these are:

- Mechanical design (absence of rotating parts and transmission mechanisms)
- Cost
- Robustness
- Safety (less likely to cause harm at low speed than thrusters)

However, with regard to traditional thrusters, water-jet actuators are less efficient.

Even if this technology is not as widespread as thruster technology, a non negligible amount of research is devoted to model and improve the characteristics of this type of actuators [79–82]. This will probably lead into efficiency improvements and eventually a bigger adoption of this technology.

3.1.3.2 Vectorial propulsion

Vectorial propulsion is a technology capable of creating an orientable propulsive force. A vectorial force actuators is then a device in which we have to control no only the generated thrust but also the orientation of said force.

Vectorial propulsion offers interesting possibilities. The first one is the generation of thrust over directions that can change during the AUV operation. Using the reorientation we have access to more degrees of freedom per thruster than a classical fixed propulsive arrangement. If used correctly, vectorial propulsion can provide an underactuated AUV similar capabilities than a fully actuated one. This means giving the possibility to an underactuated vehicle to follow trajectories that otherwise would have been unreachable [83].

Overall AUV weight reduction is a second advantage of this technology. Indeed, given that one of these actuators can be in charge or controlling different degrees of freedom, a reduced number of devices are needed to control the movement of the AUV. A reduced number of actuators results in a reduced AUV weight.

Vectorial propulsion is, however, not exempt of drawbacks and limitations. From a fundamental point of view, vectorial propulsion generate new challenges in control and trajectory planning of AUVs, such as the increase of nonlinearities in the propulsive model [84] and coupling between its actuated degrees of freedom [85]. The versatility of these actuators, when not properly assessed, can be detrimental to the controllability of underwater robots.

From a practical point of view, the biggest limitation is the way of creating the reorientation. Several approaches exist, using different technologies such as water-jets [86, 87], creative mechanism [88], servomotors assemblages [89] and magnetic couplings [90]. However, most of the current solutions are not yet able to match the simplicity of the fixed propulsion approach.

As for water-jet propulsion, it does not mean that development of these actuators has stalled. Improvements and studies about their design and performance are conducted [91], making this a promising technology for the years to come.

3.1.3.3 Control surfaces

Historically, the first applications of AUV involved performing survey-style tasks. The classic AUV configuration for these missions was a torpedo-shaped hull, a propeller in the rear of the structure and control surfaces [92–94].

In this type of propulsion, the propeller and control surfaces act together in order to propel and steer the robot. The propeller, which is a device consisting of several rotating blades, accelerates the water creating a propulsive force. The control surfaces are in charge of controlling the thrust generated by the propeller, they are airfoil shaped devices capable of generating a lift when positioned at an angle with regard to the flow [95]. The lift depend on the shape of the control surface, its angle and evidently the velocity of the water. This last element is the link between the control surface and

the propeller: in order to generate lift, the robot needs to have a non zero velocity.

The propeller and control surface solution was used on ships before their application to AUVs [70], which means that the technology was mature enough when it was implemented on underwater robots. This made the application of this technology in survey-type robots easier, given their efficiency and proven efficacy in long range and high speed missions.

Given the success and popularity of survey style AUVs, the propeller and control configuration was studied extensively [96–98]. Aspects as blade shape optimization [99], interaction between propeller and hull [100] as well as the control surface optimization [101] were thoroughly analyzed.

When AUVs missions started to include middle and short range operations, the limitations of this technology arose. Indeed, given that the steering of the vehicle is only possible when the velocity of the water (with regard with the robot) was high enough, operations with low velocities became inefficient with this technology.

Nowadays, this technology is used almost exclusively in long range AUVs [102] and in underwater vehicles with both long and short range applications [95].

3.1.3.4 Biomimetics

The biomimetic approach proposes to base the AUV development in solutions found in the nature. This means adopting shapes, propulsion systems and locomotion control methods used by animals.

The basic idea behind this design approach is that after eons of evolution, nature has found efficient and reliable locomotion solutions. Biomimetic relies on the fact that the work of creating, testing and evolving, in this case, propulsion systems, has been done already in a natural way [103].

In nature we can find fishes and other aquatic beings which are highly maneuverable, power-efficient and capable of performing demanding tasks such as station-keeping in presence of perturbations [104]. Clearly, the understanding of these systems can be used to inspire artificial human-made underwater systems.

Biomimetic design, as a method for underwater vehicles, covers robot shape, propulsion system and control (locomotion behavior control). In virtue of this, biomimetics can be included in any other section of this chapter. However, given that the final goal for biomimetic design in AUVs is to emulate animal locomotion methods, it is included in this section.

Several examples of biomimetic AUVs exist in the literature, there are robots copying the locomotion methods of squids [105], eels [106], turtles

[107], dolphins [108] and fish [109]. Even if the animals inspiring the designs are diverse, most of the biomimetic robots nowadays use (in some way or another) fin propulsion as a base [110, 111].

This technology has generated great interest in the scientific community, and as such, extensive analysis about fins hydrodynamics [112] and real prototypes have been carried out [113, 114]. Another proof of the popularity of biomimetics is the diverse methods to achieve animal-like movement. Indeed, in the literature we can find complex and creative mechanisms [106, 108] as well as solutions using smart materials [115]. These materials have special characteristics that, when applied correctly, allow biomimetic AUVs to perform complex movements. Evidently, biomimetic locomotion also supposes the use of adapted control techniques in order to master the movement of the robot, we can find several examples of these control methods in the literature [116–118].

3.1.4 Control techniques

Throughout the history of AUVs, the increasingly complex tasks have pushed the limits of the technology of these vehicles. Something similar has happened with the control methods. Indeed, the mission of AUVs have changed dramatically in the last decades, going from survey-style high speed tasks to low-speed high precision ones. Certainly, in order to perform these new tasks, designers had to make use of more efficient technology such as more powerful batteries, more responsive propulsion methods and high-end sensors. However, equally (or even more) important than technology are the control methods used to perform said missions.

In the literature we can find several types of controllers used on AUVs. From PID [119] and H_∞ [120] approaches to sliding mode [121] and neural network based techniques [122]. Most of the found techniques have been extensively tried in simulations but only few have been transferred and successfully tested on real devices [123]. This could be due to the lack of adapted hardware.

Whether the missions are survey-style or with a focus on hovering and maneuverability, most of the control techniques used for them are some variation of PID, sliding mode or adaptive control methods [123]. Given that these three techniques are the most used, the focus on this section will be put on them. These techniques will also be categorized in two groups: linear and non-linear.

3.1.4.1 Linear

Despite its simplicity, PID [124] (proportional, integral and derivative) control remains even nowadays one of the most implemented methods in underwater robotics [123].

Several factors favor the adoption of this technique. The main one is its simplicity in terms of implementation and configuration. Indeed, even if the gains K_p, K_i and K_d can be determined based on information of the system, they can also be configured manually with good performances [125].

Another non negligible reason for using this technique is the available information and examples in the literature. Indeed, we can find examples of PID implementation in AUVs from 1994 such as the N-DRE robot [126] performing survey motions and in articles from 22 years later [127]. This shows the versatility of the technique and, at the same time, the need for better and more advanced approaches in order to avoid the limitations of PID. Even if PID is easy to implement and it has shown good performances in AUVs, it is still a linear method trying to control a highly nonlinear system. Performances are expected to decay when used in variable speed scenarios (vehicle dynamics are nonlinear with respect to its velocity [128]).

PID is usually used for linear time invariant single input single output (SISO)systems. In the case of AUVs, the degrees of freedom to control are decoupled in order to apply the PID controller [119]. The correct choice of the gains have received much attention as well. We can find implementations of PID with gain selection based on the model of the AUV [129] or even using machine learning approaches such as neural networks [130] and genetic algorithms [131]. The selection of the best gains, given a robot and mission, helps to improve AUV performance, however, once the PID technique has reached its limits, it is better to use a more advanced nonlinear method.

3.1.4.2 Nonlinear

Sliding mode control and adaptive control are two popular nonlinear methods used in AUVs. In the literature we can find several applications of these controllers with good performances [132, 133].

Sliding mode is a method in which, with regard to PID, offers robustness and the possibility to better deal with disturbances and non modeled dynamics [123]. Sliding mode gets the desired systems response even in the presence of modeling error and disturbances by *sliding* through them in order to track the desired trajectory. Indeed, in sliding mode, the desired trajectory is known as the sliding surface [128]. The simplest way of implanting this control technique is to have one action to apply when the system response is

above the sliding surface and a different one when it is below it. Sliding mode has a good response and allows to rapidly converge to the desired trajectory, however it has a tendency to create chattering in the controller action. This effect can be reduced of removed with appropriate techniques [128].

Another very popular approach is adaptive control [123]. It is a type of controller that will adapt taking into account the AUV parameters, even if they are uncertain. Adaptive methods are clearly adapted to AUVs, given the nonlinear dynamics of these vehicles and their hydrodynamics, which are not always properly identified and that may vary with the velocity.

This type of controller are very well established, in the literature we can find several examples of its application to AUVs [134–136]. These controllers usually include Model Reference Adaptive Control and Model Identification Adaptive Control [137]. The first type uses a reference model which defines the desired performance of the system. the controller sends a command signal, containing variable parameters, to the system. The parameters are changed in order to obtain asymptotic convergence of the real system towards the reference one. The second approach, also called self-tuning control, performs on-line system identification. The parameters of the systems are first estimated and then, based on the parameter estimation, the controller is designed.

3.2 Design methods

To finish the state of the art, different AUV design approaches will be discussed. The AUV being a complex mechanism with diverse subsystems from very different domains (hydrodynamics, control, localization, etc.), needs to be designed attending several constraints and specifications. Some of these are counter-productive between them. For instance, the need of a large interior volume to store the payload can result in an AUV with a mediocre hydrodynamic behavior due to drag forces.

Naturally, given the need to reconcile the diverse aspects of AUV design, different techniques and protocols have been used to obtain robots capable of attending all needed characteristics.

The first type of design method used to build AUVs has been sheer experience and intuition. Engineers with previous experience in design of robots and aquatic devices took (and take still nowadays) the daunting task of adjusting the several needed AUV characteristics. For some AUVs, methods, ideas and techniques derived from ship design were borrowed.

The great amount of heterogeneous design parameters and the difficulty of creating efficient AUVs based solely on experience, made optimization

based design appealing.

Early optimization methods involved analyzing the risk of failure of the AUVs components and using a computerized evaluation tool to evaluate the need of redundant devices [138]. With advancements in computing power, CFD (Computational Fluid Mechanics) techniques were used to optimize the shape of the AUV, in order to obtain vehicles with low drag and capable of operating at high pressures [43]. Also for reducing drag, other optimization approaches have been taken, such as pattern search [139] and genetic algorithms [140, 141].

In order to take into account the different design parameters intervening in the AUV, multidisciplinary design optimization [142] techniques are also used. These methods use individual analysis for each subsystem and then aggregate them using a system-level coordination procedure [143]. Similarly, multi-objective optimization techniques using genetic algorithms are used with the goal of optimize design parameters in order to obtain better effectiveness, and reduce cost and risk [144, 145].

3.3 Conclusion

From this review of underwater robot design, we can see how vast is the spectrum of possibilities in order to optimize the AUVs sub-system. Indeed, given the relatively young age of AUV technology, most of these design are starting to gain maturity, offering exciting possibilities.

The great amount of options, however, implies that we have to carefully choose the set of parameters to use in the optimization of the propulsive capabilities of AUVs. Each chosen set of parameters could unveil possibilities unlikely to find otherwise. For instance, the study of biomimetic design takes us to the study of the fundamentals of sea creatures locomotion, something that we probably would not do if we study the classical AUV approaches.

In this work, the design of the following sub-systems have been studied:

- Fixed and vectorial thruster technology: The former is an extensively used and established technology. The latter is a promising technology, capable of extending the propulsive capabilities of the AUV.

- Propulsive topology: Given its deep influence in the AUV capabilities to perform complex tasks. Propulsive topology includes the type of actuation (fully or underactuated), the number of thrusters and their arrangement on the robot.

- Control technique: For the sake of feasibility, a single nonlinear control technique will be used and its parameters are optimized. This is motivated by its adaptation to dynamic tasks, in which linear controllers may have a mediocre performance. Our study consist in adapting this control to different propulsion systems.

Our strategy is to use an evolutionary algorithm to optimize the design of these sub-systems. This choice is based on the good results of this technique and its capabilities of optimizing systems with heterogeneous parameters. For evaluation of robot designs, the choice have been made to rely on dynamic simulation. This can be justified by the complexity to asses the robot performance with regard to the propulsion and control systems. The simulation will show the behavior of the robot on its task without considering its theoretical capabilities, which are hardly assessable.

Chapter 4

Modeling and simulation

4.1 Autonomous underwater vehicles modeling

Our study of underwater robots will begin with their modeling, which is the fundamental stage in order to understand, control and optimize any system.

Underwater robots need several models to be completely described, the main ones represent the behavior of these vehicles from a kinematic and dynamic point of view. Additionally, there are models describing the different sub-systems of the robot, such as the propulsive topology or technology.

Said models will be described in this chapter. The first ones will be taken from [146], including the calculation of hydrodynamic coefficients, since their accuracy is enough for our purposes.

Also the development of a dynamic simulation, the tool we will use to study the behavior of the underwater robot as a whole, will be discussed.

4.1.1 Kinematics

Here, the AUV kinematic equations are presented. In order to develop them, we make the following assumptions:

- The AUV is submerged far from the bottom, walls and water surface in a homogeneous fluid
- Underwater currents are neglected
- The AUV is a rigid body of constant mass

Two orthogonal coordinate systems are used: R_0 $(O_0, \mathbf{x}_0\mathbf{y}_0, \mathbf{z}_0)$ is the earth-fixed frame and R_b $(O_b, \mathbf{x}_b, \mathbf{y}_b, \mathbf{z}_b)$ is AUV body-fixed. In Fig 4.1, body-fixed, and earth-fixed coordinate systems are shown together with a diagram of the AUV.

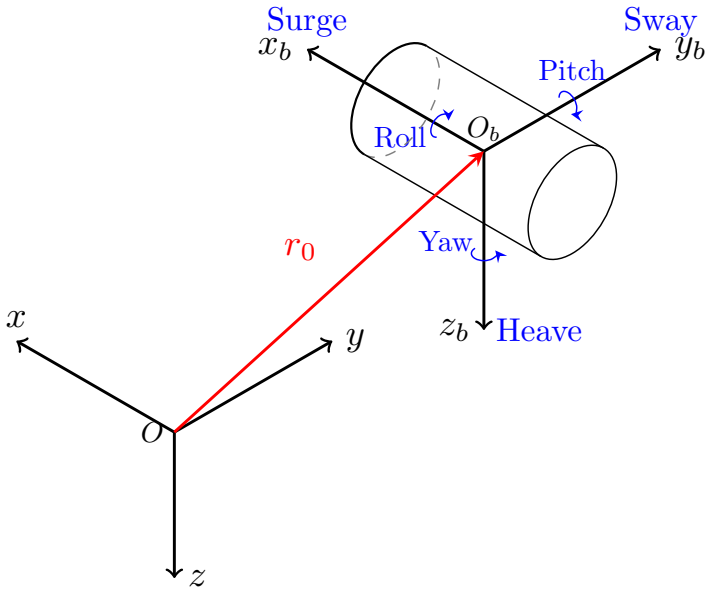

Figure 4.1: Earh-fixed and body-fixed frames.

The vectors describing the motion of the AUV in 6 DOF are:

$$\boldsymbol{\eta} = \begin{bmatrix} \boldsymbol{\eta}_1 \\ \boldsymbol{\eta}_2 \end{bmatrix} \quad \boldsymbol{\eta}_1 = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \quad \boldsymbol{\eta}_2 = \begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix} \tag{4.1}$$

$$\boldsymbol{\nu} = \begin{bmatrix} \boldsymbol{\nu}_1 \\ \boldsymbol{\nu}_2 \end{bmatrix} \quad \boldsymbol{\nu}_1 = \begin{bmatrix} u \\ v \\ w \end{bmatrix} \quad \boldsymbol{\nu}_2 = \begin{bmatrix} p \\ q \\ r \end{bmatrix}$$

$$\boldsymbol{\tau} = \begin{bmatrix} \boldsymbol{\tau}_1 \\ \boldsymbol{\tau}_2 \end{bmatrix} \quad \boldsymbol{\tau}_1 = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \quad \boldsymbol{\tau}_2 = \begin{bmatrix} K \\ M \\ N \end{bmatrix}$$

Here $\boldsymbol{\eta}$ is the vector of position and orientation in R_0. The orientation $\boldsymbol{\eta}_2$ is defined using an Euler ZYX (ψ, θ, ϕ) convention as described in [146]. $\boldsymbol{\nu}$ is the linear and angular absolute velocity vector in R_b. $\boldsymbol{\tau}$ is the external forces and moments vector in the body-fixed frame, which accounts for propulsion

forces applied on the AUV. Precisely, this vector will allow to model thrust forces from different propulsion architectures.

To change the AUV velocity vector from one to another coordinate system we use a velocity transformation matrix as given in [146]:

$$\mathbf{J}(\boldsymbol{\eta}_2) = \begin{bmatrix} \mathbf{J}_1(\boldsymbol{\eta}_2) & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{J}_2(\boldsymbol{\eta}_2) \end{bmatrix} \tag{4.2}$$

Where,

$$\mathbf{J}_1(\boldsymbol{\eta}_2) = \begin{bmatrix} c\psi c\theta & -s\psi c\phi + c\psi s\theta s\phi & s\psi s\phi + c\psi c\phi s\theta \\ s\psi c\theta & c\psi c\phi + s\theta s\theta s\psi & -c\psi s\phi + s\theta s\psi c\phi \\ -s\theta & c\theta s\phi & c\theta c\phi \end{bmatrix} \tag{4.3}$$

and

$$\mathbf{J}_2(\boldsymbol{\eta}_2) = \begin{bmatrix} 1 & s\phi t\theta & c\phi t\theta \\ 0 & c\phi & -s\phi \\ 0 & \frac{s\phi}{c\theta} & \frac{c\phi}{c\theta} \end{bmatrix}$$

Using this transformation matrix, we can obtain the AUV absolute velocity vector expressed in R_0 from its expression in R_b [146]:

$$\dot{\boldsymbol{\eta}} = \left.\frac{d\boldsymbol{\eta}}{dt}\right|_{R_0} = \mathbf{J}(\boldsymbol{\eta}_2)\,\boldsymbol{\nu} \tag{4.4}$$

4.1.2 Dynamics

The nonlinear dynamic equations of an underwater robot, given in the R_b frame, can be formulated as [3]:

$$\mathbf{M}\,\boldsymbol{\nu} + \mathbf{C}\,\boldsymbol{\nu} + \mathbf{D}\,\boldsymbol{\nu} + \mathbf{G} = \boldsymbol{\tau} \tag{4.5}$$

where $\mathbf{M} \in \mathbb{R}^{6\times6}$, $\mathbf{C} \in \mathbb{R}^{6\times6}$ and $\mathbf{D} \in \mathbb{R}^{6\times6}$ are the matrices of mass, Coriolis and centripetal terms, and damping respectively (including added mass terms). $\mathbf{G}$ is the vector of gravitational forces and moments. Lastly, $\boldsymbol{\tau}$ is the wrench of external forces and moments.

The wrench $\boldsymbol{\tau}$, accounts for the propulsive forces generated by the thrusters. It is calculated as follows:

$$\boldsymbol{\tau} = \mathbf{B}\mathbf{u}_p \tag{4.6}$$

where $\mathbf{B}$ is the thrust control matrix (TCM), which depends on the propulsive architecture (number, position and orientation) of the actuators. $\mathbf{u}_p$ is the vector of the actuators forces (i.e., the control input of thrusters).

These matrices are constructed using rigid body and hydrodynamic considerations. The rigid body dynamic part is based on the fact that the underwater vehicle is nothing more than a solid, so the well known rigid body dynamics applies. Given that these kind of vehicles operate solely under water, we need to add a hydrodynamic component or *layer* to the dynamic model in order to completely describe the robot. These hydrodynamic considerations are explained next.

4.1.3 Hydrodynamics

The hydrodynamic matrices described in this section, account each one for a specific dynamic effect of the water over the robot. Said effects are *added mass*, *drag forces* and *gravitational forces and moments*, all of which are described as follows.

4.1.3.1 Added mass

During its motion, underwater vehicles displace the water in its surroundings. Since the AUV applies a force and a moment on the neighboring fluid, reaction forces are applied to the AUV [146]. We take into account this reactive force with the concept of added mass. Added mass is not a finite amount of fluid to add to the AUV mass but a function of the robot geometry [3] and motion velocities.

The global mass matrix ($\mathbf{M}$) in the general nonlinear dynamic model (Eq. 4.5) accounts not only for the mass and inertia of the underwater robot but also for the added mass contribution, namely:

$$\mathbf{M} = \mathbf{M}_b + \mathbf{M}_a \tag{4.7}$$

where $\mathbf{M}_b \in \mathbb{R}^{6\times6}$ is the mass matrix of the robot rigid body:

$$\mathbf{M}_b = \begin{bmatrix} m\mathbf{I}_{3\times3} & -m\mathbf{S}({}^bP_G) \\ m\mathbf{S}({}^bP_G) & \mathcal{J} \end{bmatrix}$$

with

$$\begin{array}{ccc} m & : & \text{mass of the AUV} \\ \mathbf{I}_{3\times3} & : & 3 \times 3 \text{ Identity matrix} \\ {}^{b}P_{G} & : & \text{position vector of AUV CoG in } R_b \\ \jmath & : & \text{Inertia matrix of AUV in } R_b \\ \mathbf{S}(X) & : & \text{cross pre-product matrix of X} \end{array}$$

$$\mathbf{S}([pqr]^T) = \begin{bmatrix} 0 & -r & q \\ r & 0 & -p \\ -q & p & 0 \end{bmatrix}$$

and $\mathbf{M}_a \in \mathbb{R}^{6\times6}$ is the added mass contribution, which can be defined in an abbreviated form [146]:

$$\mathbf{M}_a = \begin{bmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{bmatrix}$$

Where $\mathbf{A}_{11}$,$\mathbf{A}_{12}$,$\mathbf{A}_{21}$ and $\mathbf{A}_{22} \in \mathbb{R}^{3\times3}$ represent the added mass terms. This matrix can be simplified for certain conditions and AUV structures. For a robot completely submerged in the water, with a three plane of symmetry structure and working at low speed, we can neglect the contribution of the non-diagonal terms in $\mathbf{M}_a$.

Added mass also contributes with Coriolis and centripetal terms. In the general non linear dynamic equation (eq. 4.5), the Coriolis and centripetal terms are included in the $\mathbf{C}$ matrix:

$$\mathbf{C} = \mathbf{C}_b + \mathbf{C}_a \tag{4.8}$$

where $\mathbf{C}_b \in \mathbb{R}^{6\times6}$ is the Coriolis and centripetal matrix of the rigid body:

$$\mathbf{C}_b = \begin{bmatrix} m\mathbf{S}(\boldsymbol{\nu}_2) & -m\mathbf{S}(\boldsymbol{\nu}_2)\mathbf{S}({}^{b}P_{G}) \\ m\mathbf{S}({}^{b}P_{G})\mathbf{S}(\boldsymbol{\nu}_2) & -\mathbf{S}(\jmath\boldsymbol{\nu}_2) \end{bmatrix}$$

and $\mathbf{C}_a \in \mathbb{R}^{6\times6}$ is the contribution due to the added mass:

$$\mathbf{C}_a = \begin{bmatrix} \mathbf{0}_{3\times3} & -\mathbf{S}(A_{11}\boldsymbol{\nu}_1 + A_{12}\boldsymbol{\nu}_2) \\ -\mathbf{S}(A_{11}\boldsymbol{\nu}_1 + A_{12}\boldsymbol{\nu}_2) & -\mathbf{S}(A_{21}\boldsymbol{\nu}_1 + A_{22}\boldsymbol{\nu}_2) \end{bmatrix}$$

4.1.3.2 Damping forces

Matrix $\mathbf{D} \in \mathbb{R}^{6\times6}$ in the dynamic equation accounts for dissipative effects (drag forces due to shape and friction). As for the added mass matrix, we can make simplifications for three plane of symmetry structures, which leads to neglect the contribution of the non-diagonal terms. Giving the nature of dissipative effects, this matrix is positive.

$$\mathbf{D}(\boldsymbol{\nu}) > 0 \tag{4.9}$$

4.1.3.3 Gravitational forces

These are gravitational and buoyancy forces, which act respectively through the center of gravity ${}^{b}\mathbf{P}_{G}$ and the center of buoyancy ${}^{b}\mathbf{P}_{B}$. Restoring force and moment can be calculated in R_b as follows:

$$G = -\begin{bmatrix} {}^{b}\mathbf{f}_{G}(\boldsymbol{\eta}_2) + {}^{b}\mathbf{f}_{B}(\boldsymbol{\eta}_2) \\ {}^{b}\mathbf{P}_{G} \times {}^{b}\mathbf{f}_{G}(\boldsymbol{\eta}_2) + {}^{b}\mathbf{P}_{B} \times {}^{b}\mathbf{f}_{B}(\boldsymbol{\eta}_2) \end{bmatrix} \tag{4.10}$$

with

$$\begin{cases} {}^{b}\mathbf{f}_{G}(\boldsymbol{\eta}_2) = \mathbf{J}_1^{-1}(\boldsymbol{\eta}_2) \begin{bmatrix} 0 \\ 0 \\ W \end{bmatrix} & \text{the weight of the robot} \quad (4.11) \\ {}^{b}\mathbf{f}_{B}(\boldsymbol{\eta}_2) = -\mathbf{J}_1^{-1}(\boldsymbol{\eta}_2) \begin{bmatrix} 0 \\ 0 \\ B \end{bmatrix} & \text{the buoyancy force} \quad (4.12) \end{cases}$$

$W = m\,g$	:	is the robot weight
$B = \rho\,g\,\nabla$	:	is the buoyancy force
$g = 9.81\ m/s^2$	:	is earth gravity acceleration
$\rho = 998\ kg/m^3$	:	water density
∇	:	displaced volume of water

4.2 Propulsion modeling

The next step in the modeling of the underwater robot is to describe its propulsion. We can define two modeling levels for propulsion, the first and more global one, is the one that mathematically describe the way in which the thrusters are positioned and oriented in the robot. The second level,

more specific, determines the technology used in each thruster. In this section we will discuss first the *Thrust configuration* model and then we will present models for two types of propulsion technologies: *fixed* and *vectorial* propulsion.

4.2.1 Thrust configuration

From Eq. 4.6 we see that $\boldsymbol{\tau}$, is calculated by multiplying $\mathbf{B}$ and $\mathbf{u}_p$.

$\mathbf{u}_p$ is a vector that contains the forces generated by the thrusters. For a robot of n thrusters, this vector would be:

$$\mathbf{u}_p = \begin{bmatrix} u_1 \\ \vdots \\ u_i \\ \vdots \\ u_n \end{bmatrix} \tag{4.13}$$

where u_i is the force generated by the i^{th} thruster.

The $\mathbf{B}$ matrix represents the organization of the thrusters on the underwater robot. It has a column per thruster and 6 lines (one per degree of freedom in $\boldsymbol{\tau}$). In each column, the three top elements represent the force created by the thruster, whereas the bottom three elements represent the created moment, which is a function of the thruster position.

Given that the lines of this matrix represent the contribution of all thrusters to create $\boldsymbol{\tau}$, for different propulsive arrangements (propulsive topologies) we will have different $\mathbf{B}$ matrices. In figure 4.2, we see the RSM robot with a fixed thruster topology and its corresponding $\mathbf{B}$ matrix. In this figure (4.2), the values of $\vec{P}$ account for the position of thrusters 1 to 4 in the R_B frame.

$$\mathbf{B} = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & -P_{3y} & -P_{4y} \\ P_{1z} & P_{2z} & P_{3x} & P_{4x} \\ -P_{1y} & -P_{2y} & 0 & 0 \end{bmatrix}$$

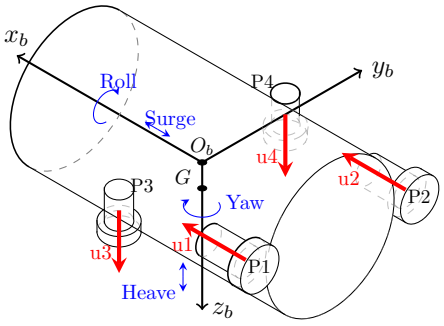

Figure 4.2: RSM fixed propulsive topology.

Similarly, Fig. 4.3 shows the RSM robot with a vectorial propulsive topology. This type of propulsive topology uses one thruster capable of changing its direction, thus creating a variable force vector. In figure 4.3, $\vec{P}$ also accounts for the position of the only thruster of the topology, θ_1 and ψ_1 are the two reconfiguration angles and $s(\cdot)$ and $c(\cdot)$ designate *sin* and *cos* respectively. The reconfiguration angles are the successive angles (using an Euler ZYX convention) defining the orientation of the thruster axis. In the first three lines of the thrust configuration matrix we see the components on R_b of the thrust vector, whereas in the last three lines we see the moment these components create. We define $\mathbf{u}_v$ as the Cartesian space vector of the thrust.

Defining the three components of $\mathbf{u}_v$ on R_b as u_x, u_y and u_z, we can recalculate the configuration of the thrust vector as follows:

$$\mathbf{u}_v = [u_x \;\; u_y \;\; u_z]$$

$$u_p = ||\mathbf{u}_v|| = \sqrt{u_x^2 + u_y^2 + u_z^2}$$

$$\theta_1 = \arcsin(\frac{u_z}{u_p}) \qquad\qquad \psi_1 = \arctan(\frac{u_y}{u_x})$$

$$\mathbf{B} = \begin{bmatrix} c\theta_1 c\psi_1 \\ c\theta_1 s\psi_1 \\ -s\theta_1 \\ -s\theta_1\, P_{1y} - c\theta_1 s\psi_1\, P_{1z} \\ c\theta_1 c\psi_1\, P_{1z} + s\theta_1\, P_{1x} \\ c\theta_1 s\psi_1\, P_{1x} - c\theta_1 c\psi_1\, P_{1y} \end{bmatrix}$$

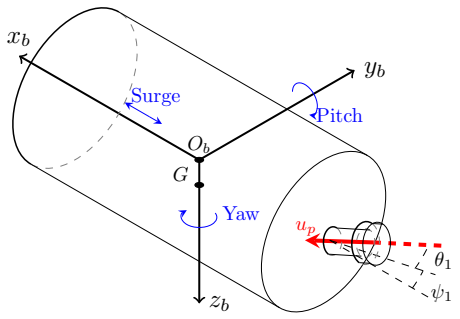

Figure 4.3: RSM vectorial propulsive topology.

Finally, a last example is shown in figure 4.4, in which we see the Odin robot [3] and its **B** matrix. The Odin robot has eight thrusters, which means the thrust configuration matrix has eight columns as well. In the figure $s = sin(\pi/4)$, $l_1 = 0.381\,m$ and $l_2 = 0.508\,m$.

$$\mathbf{B} = \begin{bmatrix} s & -s & -s & s & 0 & 0 & 0 & 0 \\ s & -s & -s & -s & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & -1 & -1 & -1 \\ 0 & 0 & 0 & 0 & l_1 s & l_1 s & -l_1 s & -l_1 s \\ 0 & 0 & 0 & 0 & l_1 s & -l_1 s & -l_1 s & l_1 s \\ -l_2 & -l_2 & -l_2 & -l_2 & 0 & 0 & 0 & 0 \end{bmatrix}$$

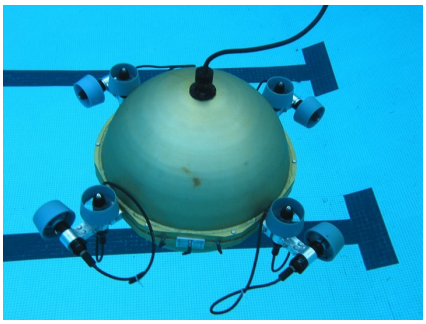

Figure 4.4: Odin propulsive topology.

4.2.2 Fixed propulsion modeling

A fixed thruster is a type of thruster that, once it has been installed on the robot structure, it can not change its configuration (orientation). These type of thrusters are the most widely used in underwater robotics, which means that this technology is in its maturity. As such, several models exist to describe the behavior of these actuators, here we will use the most broadly known [147, 148].

To develop the model of this type of thruster, we will take as a reference the actuators used in the RSM robot, the *Seabotix* BTD150 [1] (Fig. 4.5), a ducted oil-filled direct-drive DC motor thruster.

Figure 4.5: Seabotix BTD 150 thruster.

4.2.2.1 Steady-state model

The hydrodynamic thruster model can be written as follows [147, 148]:

[1] http://www.seabotix.com/products/auv_thrusters.htm

$$u = K_{T0}\,\dot{\omega} + K_T\,\omega\,|\omega| \tag{4.14}$$

where u is the thrust force and ω is the rotational velocity of the propeller shaft. K_{T0} and K_T are lump parameters of various constants that need to be determined.

The relation between the voltage V and he thrust force u is given by the following motor electromechanical model:

$$\begin{cases} V = & R\,I + L\,\dot{I} + \kappa\,\omega \\ u = & K_M\,I \end{cases} \tag{4.15}$$

Where R is the wiring resistance, L the inductance, I the current, κ is the back-EMF constant of the motor and K_M is the specific torque constant related to thrust force.

Considering only the steady-state components of the model ($\dot{I} = 0$ and $\dot{\omega} = 0$), Eq. 4.14 and Eq. 4.15 can be reformulated as:

$$\begin{cases} V = & R\,I + \kappa\,\omega \\ u = & K_T\,\omega|\omega| \end{cases} \tag{4.16}$$

Combining these equations (4.15 and 4.16) we can derive the thruster steady-state model:

$$V = \frac{R}{K_M}u + \kappa\sqrt{\frac{u}{K_T}} \tag{4.17}$$

Eq. 4.17 will be used to calculate the open-loop control input of the thrusters, for a desired thrust u.

4.2.2.2 Dynamic model

The thrusters DC brushed motor can be mechanically modeled as follows:

$$J_m\,\dot{\omega} + \Gamma_f + \Gamma_L = \lambda\,I \tag{4.18}$$

where J_m is the motor shaft inertia, Γ_f is the resistive torque and Γ_L is the load torque. κ and λ, the torque constant, are considered equal if expressed in SI units.

The resistive torque can be calculated as:

$$\Gamma_f = k_{fv}\,\omega + k_{fs}\,\frac{\omega}{|\omega|} \tag{4.19}$$

with k_{fs} and k_{fv} the dry and viscous friction coefficients, respectively. The load torque can be calculated as [76]:

$$\Gamma_L = K_L\,\omega|\omega| \tag{4.20}$$

where K_L is a proportionality constant.

Using Eq. 4.15 and 4.18 we get an electro-mechanical model of the thruster. The state-space system equations are:

$$\begin{bmatrix} \dot{\omega} \\ \dot{I} \end{bmatrix} = \begin{bmatrix} -\frac{k_{fv}}{J_m} & \frac{\lambda}{J_m} \\ -\frac{\kappa}{L} & -\frac{R}{L} \end{bmatrix} \begin{bmatrix} \omega \\ I \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{L} \end{bmatrix} V + \begin{bmatrix} -K_L\omega|\omega| - k_{fs}\frac{\omega}{|\omega|} \\ 0 \end{bmatrix} \tag{4.21}$$

This last set of equations are important since they allow to calculate the propeller angular speed ω, and thus the generated dynamical thrust u for the dynamic simulation. To be able to apply the proposed models, a further step is needed, the thruster parameter identification.

4.2.2.3 Parameter identification

Electromechanical parameters of the brushed DC motor are identified by measuring motor voltage, current, and propeller angular speed. These measurements took place outside the water, in order to neglect the influence of the unknown resistive torque and to allow speed measurement. The identified parameters are as follows: $R = 2.2\Omega$, $L = 1.5\,mH$, $J_m = 0.000562kgm^2$, $\lambda = \kappa = 0.036Nm/A$, $k_{fv} = 0.00012Nms$, and $k_{fs} = 0.00135Nm$.

Hydromechanic parameters are identified using an ATI Nano 17 multiaxial submarine force sensor [149]. The test rig shown in Fig. 4.6 allows performing static or dynamic force measurements. The identified parameters are as follows: $K_M = 3.46N/A$, $K_T = 0.00015Ns^2$, and $K_L = 0.00135Nms^2$. Figure 4.7 shows that the model describes with accuracy the thruster real behavior (maximum error of 8.6% on the higher extremity). For higher thrust forces, the model seems to over-estimate the needed voltage value. This could be explained by perturbations in the water surrounding the thruster due to limited ($1300\,l$) basin dimensions.

4.2.3 Vectorial propulsion modeling

A vectorial thruster is a type of thruster capable of changing the direction of its thrust force. In order to model this type of actuator, we will base our

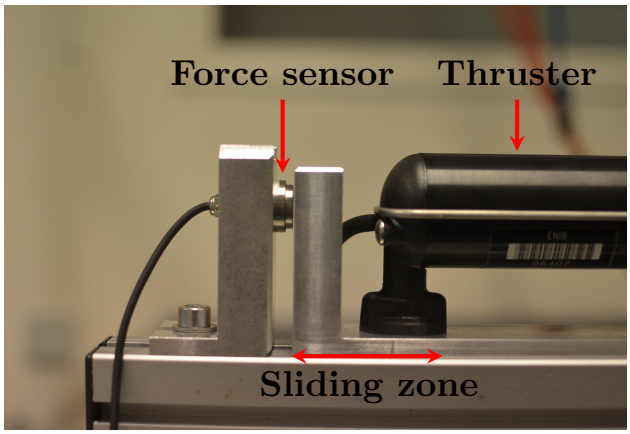

Figure 4.6: Force measuring test rig.

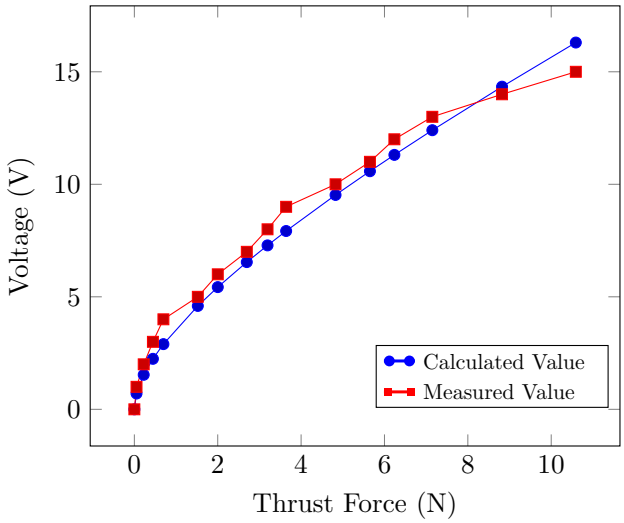

Figure 4.7: Thrust model validation.

equations in the Flat reconfigurable magnetic coupling thruster (F-RMCT) [150].

Figure 4.8 shows a diagram of the F-RMCT. This system consists of an electric motor attached to an axis. An axis attached to the propeller provides the propulsive force. The propeller axis is orientable with regard to the motor axis thanks to the axis support (fork), which is the link between the frame of the system and the propeller axis. The torque generated by the motor is transmitted to the propeller axis via a magnetic coupling.

The magnetic coupling is made by two identical rotors (Fig. 4.9). Each rotor consists of a circular polar part and four magnets with axial magne-

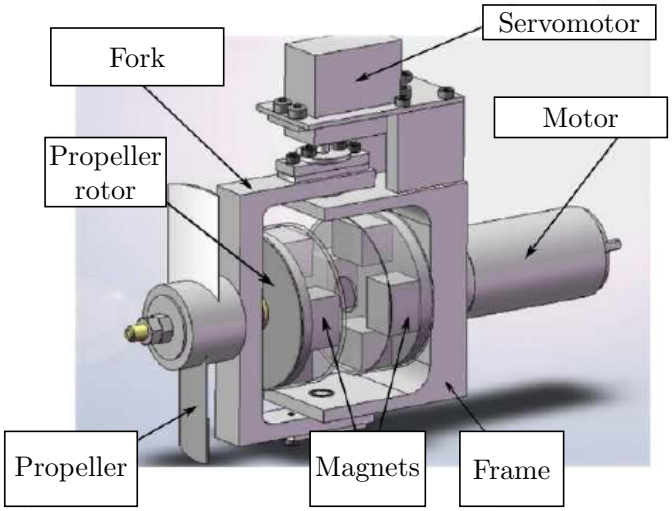

Figure 4.8: RMCT prototype.

tization. The magnets are glued to the polar part symmetrically and are placed with opposed magnetic orientations with regard to its neighbor in the rotor. Magnetic studies of F-RMCT has been undergone theoretically using finite elements method (FEM). This work has been undergone by TE2M[2] company.

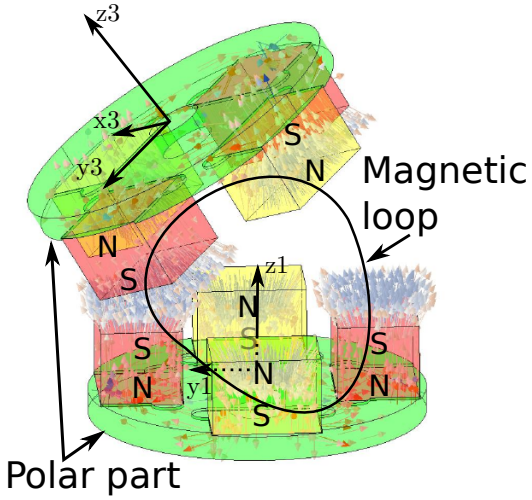

Figure 4.9: Magnetic coupling and forces (TE2M using Flux3D).

[2]Technique et Matériel Magnetique. Company based in Brest, France

Figure 4.10 shows the kinematic diagram of the F-RMCT. R_1,R_2 and R_3 are the frames attached to the motor shaft, the fork and the propeller shaft respectively. $\theta_m = (\vec{x}_0,\hat{\vec{x}}_1)$ is the motor rotor angle, the rotation angle of the x_1 axis attached to the motor with respect to the reference axis $\vec{x}_0$. $\theta_h = (\vec{y}_3,\hat{\vec{y}}_2)$ is the rotation of the propeller axis, that is, the rotation angle of the y_3 axis attached to the propeller with respect to the reference axis $\vec{y}_0 = \vec{y}_2$. α is the reconfiguration angle (angle between z_0 and z_3).

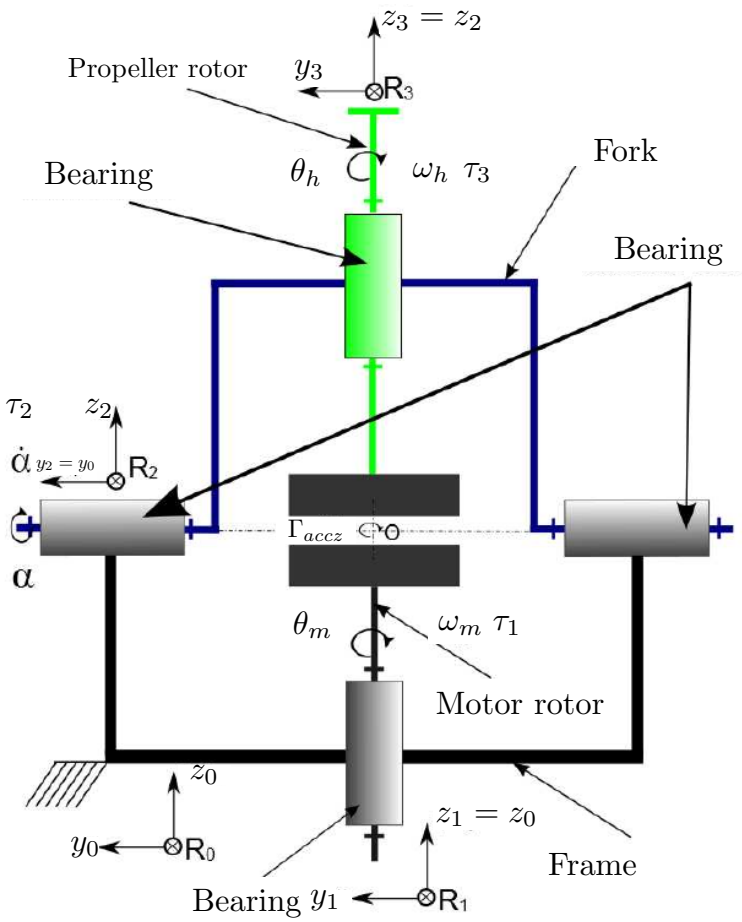

Figure 4.10: Flat RMCT kinematic diagram (shown in the neutral configuration, $\alpha = 0$).

4.2.3.1 Mechanical model

To model the mechanism, the recursive Newton-Euler algorithm will be used [124]. Thanks to this algorithm, we calculate all the kinematic and dynamic equations. τ_1, τ_2, τ_3 are the joint (external) torques between each solid (Fig.4.10).

The torque transmitted by the motor shaft to the magnetic coupling is:

$$\vec{\mathcal{M}}(\vec{F}_{mag}) \cdot \vec{z}_1 = \Gamma_{accz1}$$

As well for the transmitted torque to the propeller:

$$\vec{\mathcal{M}}(\vec{F}_{mag}) \cdot \vec{z}_3 = \Gamma_{accz3}$$

with $\vec{F}_{mag}$ the magnetic force generated by the magnets of the rotors.

There is a planar symmetry between the two rotors, since all the rotation axes pass through the center point O (Fig.4.10). This plan is defined by O, $\vec{y}_0$ and $\vec{x}_0^*$ (median between $\vec{x}_1$ and $\vec{x}_3$). Thanks to said symmetry we can establish:

$$\begin{aligned} \mathcal{M}(\vec{F}_{mag}) \cdot \vec{z}_1 &= \mathcal{M}(\vec{F}_{mag}) \cdot \vec{z}_3 \\ \Gamma_{accz1} &= \Gamma_{accz3} = \Gamma_{accz} \end{aligned}$$

Even though Γ_{accz} varies according to α, θ_m and θ_h, its projections on both axis $\vec{z}_1$ and $\vec{z}_3$ are equal. (Γ_{accz} will be used for both equations).

Γ_{Rest} is the restoring moment created by the magnetic coupling along $\vec{y}_2$ when the rotors are not parallel ($\alpha \neq 0$):

$$\vec{\Gamma}_{Rest}(\theta_m, \theta_h, \alpha) \cdot \vec{y}_2 = \Gamma_{Resty} \tag{4.22}$$

I_1,I_2 and I_3, are the motor, fork and propeller axes inertia matrices respectively [146]:

$$\begin{aligned} I_1 &= \begin{bmatrix} I_{xx1} & I_{xy1} & I_{xz1} \\ I_{yx1} & I_{yy1} & I_{yz1} \\ I_{zx1} & I_{zy1} & I_{zz1} \end{bmatrix} \quad I_2 = \begin{bmatrix} I_{xx2} & I_{xy2} & I_{xz2} \\ I_{yx2} & I_{yy2} & I_{yz2} \\ I_{zx2} & I_{zy2} & I_{zz2} \end{bmatrix} \\ I_3 &= \begin{bmatrix} I_{xx3} & I_{xy3} & I_{xz3} \\ I_{yx3} & I_{yy3} & I_{yz3} \\ I_{zx3} & I_{zy3} & I_{zz3} \end{bmatrix} \end{aligned} \tag{4.23}$$

The dynamic equations (without friction) developed with the Newton-Euler algorithm are the following, as presented in [90] for a similar RMCT kinematics:

Motor axis in R_1

$$\tau_1 = I_{zz1}\ddot{\theta}_m + \Gamma_{Accz} \tag{4.24}$$

With τ_1 the torque generated by the motor.

Fork axis in R_2

$$\tau_2 = \ddot{\alpha}(I_{yy2} + I_{xx3} - \cos^2\theta_h I_{xx3} + \cos^2\theta_h I_{yy3} + m_2\ddot{\alpha}z_{G2}^2 + m_3 z_{G3}^2) + 2\dot{\alpha}\dot{\theta}_h \cos\theta_h \sin\theta_h (I_{xx3} - I_{yy3}) + \Gamma_{Resty} \tag{4.25}$$

With τ_2 the torque generated by the servomotor and applied to the axis support mechanism (fork). Γ_{Resty} is the restoring moment created by the magnetic coupling. z_{G2} and z_{G3} are the position of the center of gravity of the axis support and the propeller axis, respectively.

Propeller axis in R_3

$$\tau_3 = \ddot{\theta}_h I_{zz3} + \dot{\alpha}^2 \sin\theta_h \cos\theta_h (I_{yy3} - I_{xx3}) - \Gamma_{Accz} \tag{4.26}$$

With τ_3 the joint torque. No load torque is applied since the propeller is removed in our set up. Additionally, the experiments are made in the air, so the hydrodynamic effects responsible for this load are not present. An hydrodynamic model should be applied to compute τ_3 for an underwater propeller model.

Friction

Until now the model did not include the friction in the system. We call Γ_m, Γ_f and Γ_h the frictions torques of the motor, fork and propeller axes respectively. These terms are defined as follows:

$$\Gamma_m = sign(\dot{\theta}_m)\Gamma_{ms} + \dot{\theta}_m\Gamma_{mf}$$

$$\Gamma_f = sign(\dot{\alpha})\Gamma_{fs} + \dot{\alpha}\,\Gamma_{ff}$$

$$\Gamma_h = sign(\dot{\theta}_h)\Gamma_{hs} + \dot{\theta}_h\Gamma_{hf1} + sign(\dot{\theta}_h)\,\dot{\theta}_h^2\,\Gamma_{hf2} \tag{4.27}$$

where Γ_{ms}, Γ_{fs}, and Γ_{hs} are dry friction coefficients and Γ_{mf}, Γ_{ff}, and Γ_{hf} are viscous friction coefficients. Γ_{hf2} accounts for hydrodynamic effects, even if these effects are not expected to appear here, it has been included

in the model as a placeholder for future experimentation. We can write the complete joint torque equations including the friction terms in the following way:

$$\tau_m = \tau_1 + \Gamma_m \tag{4.28}$$

$$\tau_f = \tau_2 + \Gamma_f \tag{4.29}$$

$$\tau_h = \tau_3 + \Gamma_h \tag{4.30}$$

These torques contain the dynamic behavior of the subsystems, calculated thanks to the Newton-Euler recursive method, and additionally the friction effects.

4.2.3.2 Magnetic model

Propeller dynamic model

For our prototype we will consider $\tau_h = 0$ since there is no driving torque other than Γ_{accz}. Then from (4.26) and (4.30) Γ_{Accz} can be deduced:

$$\Gamma_{Accz} = I_{zz3}\,\ddot{\theta}_h + sign(\dot{\theta}_h)\,\Gamma_{hs} + \dot{\theta}_h\,\Gamma_{hf1} + sign(\dot{\theta}_h)\,\dot{\theta}_h^2\,\Gamma_{hf2} \tag{4.31}$$

which allows us to rewrite (4.28) as:

$$\tau_m = I_{zz1}\,\ddot{\theta_m} + sign(\dot{\theta}_m)\,\Gamma_{ms} + \dot{\theta}_m\,\Gamma_{mf} + \Gamma_{Accz} \tag{4.32}$$

This reduced model is developed to be used in the identification of the dynamic parameters of the system, these parameters are independent of the load τ_3 and of α.

Magnetic coupling torque model

We use the reference angles (θ_m and θ_h) to establish the magnetic angle θ. We can define it with the following equation (from Fig. 4.12),

$$\theta = \theta_m - \theta_h \tag{4.33}$$

as the difference between the angular position of the motor axis and the angular position of the propeller axis. This angle will determine the torque transmitted by the magnetic coupling. Indeed, the torque is created by the

attraction force of the magnets, which is a function of θ since it regulates the distance between magnets.

Coupling torque Γ_{accz} interpolation

To model the torque transmitted by the magnetic coupling we need to know the variation of the torque with respect to the reconfiguration angle α and the magnetic angle θ. The magnetic coupling behavior can be modeled analytically doing some simplifications [151, 152] but to account for its full complexity we use FEM computations [153].

Using two experimental curves (Fig. 4.11) provided by TE2M[3] (Flux3D) we can "build" an interpolating function that will describe the variation of the torque with α and θ.

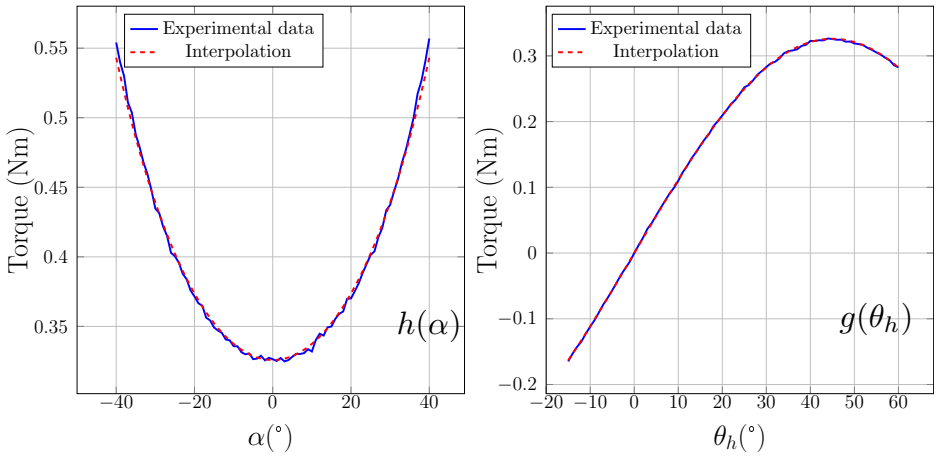

Figure 4.11: Interpolation curves for the magnetic coupling torque.

The variation of Γ_{accz} in neutral configuration ($\alpha = 0$) follows the magnetic forces behavior (Fig. 4.12): when $\theta = 0$ two magnets of opposite polarity are aligned, creating mutual attractive forces along z_1 and no magnetic torque is created on this axis. When θ increases, a magnetic force arises because of magnetic attraction. Halfway between two opposite polarity poles ($\theta = 45°$) the force is maximum, then decreases to zero when same polarity poles are aligned ($\theta = 90°$). In our application case, θ only varies from -45°to 45°. As it was shown in [152], the overall coupling torque dwindles exponentially with regard to coupling rotors distance along the z_1 axis and conversely.

[3]Technique et Matériel Magnetique. Company based in Brest, France

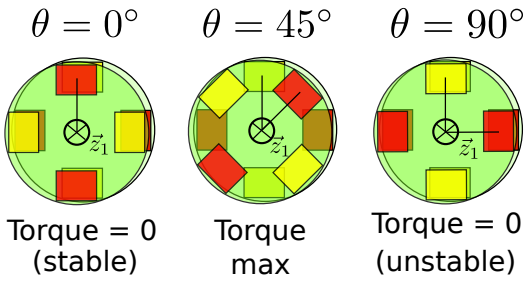

Figure 4.12: Magnetic coupling torque according to θ and polarity ($\alpha = 0$).

Studying each curve individually, we obtain two interpolating functions (with separated variables θ and α):

$$\Gamma_{Accz}(\theta, \alpha = 0°) = 0.3256 \sin 2\theta = g(\theta) \tag{4.34}$$

and

$$\Gamma_{Accz}(\theta = 45°, \alpha) = \frac{1}{\cos(0.876\alpha)} - 0.6736 = h(\alpha) \tag{4.35}$$

Combining these two functions in an extrapolating formula:

$$\Gamma_{accz}(\theta, \alpha) = 1.75\, sign(\theta) \sqrt{g(\theta)^2 h(\alpha)^2} \tag{4.36}$$

we get one bi-variable function that allows to extrapolate the value of the torque for the different positions of the α and θ angles. Figure 4.13 shows the surface generated by the interpolating bi-variable function (interpolating error $< 3\%$), including the two interpolated curves.

4.2.3.3 Electro-mechanical model

In order to complete the model of the prototype, the motor electrodynamics are needed. The equation describing the torque generated by the motor is the following [90]:

$$\dot{\tau}_m = \frac{\lambda}{L} U_m - \frac{\lambda^2}{L} \omega_m - \frac{R}{L} \tau_m \tag{4.37}$$

with

$$E = \lambda\, \omega_m$$

and

$$\tau_m = \lambda\, I_m$$

for a DC motor, λ is the speed or torque constant, I_m the current, L the inductance, R the winding resistance and E the back-EMF. These parameters

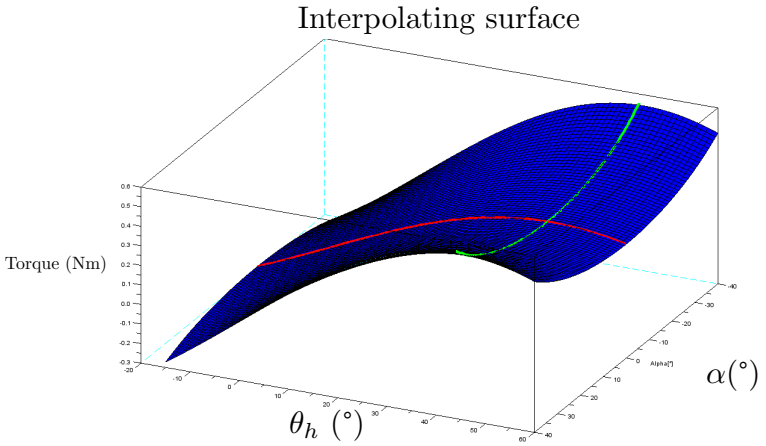

Figure 4.13: Extrapolating surface with experimental results (red for $\alpha = 0°$ and green for $\theta = 45°$).

are found in the technical documentation of the motor and are considered accurate enough to be used straightforwardly (confirmed by measurements):

- $\lambda = 38.9\, mNm/A$,
- $R = 2.7\, \Omega$,
- $L = 0.34\, mH$

With these equations and using the previously developed dynamic model, we can write the full state-space system equations:

$$\begin{cases} \dot{\theta}_m = \omega_m \\ \dot{\theta}_h = \omega_h \\ \dot{\tau}_m = \dfrac{\lambda}{L} U_m - \dfrac{\lambda^2}{L} \omega_m - \dfrac{R}{L} \tau_m \\ \dot{\omega}_m = \dfrac{1}{I_{z1}} \left(\tau_m - \Gamma_{Accz} - \Gamma_{Fm} \right) \\ \dot{\omega}_h = \dfrac{1}{I_{z3}} \left(\Gamma_{Accz} - \Gamma_{Fh} \right) \end{cases} \tag{4.38}$$

4.2.3.4 Experimental validation

Using the complete system model (with $\alpha = 0$) as well as the magnetic coupling curve interpolation and identified dynamic parameters values [150], we can model the prototype and simulate it. The model will be compared with experimental results in order to be validated.

This simulation will be carried out in open-loop and the results will be compared to the experimental data, obtained using the real prototype Fig. 4.14. This will allow to validate the models and the simulation.

In the prototype, the acquisition interface controls the motor and measures θ_m with its encoder. θ_h is measured with an encoder mounted on the propeller rotor shaft. The reconfiguration angle α can be controlled by the servomotor.

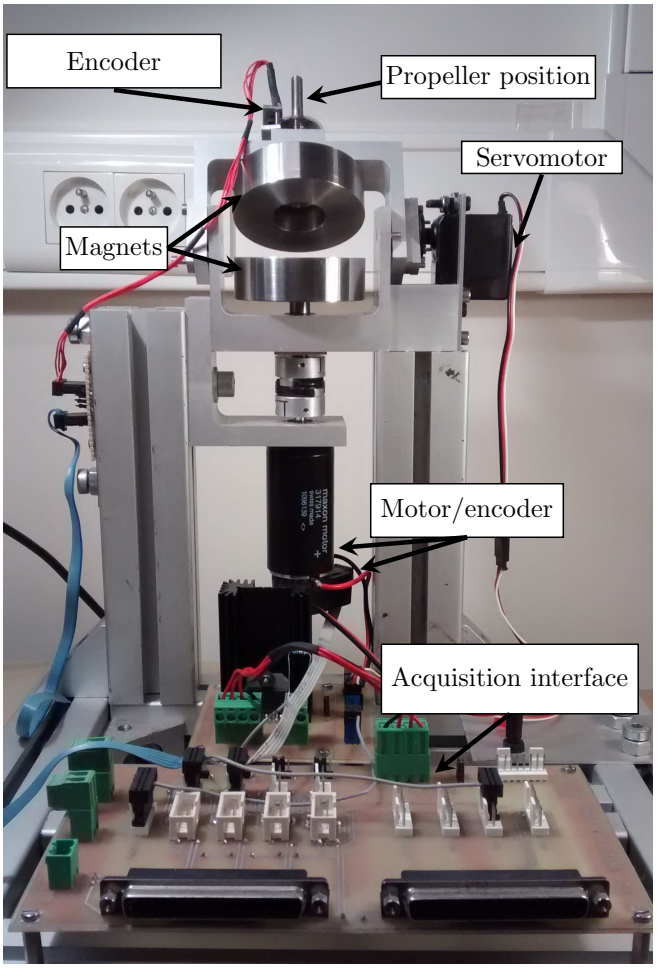

Figure 4.14: Flat RMCT prototype.

Results

Figure 4.15 shows the comparison between simulated and experimental angular speed of motor axis.

Making a zoom on the figure (transient and steady-state areas) we can observe oscillations in the speed, which is caused by the spring-mass behavior of the magnetic coupling and the motor frequency (due to mechanical defects).

We see that the simulation is capable of reproducing this oscillatory behavior in the transient zone (corresponding to the natural frequency of the magnetic coupling). In the steady-state zone we see that the oscillations do not appear in the simulation, this is because the mechanical defects are not simulated. In this area, however, there is a small error ($< 2\%$).

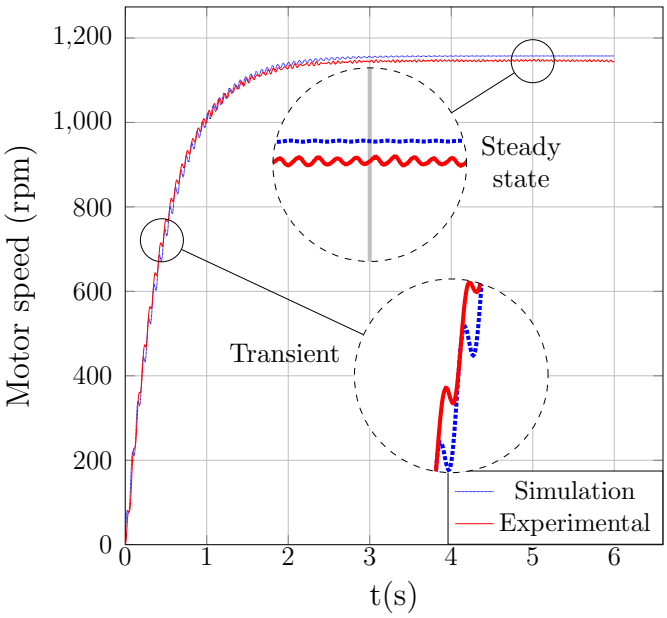

Figure 4.15: Simulation and experimental motor axis speeds. Input voltage of 6V.

Figure 4.16 shows the evolution of the magnetic angle θ, thus, the propeller axis behavior. The dynamics of this magnetic angle is well described by the simulation, since we see a good correlation between simulated and experimental magnetic angles (error $\simeq 10\%$). We can see as well, that the simulation also reproduces correctly the spring-inertia system oscillation at natural frequency. Also in Fig. 4.17, it is possible to observe that the simulated motor torque describes the behavior of the experimental one during

both the acceleration and steady-state phases (error $< 8\%$). These errors can be explained by the quantization error of the incremental encoders ($10^{-2} rad$) and the small amplitude of the magnetic angle θ, which is used to calculate Γ_{accz}. Another factor to take into account are the perturbations generated by the motor as well as the ones due to friction.

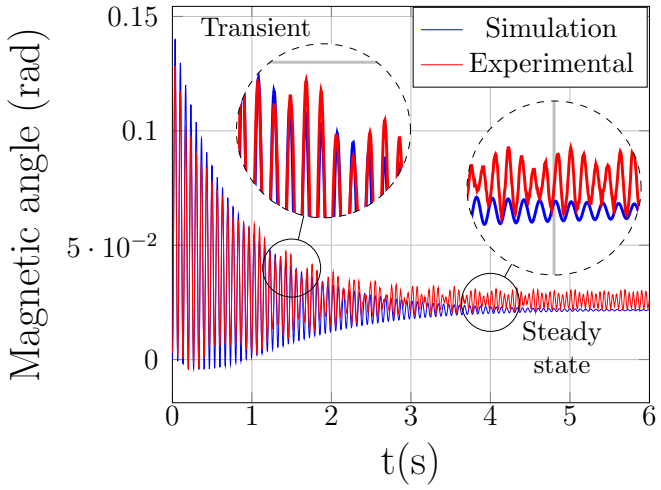

Figure 4.16: Evolution of simulated and experimental magnetic angle θ. Input voltage of 6V.

Periodic disturbances

The frequencies of the system are analyzed in Fig. 4.18 using a Fourier transform based on the motor angular speed, this frequency is the same for the motor and propeller rotor. In this figure we can see two main frequencies: One frequency ($\simeq 19Hz$) is generated by the rotation of the motor which means that the mechanism is not perfectly aligned. This frequency can also be calculated as:

$$f_{mot} = \frac{\omega_m}{2\pi} = 19.1Hz \tag{4.39}$$

where $\omega_m = 119.8\, rad/s$

Since this effect is due to imperfections in the system assembly, it can not appear in the simulation unless we need to estimate this effect.

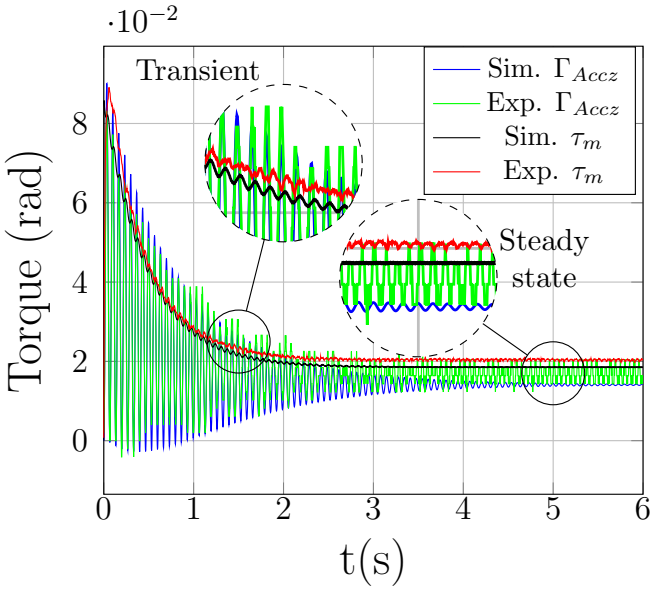

Figure 4.17: Simulation and experimental values of motor and magnetic coupling torque. Input voltage of 6V.

The second frequency is due to the natural frequency of the magnetic coupling ($15.1Hz$ from Fig.4.18). This frequency appears in the simulations since it depends on the mass-spring features of the magnetic coupling. It can be calculated using the following equation:

$$f_{\text{mag}} = \frac{1}{2\pi}\sqrt{\frac{K_\theta}{I_{zz3}}(1+\alpha^2)} = 15.14\,Hz \tag{4.40}$$

With $K_\theta = 6.54 \times 10^{-1}\,Nm/rad$, the magnetic coupling stiffness, which was calculated experimentally as the quotient of the transmitted torque and the magnetic angle θ.

$$K_\theta = \frac{\Gamma_{Accz}}{\theta}$$

$$\alpha^2 = \frac{I_{zz3}}{I_{zz1}}$$

where $I_{zz1} = 1.48 \times 10^{-4}$ and $I_{zz3} = 1.41 \times 10^{-4}$ as identified in [150].

The fact that we find the same frequencies shows that not only the identified inertias are correct but also that the magnetic coupling dynamic model is validated.

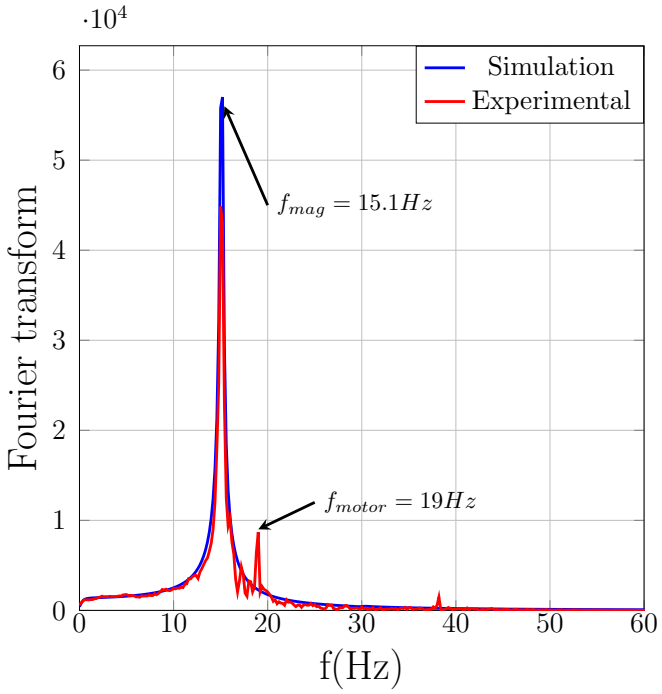

Figure 4.18: Frequency analysis of the rotor axis angular speed using Fourier Transform.

4.3 Simulation

In order to apply these models and to analyze the behavior of the underwater robot as a whole, a simulator has been developed. This simulator, called *EAUVIVE* [4], includes the robot models along with its propulsion system. The simulation has been written in several programming languages, such as MATLAB, Scilab and C++, and it is expandable, meaning that a control technique can be added without much effort.

The EAUVIVE simulator is programmed to represent the behavior of the RSM robot (Fig. 4.19), the dynamic and hydrodynamic parameters will be based in its structure and characteristics. Furthermore, two propulsive topologies will be implemented in order to compare the different propulsive topologies under different working conditions. The simulation results will also allow to perform a preliminary analysis of the propulsive systems.

[4]ENIB AUV In Virtuo Experiment

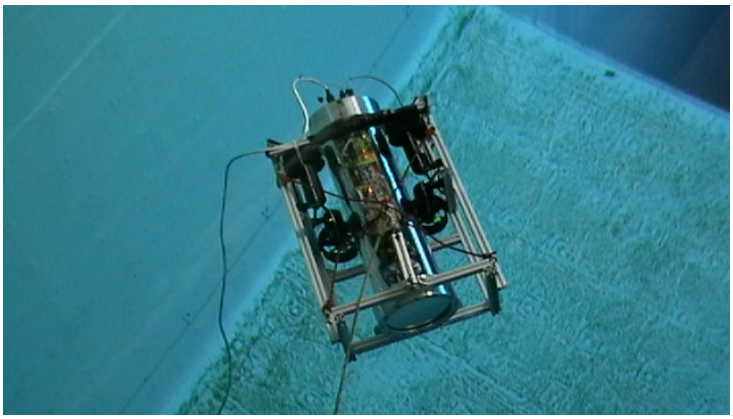

Figure 4.19: RSM robot tested in Ifremer bassin (sea water, 20 m).

4.3.1 The RSM robot model

4.3.1.1 Hydrodynamic model

In our simulations and analyzes an AUV with cylindrical hull will be used, which is meant to approximate the shape of the RSM robot. This will allow to calculate the added mass and damping terms using the simplifications proposed in the literature [3,146]. The dimensions shown in Fig. 4.20 correspond to an existing AUV (robot RSM, Fig. 4.19), available in the laboratory as an experimental platform. Details regarding the RSM AUV characteristics are shown in Table 4.1.

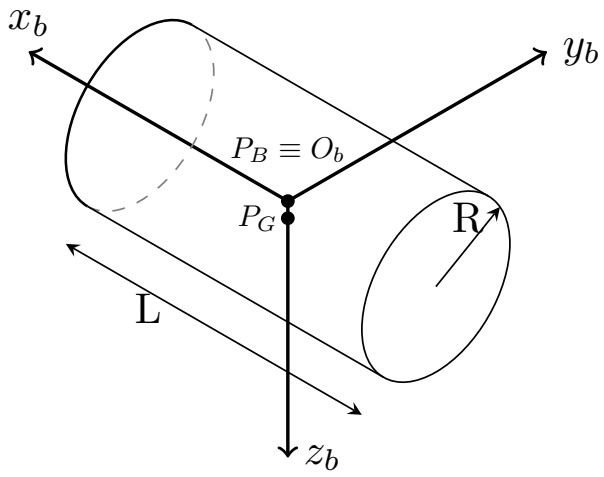

Figure 4.20: Simplified representation of the RSM AUV.

Table 4.1: RSM characteristics.

Category	Symbol	Name and unit	Value
Rigid Body	R	Circular section radius [m]	0.1
	L	Length [m]	0.6
	m	Mass [kg]	18.71
	W	Weight [N]	183.6
	B	Buoyancy force [N]	184.5
	${}^b\mathbf{P}_B$	Pos. buoyancy center [m]	$[0 \;\; 0 \;\; 0]^T$
	${}^b\mathbf{P}_G$	Pos. gravity center [m]	$[0 \;\; 0 \;\; 0.0125]^T$
	J	Inertia matrix $[kg\,m^2]$	$\begin{bmatrix} 0.097 & 0 & 0 \\ 0 & 0.611 & 0 \\ 0 & 0 & 0.608 \end{bmatrix}$

These values will be useful later, to calculate different parameters and coefficients to run the simulation.

In what concerns the added mass effects, we can make some simplifications [146]. For a three plane of symmetry AUV we can neglect the contribution of the non-diagonal elements of the added mass matrix. Consequently, only the diagonal elements are taken into account.

$$\mathbf{M}_a = -diag\{X_{\dot{u}}, Y_{\dot{v}}, Z_{\dot{w}}, K_{\dot{p}}, M_{\dot{q}}, N_{\dot{r}}\} \tag{4.41}$$

For the cylindrical model of the RSM robot:

$$X_{\dot{u}} = -0.1m$$

$$Y_{\dot{v}} = -\pi\rho r^2 L$$

$$Z_{\dot{w}} = -\pi\rho r^2 L$$

$$K_{\dot{p}} = 0$$

$$M_{\dot{q}} = -\frac{1}{12}\pi r^2 L^3$$

$$N_{\dot{r}} = -\frac{1}{12}\pi\rho L^3$$

In the case of $K_{\dot{p}}$, we can consider that no water is moved for roll motion.

For the same type of vehicle, the general form of the added mass contribution to the Coriolis and centripetal effects matrix can be simplified as follows:

$$\mathbf{C}_a = \begin{bmatrix} 0 & 0 & 0 & 0 & -Z_{\dot{w}}w & Y_{\dot{v}}v \\ 0 & 0 & 0 & Z_{\dot{w}}w & 0 & -X_{\dot{u}}u \\ 0 & 0 & 0 & -Y_{\dot{v}}v & X_{\dot{u}}u & 0 \\ 0 & -Z_{\dot{w}}w & Y_{\dot{v}}v & 0 & -N_{\dot{r}}r & M_{\dot{q}}q \\ Z_{\dot{w}}w & 0 & -X_{\dot{u}}u & N_{\dot{r}}r & 0 & -K_{\dot{p}}p \\ -Y_{\dot{v}}v & X_{\dot{u}}u & 0 & -M_{\dot{q}}q & K_{\dot{p}}p & 0 \end{bmatrix} \tag{4.42}$$

Likewise, the cylindrical geometry of the AUV allows to simplify the damping matrix. Only the linear and quadratic diagonal terms are considered [3].

$$\mathbf{D}_v = -diag\{X_u, Y_v, Z_w, K_p, M_q, Nr\}+ \tag{4.43}$$
$$-diag\{X_{u|u|}|u|, Y_{v|v|}|v|, Z_{w|w|}|w|, K_{p|p|}|p|, M_{q|q|}|q|, N_{r|r|}|r|\}$$

Furthermore, another simplification can be made if we consider that the AUV speed is high enough to neglect linear terms with respect to the quadratic ones [154] (and while the robot speed is very low, none of these effect are significant anyway.). In that case, only quadratic terms are included in the damping matrix, using formulas adapted from [155, 156]:

$$X_{u|u|} = -\frac{1}{2}\rho\ c_d\ A_f$$

$$Y_{v|v|} = Z_{w|w|} = -\rho\ c_{dc}\ r\ L$$

$$M_{q|q|} = N_{r|r|} = -\rho\ c_{dc}\ r\ \frac{L^4}{4}$$

$$K_{p|p|} = 0$$

Where c_d is the axial drag coefficient, which is a function of the Reynolds number [3]; A_f is the vehicle frontal area; c_{dc} is the cross flow drag coefficient.

In our case we are analyzing a finless structure, thus rolling drag coefficient, $K_{p|p|}$ is zero. An exception should be made for this term since it is null. Then [157]:

$$K_{p|p|} = 2\,\pi\,\mu\,R^2\,L = 3.77\ \times 10^{-5}\ Nm/rad/s \tag{4.44}$$

with $\mu = 0.001\,Pa\,s$, the dynamic viscosity of the water ($20°C$).

Considering the very small value of this torque, it can therefore be neglected in our simulations (with regard to the other effects). However, given

that our robot has a square superstructure holding the tube, we set an experimentally estimated coefficient $K_{p|p|} = 0.38\ Nm/rad/s$.

Regarding restoring forces and moments vector, **G**, its calculation is straightforward if the mass, volume, center of gravity and buoyancy are known.

A list of the added mass and hydrodynamic damping terms (based in Eq. 4.41 and Eq. 4.43) is shown on Table 4.2

Table 4.2: Added mass and hydrodynamic terms.

RSM Robot			
Added Mass	$X_{\dot{u}}$	-1.872	kg
	$Y_{\dot{v}}$	-1.881	kg
	$Z_{\dot{w}}$	-1.881	kg
	$K_{\dot{p}}$	0.0	$kg.m^2$
	$M_{\dot{q}}$	-0.057	$kg.m^2$
	$N_{\dot{r}}$	-0.057	$kg.m^2$
Hydrodynamic Damping	$X_{u\|u\|}$	-12.541	kg/m
	$Y_{v\|v\|}$	-71.856	kg/m
	$Z_{w\|w\|}$	-71.856	kg/m
	$K_{p\|p\|}$	0.0	$kg.m^2/rad^2$
	$M_{q\|q\|}$	-3.88	$kg.m^2/rad^2$
	$N_{r\|r\|}$	-3.88	$kg.m^2/rad^2$

4.3.1.2 Propulsion system model

The simulator needs to know now the propulsive topology of the robot. Here we are going to present two different propulsions and apply them to the structure of the RSM robot. The first one is the widely used fixed propulsion system and the second one is the, relatively recent, vectorial propulsion system. Additionally, the advantages and drawbacks of these systems will be discussed.

Fixed propulsion

Robots with this type of propulsion are powered by several thrusters placed along different axis in order to combine their thrust and provide six (or less) actuated DOF to the robot. Various architectures within this group can be found, each one with advantages and drawbacks. they share the same principle, which is multi-directional propulsion based on thrust combination.

In this work a four thruster configuration is tested (corresponding to our RSM robot). As shown in Fig. 4.21, two horizontal thrusters are placed at the rear of the AUV and two vertical thrusters are centered symmetrically on the sides of the robot. This configuration provides actuation over the following DOF:

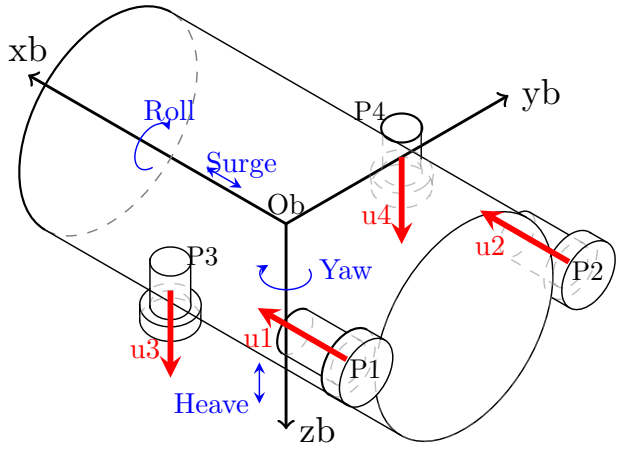

Figure 4.21: RSM fixed propeller architecture.

- Surge (linear motion along the $\mathbf{x}_b$-axis)
- Heave (linear motion along the $\mathbf{z}_b$-axis)
- Yaw (rotation along the $\mathbf{y}_b$-axis)
- Roll (rotation along the $\mathbf{x}_b$-axis)

Conversely, due to the location of the four thrusters, the following DOF can not be actuated:

- Pitch (rotation along $\mathbf{y}_b$-axis)
- Sway (linear motion along $\mathbf{y}_b$-axis)

Roll and Pitch motion are mechanically stabilized thanks to the relative position of the buoyancy and gravity centers.

The sway is not controlled and not stabilized, thus free motion on $\mathbf{y}_b$-axis (drift) is expected to occur.

The fact that more thrusters should be added to the architecture in order to obtain controlled pitch and sway motions, shows a disadvantage of this type of propulsive strategy (lack of control in cartesian space). To control the 6 DOF at least 6 thrusters must be installed in the underwater vehicle, which means a large power consumption, an increased weight and a higher cost. Another important drawback is that in order to turn, the AUV propellers (and motors) must accelerate/decelerate. This implies a lag time between the moment the force is needed and the instant it is generated.

The position of each thruster is given, with regard to R_b, by the following vectors (in meters):

$$ {}^b\mathbf{P}_1 = \begin{bmatrix} P_{1x} \\ P_{1y} \\ P_{1z} \end{bmatrix} = \begin{bmatrix} -0.21 \\ -0.12 \\ 0 \end{bmatrix} ; {}^b\mathbf{P}_2 = \begin{bmatrix} P_{2x} \\ P_{2y} \\ P_{2z} \end{bmatrix} = \begin{bmatrix} -0.21 \\ 0.12 \\ 0 \end{bmatrix} $$

$$ {}^b\mathbf{P}_3 = \begin{bmatrix} P_{3x} \\ P_{3y} \\ P_{3z} \end{bmatrix} = \begin{bmatrix} 0 \\ -0.105 \\ 0 \end{bmatrix} ; {}^b\mathbf{P}_4 = \begin{bmatrix} P_{4x} \\ P_{4y} \\ P_{4z} \end{bmatrix} = \begin{bmatrix} 0 \\ 0.105 \\ 0 \end{bmatrix} $$

The TCM can then, taking into account the alignments, be calculated as:

$$ \mathbf{B} = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & -P_{3y} & -P_{4y} \\ P_{1z} & P_{2z} & P_{3x} & P_{4x} \\ -P_{1y} & -P_{2y} & 0 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & -P_{3y} & -P_{4y} \\ 0 & 0 & 0 & 0 \\ -P_{1y} & -P_{2y} & 0 & 0 \end{bmatrix} \tag{4.45} $$

for the input vector:

$$ \mathbf{u}_p = \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} $$

Where u_1, u_2, u_3 and u_4 are the thruster propulsion forces.

Vectorial Propulsion

Vectorial propulsion combines thrust and steering in the same thruster since the direction of thrust force can be changed with regard to R_b. In an underwater robot fitted with a vectorial thruster placed at the rear, the actuated DOF are:

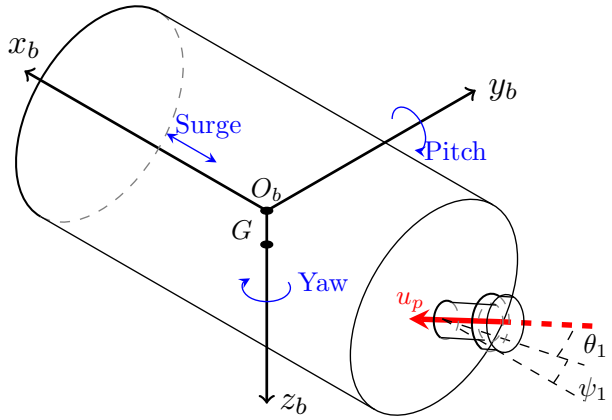

Figure 4.22: RSM Vectorial Propulsion.

- Surge
- Pitch
- Yaw

The actuation of these DOF is achieved using only one propeller, which is an interesting feature of this propulsive strategy, because it minimizes on board devices.

The remaining DOF can not be actuated separately because whether they are coupled or simply not actuated:

- Heave (coupled with pitch)
- Sway (coupled with yaw)
- Roll (not actuated)

The limitations of this propulsive strategy must be taken into account in its evaluation. For instance, the angle range depends on the technology carrying out the propulsion and steering task. Previous works [90] indicates that a $\pm 30°$ range is feasible using magnetic coupling technology. 45°could

be achieved without too much efforts, (even up to 60°) but in theory nothing but practical implementation prevents from reaching 90°or even more).

The vectorial thruster is placed at the rear of the AUV along the $\mathbf{x}_b$-axis as shown in Fig. 4.22. Since the thrust force can be reoriented, vectorial propulsion is obtained. This means very complex 3D movements can be achieved by controlling this vector. Another important advantage of this technology is that we can change the direction of the thrust without inverting the rotation of the propeller. Therefore, losses (in time and energy) caused by acceleration and deceleration are avoided. Nevertheless, a rapid change in the direction of the rotating propeller will create a perturbation in the thrust vector (conservation of angular momentum). Said perturbation has to be rejected by the controller in order to ensure the performance of the propulsion system. This means that the complexity of the controller will increase, making the design of the underwater vehicle more challenging.

The position of the vectored thruster with regard to R_b is given by (in meters):

$$ {}^b\mathbf{P}_1 = \begin{bmatrix} P_{1x} \\ P_{1y} \\ P_{1z} \end{bmatrix} = \begin{bmatrix} -0.21 \\ 0 \\ 0 \end{bmatrix} $$

The configuration matrix of this propulsive strategy is then:

$$ \mathbf{B} = \begin{bmatrix} \cos\theta_1 \cos\psi_1 \\ \cos\theta_1 \sin\psi_1 \\ -\sin\theta_1 \\ -\sin\theta_1 P_{1y} - \cos\theta_1 \sin\psi_1 P_{1z} \\ \cos\theta_1 \cos\psi_1 P_{1z} + \sin\theta_1 P_{1x} \\ \cos\theta_1 \sin\psi_1 P_{1x} - \cos\theta_1 \cos\psi_1 P_{1y} \end{bmatrix} = \begin{bmatrix} \cos\theta_1 \cos\psi_1 \\ \cos\theta_1 \sin\psi_1 \\ -\sin\theta_1 \\ 0 \\ \sin\theta_1 P_{1x} \\ \cos\theta_1 \sin\psi_1 P_{1x} \end{bmatrix} \quad (4.46) $$

for the input vector:

$$ u_p = \begin{bmatrix} u_1 \end{bmatrix}; \quad \mathbf{u}_v = [u_x \;\; u_y \;\; u_y]^T; \quad u_1 = ||\mathbf{u}_v|| $$

Where ψ_1 and θ_1 are the configuration angles (RPY convention, with simultaneous rotations around the fixed axes $\mathbf{y}_b$-axis and $\mathbf{z}_b$-axis, respectively). These angles are necessary to define $\mathbf{u}_v$, the Cartesian thrust vector in R_b as shown in Fig. 4.22.

4.3.2 EAUVIVE simulator

4.3.2.1 Numeric resolution

Mathematical models provided in the previous sections are integrated in the EAUVIVE numerical simulation code (Fig. 4.23), developed in the IRDL lab. This simulator (developed in Scilab, Matlab and C++), allows evaluating AUV behavior from the point of view of 3D solid -and hydrodynamics. In its simulation, EAUVIVE integrates the models of the robot, a control module (open loop for the following tests) and a graphical 3D visualization of the robot. Additionally, the simulator allows to add a closed loop controller, in that case, EAUVIVE includes also a desired trajectory to be followed by the AUV and a control algorithm.

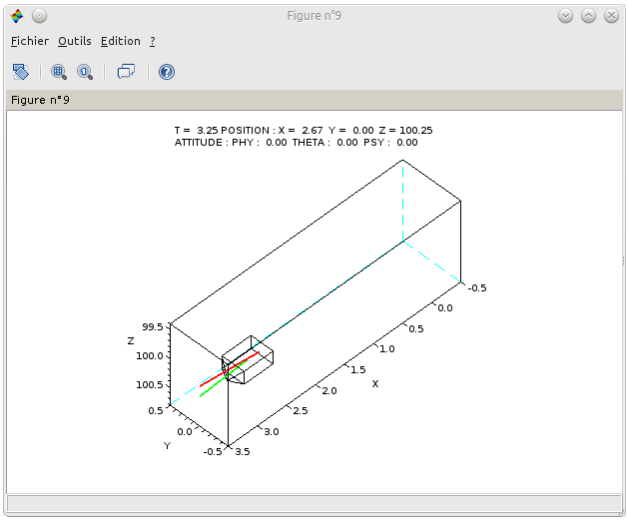

Figure 4.23: EAUVIVE Simulator (Scilab Version)

For the sake of the analysis, we will use the simulator in a set of maneuvers. These maneuvers are established in order to have a performance evaluation of the presented propulsion systems. Proposed maneuvers aim to replicate the most common locomotion tasks performed by AUVs during its deployment [158, 159]. Since this is a preliminary work, only speed will be the objective criterion for performance of the propulsive strategies.

Normalization of the propulsion:

Since the propellers are not the same in both architectures, a maximum overall total thrust of 20 N is assigned to each propulsive strategy. In the

case of the vectorial propulsion, the two servomotors controlling the propeller configuration have been assigned 1 N equivalent each. This value has been estimated using considerations based on the power consumption of the servomotor used in RMCT (Futaba S3010)[5] and RSM thruster devices (Seabotix BTD 150)[6].

4.3.2.2 Tasks and trajectories

Basic tasks

Each task being driven by open-loop controller, the control law have been designed and tuned by hand with trial and error method (respecting given limitations).

1. **Top Speed**:

   Top speed is determined for forward motion (along the $\mathbf{x}_0$-axis). The proposed hull has positive buoyancy, this means that a part of the power consumption in both systems is spent for keeping the AUV at the same depth during the task (depth regulation).

   For the fixed propeller architecture, surge force is given by the two rear thrusters, both working at 9.425 N. Depth regulation is obtained by applying a downward thrust force of 0.575 N on each vertical thruster. Top speed is then 1.22 m/s.

   In the case of the vectorial propulsion strategy, forward motion and depth regulation are achieved by orientating the thrust vector on pitch axis (θ_1 = -7°). Since the thrust force is 18 N (20 N minus 2 N for servomotors), the resultant speed is slightly below to the previous configuration: 1.19 m/s.

2. **Dive Speed:**

   This simple task shows diving capabilities of the AUV equipped with the proposed propulsive strategies. Each type of propulsion requires a different method to achieve downward motion. This particular maneuver is focused only in finding the maximum dive speed without taking into account nor moderating any motions in the horizontal plane.

   For the fixed strategy, the vertical thrusters pull the vehicle downwards with a thrust force of 10 N each. In this mission, the drag forces along

[5] www.futaba-rc.com/servos/analog.html
[6] www.seabotix.com/products/auv_thrusters.htm

$\mathbf{z}_b$ are higher. This is why the dive speed is 0.515 m/s, lower than forward speed.

For the vectorial propulsion strategy, the descent motion is achieved by pitching the vectorial thruster (ψ_v = -30°), allowing the AUV to rotate downwards. Nevertheless, due to the restoring moment of the body, a pure vertical descent is not possible and it is always coupled with a forward displacement (on $\mathbf{x}_0$-axis) as shown in Fig. 4.24.

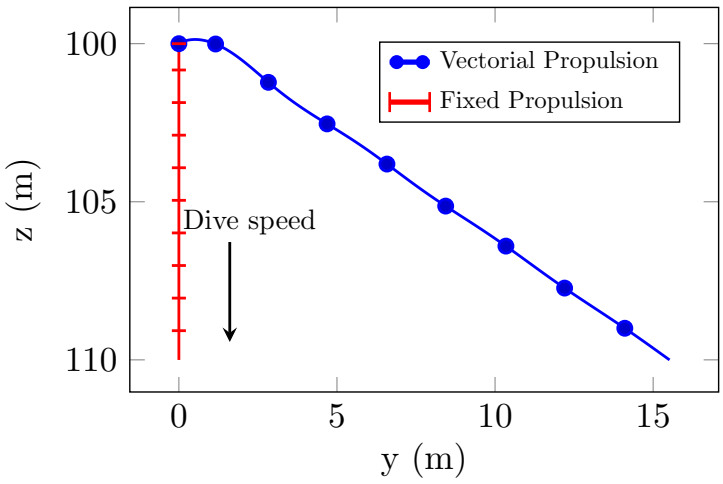

Figure 4.24: Displacements in dive tasks.

In this conditions, the dive speed is 0.641 m/s, which is significantly better than the one achieved by the fixed propulsion system.

3. **Circular Speed:**

Turning capability is an appropriate way to determine the maneuverability of an AUV. This task aims to determine the highest tangential speed achievable by the AUV while following a 5 L radius circular trajectory.

For the case of the fixed propulsion strategy, vertical thrusters have to be active to counteract buoyancy forces. Vertical thrusters are set to 0.3 N each while horizontal thrusters are set to 18,375 N and 1.025 N. The speed achieved with this strategy is 0.992 m/s. Figure 4.25 shows the circular trajectory for both strategies.

In vectorial propulsion, a vertical component of the thrust vector must be applied in order to cancel buoyancy forces (θ_1 = -5.2°and ψ_1 = 26.5°).

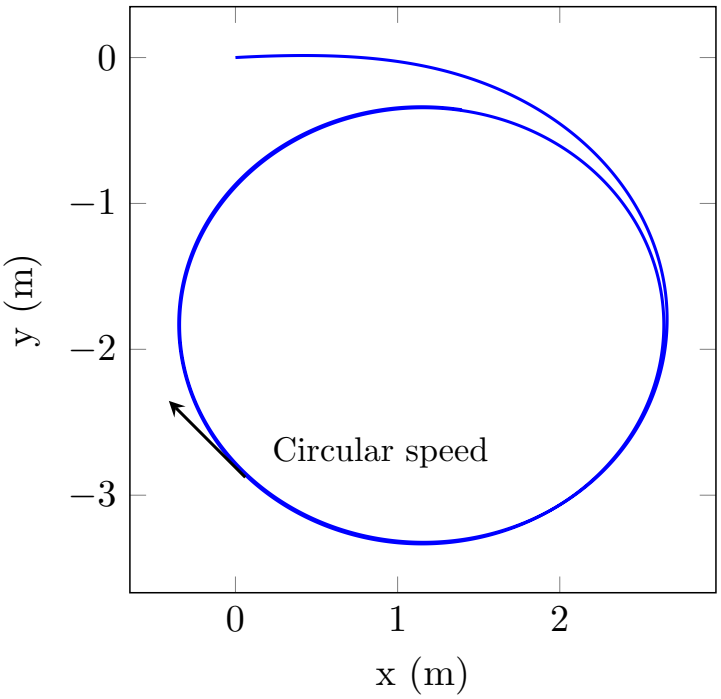

Figure 4.25: Circular trajectory.

The achieved tangential velocity of the AUV with vectorial propulsion is 0.996 m/s, which is slightly better than the fixed thruster system.

Mission-oriented tasks

1. **Diving in a confined space:**

   The motion will be limited by a maximum displacement in the horizontal plane (descent into a well), this allows to evaluate the propulsion strategy in a realistic mission.

   Available diameter to perform the maneuver will be limited to 5 L (Fig. 4.26).
   Since the fixed propeller configuration has two vertical thrusters placed near the center of gravity, this task is straightforwardly performed. In fact, the AUV descends along a vertical line (as for dive speed task). The dive speed of the AUV is 0.515 m/s.

   Because of coupled forward displacement seen in the diving task, the vectorial propulsion needs to follow a spiral trajectory as shown in Fig.

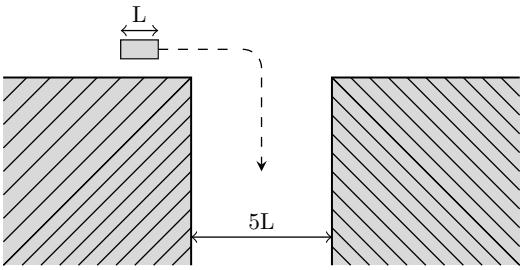

Figure 4.26: Diving in confined space.

4.27. In order to perform such a motion, vectored thruster angles ψ_1 and θ_1 have been set to 10 and -30 degrees respectively.

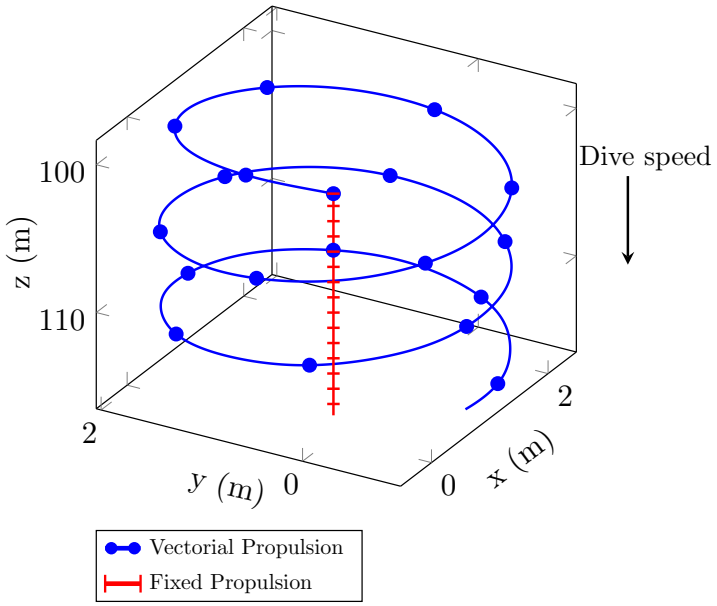

Figure 4.27: Spiral trajectory in confined dive motion.

Due to a better orientation of the robot, drag forces are lower than in the case of the previously discussed propulsive strategy. This advantage has a positive effect on the AUV dive speed, which is 0.555 m/s. This strategy gets better results despite of the higher traveled distance involved by the spiral trajectory.

2. **Horizontal slalom:**

A more dynamic insight of the agility of an AUV is given by its performance in a slalom trajectory around gates. For this mission, a tra-

jectory based on the robot length is proposed. The amplitude of the slalom is 3.33 L and the longitudinal distance between two gates is 9 L (Fig. 4.28). The average speed along the $\mathbf{x}_0$-axis is computed to evaluate the performance of the AUV to complete the slalom mission.

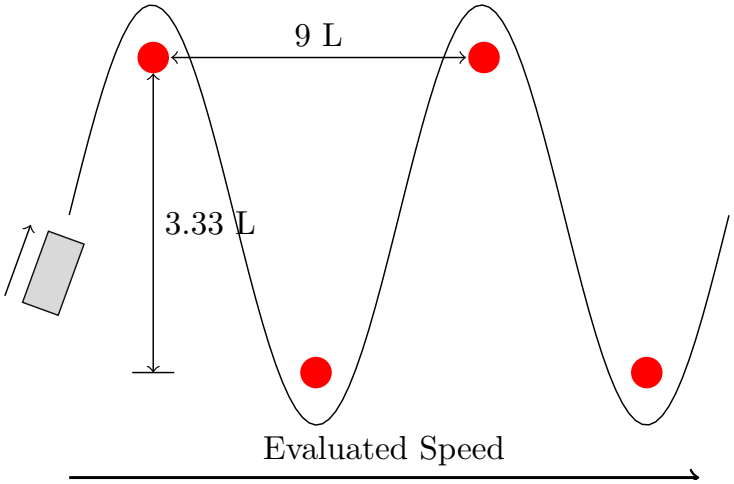

Figure 4.28: Slalom trajectory (horizontal or vertical).

For fixed-propulsion, the open loop control law to follow the proposed path (Fig. 4.28) is:

$$u_1 = 9.4 - 4.5\cos(0.75\ t)$$
$$u_2 = 9.4 + 4.5\cos(0.75\ t)$$
$$u_3 = 0.35$$
$$u_4 = 0.35$$

The task speed to follow this trajectory is 1.066 m/s.

The vectorial thruster configuration is:

$$u_1 = 18$$
$$\theta_1 = -5$$
$$\psi_1 = 12.75\cos(0.72\ t)$$

the AUV equipped with vectorial propulsion achieves a speed of 1.048 m/s, slower but close to the fixed strategy.

3. **Vertical slalom:**

   This slalom is similar to the previous one but must be performed in the vertical plane. An additional difficulty for the AUV is that it has to cope with its positive buoyancy. Again, the objective of the task is to evaluate the average speed along the $\mathbf{x}_0$-axis.

   For the fixed propulsion system, the following open loop control law has been used:

$$u_1 = 3$$
$$u_2 = 3$$
$$u_3 = 0.5 + 6.5\cos(0.4\ t)$$
$$u_4 = 0.5 + 6.5\cos(0.4\ t)$$

   The evaluated speed is 0.692 m/s.

   The vectorial system open loop control is is:

$$u_1 = 18$$
$$\theta_1 = -1.6 + 30\cos(0.7\ t)$$
$$\psi_1 = 0$$

   Despite the presence of a restoring moment, the vectored architecture succeed to follow the proposed path, achieving a speed of 1.019 m/s, which is significantly better.

4.3.2.3 Results

Table 4.3 summarizes the results for the two propulsive strategies analyzed in the previous section. As can be seen, some of them give advantage to fixed propulsion (rather small) and some others to its vectorial counterpart (more significant).

In forward speed and in horizontal slalom tasks, the fixed strategy is slightly better, which can be explained by the use of non thrusting actuators (servomotors) for the vectorial one.

Vectorial propulsion has a better performance in diving tasks. This is explained by the fact that vectorial propulsion can orientate the robot body in order to have weaker drag forces during the progression. This gives a

Table 4.3: Summary of results

RSM Robot		
Task	Fixed Propulsion	Vectorial Propulsion
Forward speed	1.22 m/s	1.19 m/s
Dive Speed	0.515 m/s	0.641 m/s
Circular Speed	0.992 m/s	0.996 m/s
Diving in a confined space	0.515 m/s	0.555 m/s
Horizontal slalom	1.066 m/s	1.048 m/s
Vertical slalom	0.692 m/s	1.019 m/s

significant advantage to the vectorial propulsion approach and shows the potential superiority of this technology.

It can be noticed that the vectorial propulsion exhibits better performances for the two missions that require a depth variation (dive speed and vertical slalom). This superiority over the fixed propulsion is due to the fact that the vectorial one uses its full power in all directions while the other needs to share its power between different directions.The vectorial strategy improves the isotropy of the propulsion system.

One must notice than the dynamics of thruster has not been taken into account (accelerations/deceleration and gyroscopic effects), nor the power consumption. If it had, then the vectorial propulsion would have clearly be favored because of the energy and time required to achieve the variations of thrust forces.

4.4 Conclusion

In this chapter, the dynamic model of an AUV including the hydrodynamic effects has been studied. This model has been adapted to the RSM robot, a prototype used as a testbed and reference in IRDL laboratory.

The models of two propulsive technologies have been presented as well. The first model describes the behavior of a fixed thruster, the Seabotix BTD150, a popular *off the shelf* thruster. The second model describes a magnetic coupling vectorial thruster, which is based on RMCT prototype developed in IRDL laboratory. This prototype allows to test an enabling technology for the concept of vectorial thruster.

Finally, an AUV simulator has been developed and a brief comparison between fixed and vectorial propulsive topologies have been carried out. The simulator will allow to evaluate and study the behavior of the AUV.

Chapter 5

Control and estimation

After having modeled the robot, the next steps in the study of the underwater vehicle are the control technique and estimation methods used on it. These are the key elements to simulate the behavior of the robot.

Given that we have a very precise model of the robot, it seems logical to make intensive use of it. The chosen method is the *computed torque* (or feedback linearization), a nonlinear control method that explicitly uses the kinematic and dynamic model of the robot. Using the limits of this control method as a starting point, the controllability of underwater robots will be discussed as well.

An extended Kalman filter will be used in order to estimate the needed signals for the controller. The implementation of the control and estimation methods will be done using the ROS (Robotic Operative System) framework.

5.1 Torque computation

5.1.1 Principle

This control method is broadly used to design feedback controllers for AUVs [160, 161]. The method straightforwardly uses the kinematic and dynamic models of the robot. This characteristic offers a clear view of the internal workings of the controller, which will allows us to draw conclusions about controllability and learn about the robot in general. The idea behind the computed torque method is to algebraically transform nonlinear systems into (fully or partly) linear ones, so that linear control techniques can be applied (Fig. 5.1).

This control differs from conventional linearization method as lineariza-

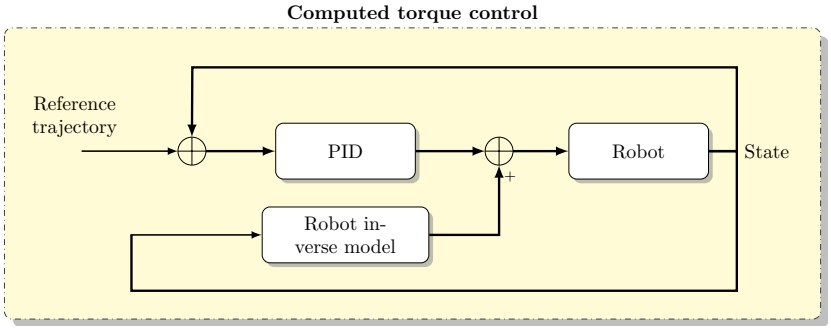

Figure 5.1: Computed torque control method principle.

tion is achieved by exact state transformation and feedback, rather than by dynamic linear approximations. Computed torque control is based on the idea of controlling a system by generating a control input that cancels out the nonlinearity of the dynamic model and then applies a correction to the therefore linearized system.

The control method used here consists of a kinematic and a dynamic control level (Fig 5.2). The kinematic level generates the AUV reference velocity ν_{ad} which will make an arbitrary point e of the robot follow a desired trajectory η_{ed}. In order to calculate this speed, it uses the information of the velocity of the desired trajectory $\dot{\eta}_{ed}$ and a correction term proportional to the error in position and orientation.

The dynamic level computes the needed wrench to be applied to the AUV (linearization) such that it follows ν_{ad}. It also adds a correction term proportional to the velocity error K_p. This method is quite efficient as long as the propulsive architecture is able to generate the calculated wrench τ_a and the model is very precise.

5.1.2 Kinematic level

The kinematic level is in charge of the calculation of the correct robot velocity $\boldsymbol{\nu}$ in order to make the point e converge toward the reference trajectory $\boldsymbol{\eta}_{ed}$. To develop the corresponding control law, we start by determining the velocity of the tracking point e.

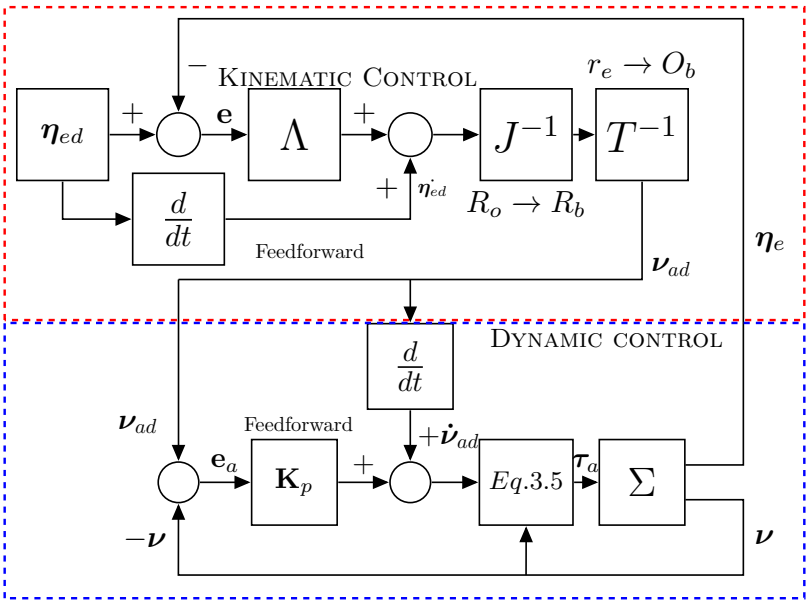

Figure 5.2: Computed torque control layers.

$$\begin{cases} \boldsymbol{\nu}_{1e} = & \boldsymbol{\nu}_1 + \boldsymbol{\nu}_2 \times \vec{r}_e \\ \boldsymbol{\nu}_{2e} = & \boldsymbol{\nu}_2 \end{cases} \quad \text{in} \quad R_b \tag{5.1}$$

$$\begin{cases} \dot{\boldsymbol{\eta}}_{1e} = & \mathbf{J}_1(\boldsymbol{\eta}_2)\,\boldsymbol{\nu}_{1e} \\ \dot{\boldsymbol{\eta}}_{2e} = & \mathbf{J}_2(\boldsymbol{\eta}_2)\,\boldsymbol{\nu}_2 \end{cases} \quad \text{in} \quad R_0 \tag{5.2}$$

Where the position vector of e in R_b is:

$$\vec{r}_e = [\epsilon_x, \epsilon_y, \epsilon_z]^T \tag{5.3}$$

Indeed, in Eq. 5.1 we calculate the velocity of e using rigid body kinematics.

We can re-write Eq. 5.1 in the following way:

$$\boldsymbol{\nu}_e = \begin{bmatrix} \boldsymbol{\nu}_1 + \boldsymbol{\nu}_2 \times \vec{r}_e \\ \boldsymbol{\nu}_2 \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 & \epsilon_z & -\epsilon_y \\ 0 & 1 & 0 & -\epsilon_z & 0 & \epsilon_x \\ 0 & 0 & 1 & \epsilon_y & -\epsilon_x & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}}_{\mathbf{T}} \underbrace{\begin{bmatrix} u \\ v \\ w \\ p \\ q \\ r \end{bmatrix}}_{\boldsymbol{\nu}} \tag{5.4}$$

From the previous equation we can see that matrix $\mathbf{T}$ is the link between the robot velocity and velocity of the tracking point.

Taking into account Eq. 5.4, we can turn Eq. 5.2 into:

$$\dot{\boldsymbol{\eta}}_e = \mathbf{J}(\boldsymbol{\eta}_2)\,\nu_e = \mathbf{J}(\boldsymbol{\eta}_2)\,\mathbf{T}\,\boldsymbol{\nu} \tag{5.5}$$

allowing us to determine the relation between the point e velocity $\dot{\boldsymbol{\eta}}_e$ and the robot velocity $\boldsymbol{\nu}$:

$$\boldsymbol{\nu} = \mathbf{T}^{-1}\,\mathbf{J}(\boldsymbol{\eta}_2)^{-1}\,\dot{\boldsymbol{\eta}}_e \tag{5.6}$$

Based on this relation, and using the desired trajectory $\boldsymbol{\eta}_{ed}$ and its derivative $\dot{\boldsymbol{\eta}}_{ed}$, we can develop the following kinematic control law [162]:

$$\boldsymbol{\nu}_{ad} = \mathbf{T}^{-1}\left\{\mathbf{J}^{-1}\left[\dot{\boldsymbol{\eta}}_{ed} + \boldsymbol{\Lambda}\,\mathbf{e}_{kin}\right]\right\} \tag{5.7}$$

With $\boldsymbol{\Lambda} > 0$ the gain matrix of the proportional kinematic controller(diagonal) and $\mathbf{e}_{kin} = \boldsymbol{\eta}_{ed} - \boldsymbol{\eta}_e$ the kinematic (position + orientation) error.

When the kinematic error $\mathbf{e}_{kin}$ is zero, we see that Eq. 5.7 turns into Eq. 5.6. Indeed, when the controller manages to cancel the error, the feedforward control alone drives the robot in order to continue in the tracked trajectory.

Additionally, if we replace $\boldsymbol{\nu}$ in Eq. 5.5 for the expression of $\boldsymbol{\nu}_{ad}$ from Eq. 5.7 we obtain the following dynamic error equation:

$$\dot{\mathbf{e}}_{kin} + \boldsymbol{\Lambda}\mathbf{e}_{kin} = 0 \tag{5.8}$$

which shows that the kinematic control law makes the robot converge exponentially toward the desired trajectory.

5.1.3 Dynamic level

Obviously, the desired velocity of the robot $\boldsymbol{\nu}_{ad}$ can not be correctly achieved with only kinematic control. Indeed, the robot needs to fight against the dynamic effects acting on itself in order to precisely generate this velocity. As seen in the previous chapter, these dynamic effects are a consequence of the rigid body dynamics and caused by the water surrounding the vehicle.

The dynamic layer of the controller takes in charge the compensation of all the dynamic effects on the robot and the generation of the needed torque τ_a that will make the AUV to follow $\boldsymbol{\nu}_{ad}$, that in its turn, will make the point e follow the desired trajectory.

Based on the dynamic model of the robot described in Section 4.1.2:

$$\boldsymbol{\tau} = \mathbf{M}\dot{\boldsymbol{\nu}} + \mathbf{C}(\boldsymbol{\nu})\boldsymbol{\nu} + \mathbf{D}(\boldsymbol{\nu})\boldsymbol{\nu} + \mathbf{G} \tag{5.9}$$

the following dynamic control law is created (in R_b) [162]:

$$\boldsymbol{\tau}_a = \mathbf{M}[\dot{\boldsymbol{\nu}}_{ad} + \mathbf{K_p}\mathbf{e}_{dyn}] + \mathbf{C}(\boldsymbol{\nu})\boldsymbol{\nu} + \mathbf{D}(\boldsymbol{\nu})\boldsymbol{\nu} + \mathbf{G} \tag{5.10}$$

with $\mathbf{e}_{dyn} = \boldsymbol{\nu}_{ad} - \boldsymbol{\nu}$, the dynamic error.

From the previous equation we see how all the terms of the dynamic model of the robot are present, plus a proportional ($K_p > 0$) term for error correction. If applied, the calculated torque $\boldsymbol{\tau}_a$ will compensate the hydrodynamic effects on the robot and steer the velocity $\boldsymbol{\nu}$ towards the desired velocity $\boldsymbol{\nu}_{ad}$.

If we replace $\boldsymbol{\tau}$ from Eq. 5.9 for its expression from Eq. 5.10 we obtain the following equation:

$$\dot{\mathbf{e}}_{dyn} + \mathbf{K}_p\, \mathbf{e}_{dyn} = 0 \tag{5.11}$$

Which shows that the non-linearities compensation make the AUV converge exponentially toward the desired robot velocity $\boldsymbol{\nu}_{ad}$. Given that both $\mathbf{e}_{kin}$ and $\mathbf{e}_{dyn}$ converge exponentially, the controller as a whole (kinematic and dynamic) is able of make an arbitrary point of the AUV to track a bounded space trajectory.

5.1.4 Thrust allocation

Torque computation method is capable of calculating the needed τ_a to apply on the robot in order to make the tracking point follow a desired trajectory. However, it has no information about *how* to create said wrench. In order to calculate the force on each thruster to generate $\boldsymbol{\tau}_a$ we use the thrust allocation method [163].

The thrust allocation method solves the following equation:

$$\boldsymbol{\tau}_a = \mathbf{B}\, u_p$$

Where $\mathbf{B}$ is the thrust configuration matrix, as seen in Chapter 3. $\mathbf{u}_p$ is the thrust forces vector, it contains the force generated by each thruster of the robot.

Given $\boldsymbol{\tau}_a$, we can calculate the needed force on each thruster calculating:

$$\mathbf{u}_p = \mathbf{B}^{-1}\, \boldsymbol{\tau}_a \tag{5.12}$$

Evidently, this equation is only valid if $\mathbf{B}$ is a square matrix. When this matrix is not square, its pseudo-inverse [124] is used:

$$\mathbf{u}_p = (\mathbf{B}^T\mathbf{B})^{-1}\mathbf{B}^T\,\tau_a \quad \text{for } m \geq n \tag{5.13}$$

$$\mathbf{u}_p = \mathbf{B}^T(\mathbf{B}\mathbf{B}^T)^{-1}\,\tau_a \quad \text{for } m \leq n \tag{5.14}$$

Where m and n are the number of lines and columns of $\mathbf{B}$, respectively.

5.1.5 Space reduction

The given equations are specifically valid for fully (or over) actuated robots. Indeed, in the creation of the torque $\boldsymbol{\tau}_a$, the dynamic control assumes that all the degrees of freedom in Cartesian space are actuated, which is not true for underactuated robots. A way of adapting this control method to any kind of vehicle is needed, otherwise its application will be restricted to holonomous robots only. The problem we are trying to solve here is, summing up, how to use this technique with underactuated robots.

The approach taken here will be a practical one: the robot will only try to follow the degrees of freedom of $\boldsymbol{\eta}_{ed}$ attainable by the robot given its propulsion capabilities.

An underactuated robot (incapable of creating all DOF of $\boldsymbol{\tau}_a$ independently) will only be able to follow a restricted number of trajectories. Indeed, an underactuated $\boldsymbol{\tau}_a$ will result in an underactuated $\boldsymbol{\nu}$ (it will not be able to control all its DOF independently), which in turn will not be able to generate all DOF of $\dot{\boldsymbol{\eta}}_e$ in R_0 (Eq. 5.5). The reduced actuation space reduces the reachable space of the robot, making it not controllable on all trajectories.

Evidently, this space reduction implies as well a modification on the controller equations. The matrices need to be reduced according to the actuated space and reachable space reduction. Additionally, a drift and disturbance vector have to be added to both controller laws. These vectors take into account the contribution of the non actuated DOF over the actuated ones. The first one accounts for the coupling elements in matrix $\mathbf{T}$, while the second one accounts for the efforts on the actuated space created by the non actuated DOF of $\boldsymbol{\nu}$. Indeed, the product $(\mathbf{C}+\mathbf{D})\,\boldsymbol{\nu}$ in the dynamic model will allow the non actuated terms of $\boldsymbol{\nu}$ to create efforts on the actuated DOF of $\boldsymbol{\tau}_a$, thanks to the coupling terms of these matrices. This means that even if these DOF are not controllable, it is possible to control their coupling effects in the actuated space of the robot.

To illustrate these space reductions, we will apply the computed torque technique to both studied underactuated robots: the fixed-thrusters and the vectorial-thruster RSM robot. We will make these two robots to follow a common trajectory, which will allow to compare the set up of the reduced space and analyze their characteristics.

5.1.5.1 Evaluation trajectory

The "dive in a confined space" trajectory, shown in section 4.3.2.2, will be used. It allows to evaluate the pertinence of the controller, the application to under actuated propulsive topologies and, additionally, the propulsion strategy performance in a particular mission.

To follow this trajectory, the vehicle starts moving horizontally and then dives following a vertical line. The control technique will make the AUV follow only the position of the trajectory, leaving the orientation of the vehicle uncontrolled (three DOF for task space to match the three DOF actuation space of vectored propulsion).

Both propulsive architectures will follow the same trajectory using the same control technique. However, the way of following the desired trajectory will be different for each architecture. Additionally, in order to compare the performance of the two propulsive systems, the energy consumption will be measured. This will give us a general idea of the efficiency of each strategy.

In order to follow this common trajectory, the command laws must adapt to the different propulsive topology. In the next sections we will see how.

5.1.5.2 Kinematic control law reduction

The reduced kinematic control law is described by the following equation:

$$\boldsymbol{\nu}_{ad} = \mathbf{T}^{-1}\{\mathbf{J_1^r}^{\mathbf{T}}(\boldsymbol{\eta}_2)[\dot{\boldsymbol{\eta}}_{ed} + \boldsymbol{\Lambda}(\boldsymbol{\eta}_{ed} - \boldsymbol{\eta}_e)] - [\mathbf{v}_{nc}]^T\} \tag{5.15}$$

where $[\mathbf{v}_{nc}]^T$ are the drift terms (non actuated linear motions) in R_b and $\mathbf{T}$ is the kinematic transformation matrix that brings the speed from R_b to the tracking point e.

The matrices are reduced stripping out the lines corresponding to the unused degrees of freedom of $\boldsymbol{\eta}_{ed}$ and the columns corresponding to the non actuated elements of $\boldsymbol{\tau}$ in R_b. The non actuated elements of $\boldsymbol{\tau}$ are determined by the $\mathbf{B}$ matrices of both propulsive topologies, which will determine what elements of $\boldsymbol{\nu}_{ad}$ they are capable of generating. With this information and using Eq. 5.5, we can determine the reachable space $\boldsymbol{\eta}_{ed}$.

In our first example (Fig. 5.3) the robot can generate *surge*, *heave*, *roll* and *yaw*. With these four degrees of freedom and with the position of e:

$$\vec{r_e} = [0.3,\ 0.0,\ 0.1]^T \tag{5.16}$$

it is capable to follow the position of $\boldsymbol{\eta}_{ed}$. Additionally, it can follow the orientation ϕ, which will be used in our calculations as a way of keeping a squared $\mathbf{T}$ matrix.

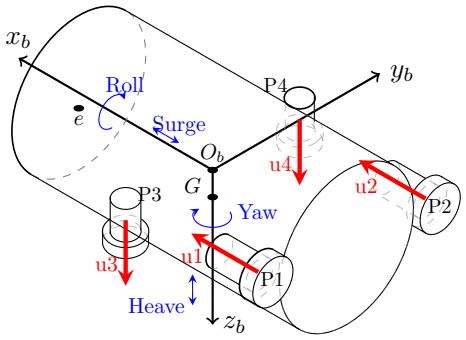

Figure 5.3: RSM robot with fixed propulsion topology.

Similarly, the vectorial example (Fig. 5.4) can generate *surge*, *pitch* and *yaw*. This makes it also able of reaching the desired position on $\boldsymbol{\eta}_{ed}$.

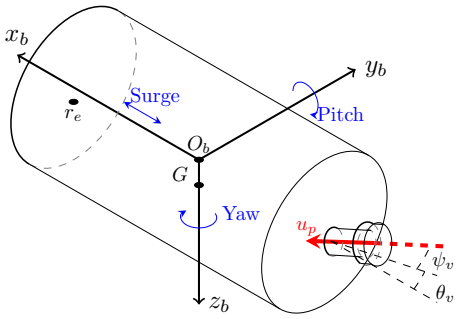

Figure 5.4: RSM robot with vectorial thruster.

A list of the reduced elements for the propulsive topologies are the following:

Fixed propulsion	Vectorial propulsion
$\boldsymbol{\nu}_{ad} = [u\ w\ p\ r]^T$	$\boldsymbol{\nu}_{ad} = [u\ q\ r]^T$
$\mathbf{v}_{nc} = [0\ v\ 0\ 0]^T$	$\mathbf{v}_{nc} = [0\ v\ w]^T$
$\boldsymbol{\eta}_{ed} = [x_{ed}\ y_{ed}\ z_{ed}\ \phi_{ed}]^T$	$\boldsymbol{\eta}_{ed} = [x_{ed}\ y_{ed}\ z_{ed}]^T$
$\mathbf{T} = \begin{bmatrix} 1 & 0 & 0 & -\epsilon_y \\ 0 & 0 & -\epsilon_z & \epsilon_x \\ 0 & 1 & \epsilon_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$	$\mathbf{T} = \begin{bmatrix} 1 & \epsilon_z & -\epsilon_y \\ 0 & 0 & \epsilon_x \\ 0 & -\epsilon_x & 0 \end{bmatrix}$
$\boldsymbol{\Lambda} = 10 * \mathbb{I}_{4\times 4}$	$\boldsymbol{\Lambda} = 10 * \mathbb{I}_{3\times 3}$

5.1.5.3 Dynamic control law reduction

Similarly, the dynamic control law is reduced using the following equation:

$$\boldsymbol{\tau}_a = \mathbf{M^r}[\dot{\boldsymbol{\nu}_{ad}} + \mathbf{K_p}e] + \mathbf{C^r}(\nu)\nu + \mathbf{D^r}(\nu)\nu + \mathbf{G}^r + \mathbf{d}(\nu) \tag{5.17}$$

Were $\mathbf{M^r}$, $\mathbf{C^r}$, $\mathbf{D^r}$, $\mathbf{G}^r$ are the reduced dynamics matrices and $\mathbf{d}(\nu)$ is the disturbance vector. For the dynamic control law, the matrix reduction is made only taking into account the reduced actuated space. Indeed, in Eq. 5.17, the reduction in the propulsive capabilities of $\boldsymbol{\tau}_a$ only affects the generated DOF of $\boldsymbol{\nu}$. The reduction is made then stripping out the lines and columns corresponding to the non actuated DOF of $\boldsymbol{\tau}_a$.

For the fixed propulsion the matrices are :

$$\mathbf{M^r} = \begin{bmatrix} m + X_{\dot{u}} & 0 & 0 & 0 \\ 0 & m + Z_{\dot{w}} & 0 & 0 \\ 0 & 0 & I_{bx} + K_{\dot{p}} & 0 \\ 0 & 0 & 0 & I_{bz} + N_{\dot{r}} \end{bmatrix}$$

with I_{bx}, and I_{bz} the diagonal components of the inertia matrix.

$$\mathbf{C^r} = \begin{bmatrix} 0 & mq & 0 & Y_{\dot{v}}v \\ -mq & 0 & -Y_{\dot{v}}v & 0 \\ 0 & Y_{\dot{v}}v & 0 & M_{\dot{q}}q - 0.6113q \\ -Y_{\dot{v}}v & 0 & 0.6113q - M_{\dot{q}}q & 0 \end{bmatrix}$$

$$\mathbf{D^r} = \begin{bmatrix} -X_{u|u|}|u| & 0 & 0 & 0 \\ 0 & -Z_{w|w|}|w| & 0 & 0 \\ 0 & 0 & -K_{p|p|}|p| & 0 \\ 0 & 0 & 0 & -N_{r|r|}|r| \end{bmatrix}$$

$$\mathbf{J_1^r} = \begin{bmatrix} c\psi c\theta & c\psi s\phi s\theta - c\phi s\psi & s\phi s\psi + c\phi c\psi s\theta & 0 \\ c\theta s\psi & c\phi c\psi + s\phi s\psi s\theta & c\phi s\psi s\theta - c\psi s\phi & 0 \\ -s\theta & c\theta s\phi & c\phi c\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{K_p} = \nVdash_{4\times 4}$$

$$\mathbf{G}^r = \mathbf{J_1^r}(\eta_2)[0 \ mg \ 0 \ \rho g \nabla]^T$$

$$\boldsymbol{\tau}_a = [X \ Z \ K \ N]^T$$

And for the vectorized propulsion :

$$\mathbf{M^r} = \begin{bmatrix} m + X_{\dot{u}} & \lambda\, m & 0 \\ \lambda\, m & I_{by} + M_{\dot{q}} & 0 \\ 0 & 0 & I_{bz} + N_{\dot{r}} \end{bmatrix}$$

with I_{by} and I_{bz} the diagonal components of the inertia matrix and λ the component on z_b of distance between the center of gravity and the center of buoyancy (placed in O_b in our robot).

$$\mathbf{C_a^r} = \begin{bmatrix} 0 & -Z_{\dot{w}}w & Y_{\dot{v}}v \\ Z_{\dot{w}}w & 0 & 0.0936p - K_{\dot{p}}p \\ -Y_{\dot{v}}v & K_{\dot{p}}p - 0.0936p & 0 \end{bmatrix}$$

$$\mathbf{D_a^r} = \begin{bmatrix} -X_{u|u|}|u| & 0 & 0 \\ 0 & -M_{q|q|}|q| & 0 \\ 0 & 0 & -N_{r|r|}|r| \end{bmatrix}$$

$$\mathbf{J_1^r} = \begin{bmatrix} c\psi c\theta & c\psi s\phi s\theta - c\phi s\psi & s\phi s\psi + c\phi c\psi s\theta \\ c\theta s\psi & c\phi c\psi + s\phi s\psi s\theta & c\phi s\psi s\theta - c\psi s\phi \\ -s\theta & c\theta s\phi & c\phi c\theta \end{bmatrix}$$

$$\mathbf{K_p} = \nVdash_{3\times3}$$

$$\mathbf{G}^r = \mathbf{J_1^r}(\eta_2)[0\ 0\ \rho g \nabla]^T$$

$$\boldsymbol{\tau}_a = [X\ M\ N]^T$$

5.1.5.4 Thrust allocation reduction

The next step in controlling the vehicle is to generate $\boldsymbol{\tau}_a$ with the thrusters. To do so, the AUV thrusters must produce an adequate thrust to comply with:

$$\boldsymbol{\tau}_a = \mathbf{B}\mathbf{u}_p \tag{5.18}$$

To calculate the needed propulsion forces in each thruster, equation (5.18) must be inverted. We need to reduce the $\mathbf{B}$ matrix for each propulsive system to their respective actuated space. The reduction of the thrust configuration matrix $\mathbf{B}_r$ is obtained, as for the dynamic control law matrices, by stripping out the lines of $\mathbf{B}$ corresponding to the non actuated elements pf $\boldsymbol{\tau}$. The amount of columns of this matrix will be determined by the quantity of thrusters used in the propulsive topology and therefore will not be reduced.

In this case, given that the $\mathbf{B}_r$ matrices are square thanks to the reduction, there is no need of using the pseudo-inverse.

For the fixed propulsion system, $\mathbf{B}_r$ is:

$$\mathbf{B}_r = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & -P_{3y} & -P_{4y} \\ -P_{1y} & -P_{2y} & 0 & 0 \end{bmatrix}$$

$$\mathbf{u}_p = \mathbf{B}_r^{-1}\tau_{\mathbf{a}} = [u_1, u_2, u_3, u_4]^T \tag{5.19}$$

For the vectorial propulsion, we can rearrange the $\mathbf{B}$ (Fig. 4.3) and turn it into a matrix. With this change, in the equation 5.18, $\mathbf{u}_p$ contains the components of the thrust vector in R_b (and not the module u_1). This is useful in our calculations since it turns $\mathbf{B}$ from a vectorial to a matrix expression, more adapted to calculate $\mathbf{u}_p$ using the inverse :

$$\mathbf{B}_r = \begin{bmatrix} 1 & 0 & 0 \\ P_{1z} & 0 & -P_{1x} \\ -P_{1y} & -P_{1x} & 0 \end{bmatrix}$$

$$\mathbf{u}_p = \mathbf{B}_r^{-1}\tau_{\mathbf{a}} = [u_x, u_y, u_z] \tag{5.20}$$

$\mathbf{u}_p$ in this configuration gives the components of the thrust on x_b, y_b and z_b.

From the components of $\mathbf{u}_p$, we calculate u_1, the force of the thruster and the reconfiguration angles ψ_1 and θ_1 as follows:

$$u_1 = ||\mathbf{u}_p|| = \sqrt{u_x^2 + u_y^2 + u_z^2}$$

$$\theta_1 = -\arcsin(\frac{u_z}{u_1}) \qquad \psi_1 = \arctan(\frac{u_y}{u_x})$$

5.1.6 Numerical validation

5.1.6.1 Mission simulation

A simulation for each propulsive strategy has been carried out. Since the fixed thruster configuration has its center of gravity between vertical thrusters, this task is straightforwardly performed. Consequently, the AUV dives along the vertical line, keeping a zero pitch angle (black dotted line if Fig. 5.5).

Because of its actuation, the AUV equipped with vectorial propulsion needs to orient itself vertically (blue dotted line in Fig. 5.5). In order to perform such a motion, the computed torque method controls the pitch angle of the propeller θ_1. Fig. 5.5 shows the paths followed by the AUVs equipped with fixed and vectorial propulsion. We can see how the vectorial AUV diverts from the desired trajectory since it must rotate. After the abrupt change of direction the AUV equipped with the vectorial propulsion manages to recover the vertical trajectory and ends up converging towards it.

Figures 5.6 and 5.7 show the control inputs for the fixed and vectorial systems respectively. We can observe that the fixed control inputs stabilize for horizontal and vertical thrusters rather quickly. Meanwhile, the vectorial system thruster pitch (θ_1) varies greatly during the diving. This can be explained by the fact that the thruster is behind the tracking point and the AUV center, which causes the control of the vehicle to be complicated.

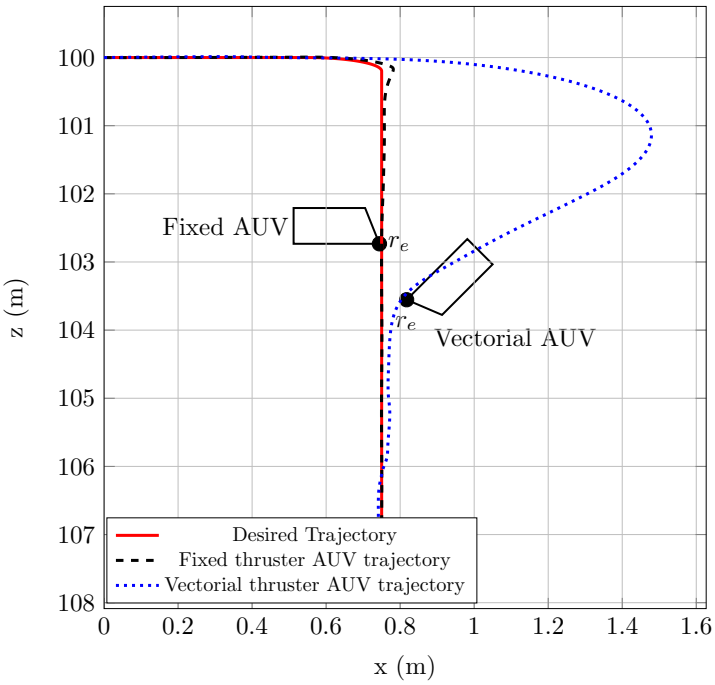

Figure 5.5: Trajectory followed by the AUV for fixed and vectorial propulsion.

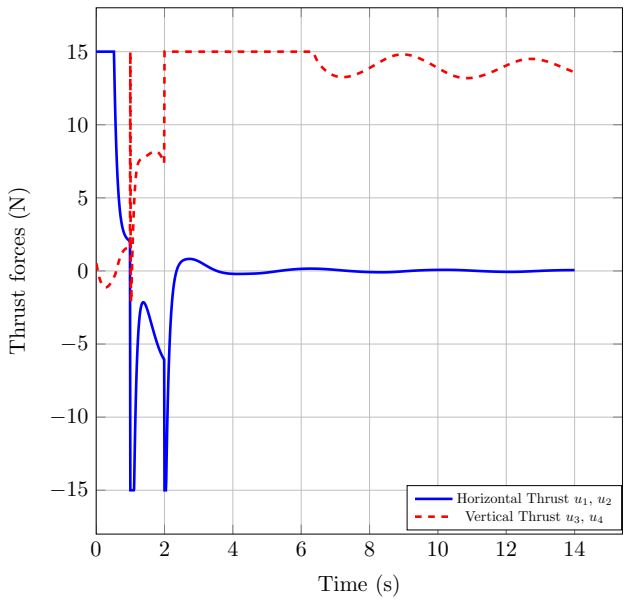

Figure 5.6: Control input for the fixed propulsion strategy (FP).

In order to evaluate the performance of the studied propulsive strategy we will use the convergence of the vehicle towards the desired trajectory and the energetic efficiency.

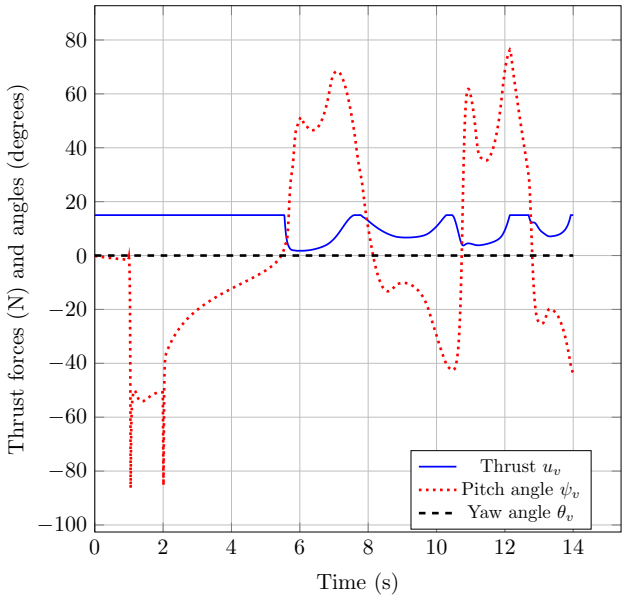

Figure 5.7: Control input for the vectorial propulsion strategy (VP).

5.1.6.2 Results

To evaluate the convergence, the quadratic tracking error norm is analyzed:

$$||e_{\text{pos}}|| = \sqrt{(x_e - x)^2 + (y_e - y)^2 + (z_e - z)^2} \tag{5.21}$$

Where x_e, y_e and z_e are the position coordinates of $\boldsymbol{\eta}_{ed}$.

Fig. 5.8 shows this error for the fixed and vectorial propulsive systems. The error for the fixed thruster system converges rapidly towards zero. The vectorial thruster AUV error takes longer to converge due to the sharp turn of the desired trajectory, which forces the AUV to change its orientation, before converging again.

To evaluate the energetic efficiency of the systems we calculate the electrical instantaneous power for each thruster:

$$P = V\,I \tag{5.22}$$

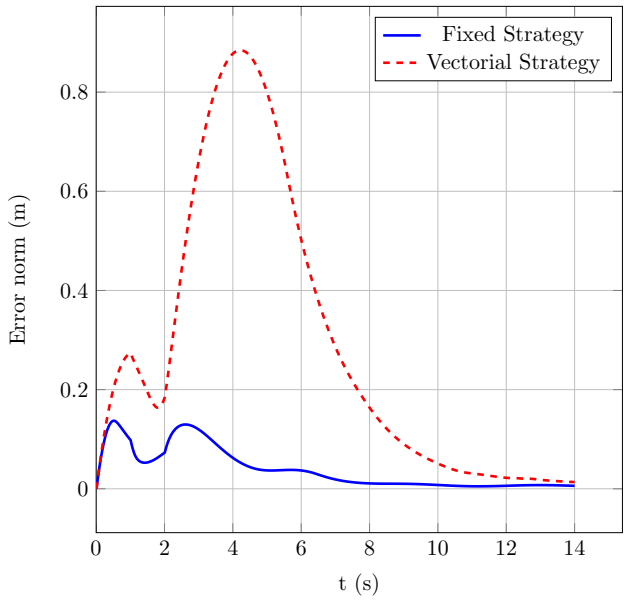

Figure 5.8: Position error quadratic norm.

Where P is the instantaneous power (Watt). V is the thruster tension (Volt) and A is the motor current (Ampere), which were obtained with the steady-state thruster model [164]:

$$V = \frac{R}{K_M} u + \kappa \operatorname{sign}(u) \sqrt{\frac{|u|}{K_T}}$$

and

$$I = \frac{u}{K_M}$$

The total power is obtained by adding each thruster power (Fig. 5.12). The energy consumption is calculated by integrating over time. For the fixed propulsion system the consumed energy is 2398 J and for the vectorial system 922 J. This comparison indicates that the vectorial thruster was 2.6 times more energy efficient for this mission. This is because the vectorial propulsion robot flips down in order to comply with the task, allowing the use of a single thruster instead of four. Indeed, diving head down generates less hydrodynamic drag, which makes the vectorial propulsion more efficient for this kind of trajectories, even if it generates more transient errors (Fig. 5.8).

The used control technique has been successfully adapted to two different propulsive technologies in an automatic way, which shows the flexibility of

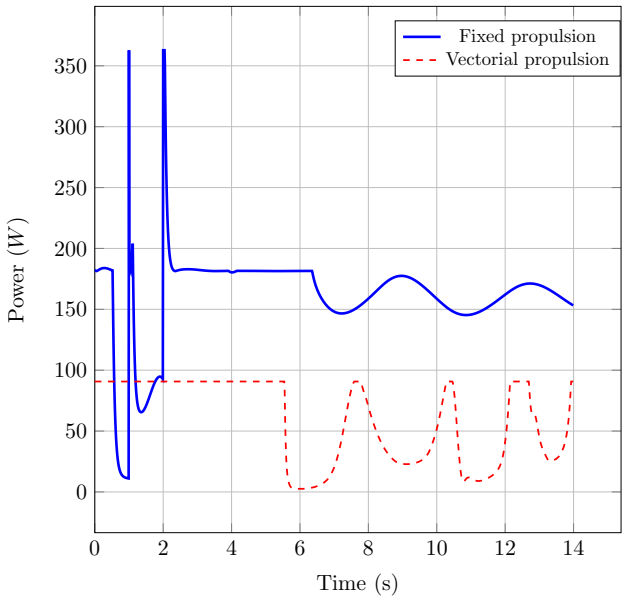

Figure 5.9: Total power consumption for fixed and vectorial propulsions.

the method and its applicability to a great number of robots. These features are particularly interesting in the optimization of the capabilities of underwater robots, since they allow to analyze their performance using a common benchmark.

5.2 Controllability

The computed torque method offers good adaptation capabilities, however, it does not mean that it can be applied to every robot to follow any trajectory. Indeed, the space reduction, which is the key to the the adaptation of the method to different types of robot, is also a limiting element of the method.

In the application of the control method, two matrices play a key role in the success of the controller: the **T** and **B** matrices. The controller will be able to follow the desired trajectory depending on the shape and conditioning of these matrices.

5.2.1 T matrix

The **T** matrix, which intervenes in the kinematic control law, determines which velocities will be created in the tracking point e. The velocities of this point, are a function of the robot velocity at the origin of R_b, and depend on

the rigid body kinematics :

$$\nu_e = \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 & \epsilon_z & -\epsilon_y \\ 0 & 1 & 0 & -\epsilon_z & 0 & \epsilon_x \\ 0 & 0 & 1 & \epsilon_y & -\epsilon_x & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}}_{\mathbf{T}} \underbrace{\begin{bmatrix} u \\ v \\ w \\ p \\ q \\ r \end{bmatrix}}_{\nu} \tag{5.23}$$

The matrix $\mathbf{T}$ expresses then the rigid body kinematics of the robot. From its shape we can clearly see how the linear velocities on e (in R_b) are created by the linear and angular velocities of the robot on R_b. This information is used by the kinematic control, through the inverse of $\mathbf{T}$ (Eq. 5.7), to define which degrees of freedom of R_b will be needed to follow a given trajectory.

In a fully actuated robot, all the degrees of freedom in R_b (and so, in R_0) are actuated. Therefore, no reduction of the matrix $\mathbf{T}$ is needed and all possible velocities of e can be created. Mathematically, it means that for any value of ϵ_x, ϵ_y and ϵ_z, $\mathbf{T}$ will be invertible. However, when we use the same control method on an underactuated robot, things are different.

In an underactuated case, we will reduce the columns of the $\mathbf{T}$ matrix stripping out the ones corresponding to the non-actuated velocities in R_b. The lines of the $\mathbf{T}$ matrix will be reduced as well, taking into account that in order to calculate the inverse we need to obtain a square matrix. The deleted lines of $\mathbf{T}$ will be those corresponding to the unreachable and undesired velocities in e.

The reduced $\mathbf{T}$ matrix, even if it is square, will not always be invertible. The invertibility of the matrix will depend, in this case, greatly of the position of e. We can use the reduced $\mathbf{T}$ matrix of the fixed-RSM robot to exemplify this:

$$\begin{bmatrix} u_e \\ v_e \\ w_e \\ p_e \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & -\epsilon_y \\ 0 & 0 & -\epsilon_z & \epsilon_x \\ 0 & 1 & \epsilon_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} u \\ w \\ p \\ r \end{bmatrix} \tag{5.24}$$

Here, the actuated degrees of freedom are u, w, p and r, whereas the wanted and reachable degrees of freedom are u_e, v_e, w_e and p_e. These degrees of freedom are those of the desired trajectory projected in R_b. From the reduced matrix we see that its invertibility will depend on the position of e. Indeed, if the values of ϵ_x and ϵ_z are zero, the inversion of $\mathbf{T}$ is no longer possible, making the control unfeasible.

The fact that matrix $\mathbf{T}$ is invertible, does not mean, however, that the control will be successful. For some configurations, even if the coordinates of e are non-zero, the controller is still not able to converge toward the desired trajectory. We obtain this behavior when elements of $\dot{\boldsymbol{\eta}}_e$ (Eq. 5.5) are generated only thanks to the laws of solid kinematics. In our example, it is the case of the second line of the reduced $\mathbf{T}$ matrix, where angular velocities p (*roll*) and r (*yaw*) are used to create linear velocity v_e (*sway*) in R_b. In R_0, this will make e describe an arc of circle, which can be used to follow a linear trajectory only for a brief period of time, after which the linear velocity in R_b is no longer useful in R_0. It means that the linear velocities created thanks to a rotational velocity may be only helpful to create small and brief corrections in the trajectory of e.

To summarize, the reduction of the lines of $\mathbf{T}$ (given that the columns are determined by the actuated space) to define the reachable space is ruled by two factors. The first one is the invertibility of this matrix, it has to be squared and with lines and columns linearly independent, which depends on the position of e. Secondly, the creation of linear velocities by the rotation motions (p, q and r) is limited. This means that the DOF of the operational space corresponding to these linear motions have to have a reduced variation.

Several possibilities to determine the controllability of AUVs on trajectories have been considered throughout this work. The kinematic compliance of the AUV with the trajectory can be checked by anticipation method. This method consists of placing successively on each trajectory position/orientation and testing the compatibility between task and actuated space (from Eq. 5.5). This method has been tried with some success but has not reached its conclusion to be mentioned here.

5.2.2 B matrix

The $\mathbf{B}$ matrix is also a critic factor in the success of the controller. It is in charge of determining *how* the $\boldsymbol{\tau}_a$ control wrench will be created. Obviously, for a given $\boldsymbol{\tau}_a$ that the robot is not capable to generate, the controller will not be able to make e converge toward the desired trajectory. The thrust allocation method is the stage to the control wrench generation. This means that even if the controller correctly calculates $\boldsymbol{\tau}_a$, an incorrect thrust allocation can create a failing control.

As seen in Section 5.1.4, the thrust allocation method uses the inverse (or pseudo-inverse) of the $\mathbf{B}$ matrix in order to calculate the force that each thruster must generate. This establishes the first constraint of this matrix: it must be invertible. A non invertible matrix means that the robot is not capable of creating, with its propulsive topology, the demanded wrench $\boldsymbol{\tau}_a$.

The $\mathbf{B}$ matrix, when multiplied by the $\mathbf{u}_p$ vector, must generate the elements of $\boldsymbol{\tau}_a$ needed to follow the trajectory. If the product $\mathbf{B}\,\mathbf{u}_p$ fails to do so, the controller will not be able to achieve its mission. As explained before, this matrix represents the propulsive configuration of the robot, each column corresponds to the contribution of a thruster to the creation of $\boldsymbol{\tau}_a$. The number of columns of this matrix represents then, the number of thrusters.

Each thruster has its own coordinate frame, the convention used to define the orientation of this frame with regard to R_b is the same as the one used to define the orientation of R_b with regard to R_0 (Eq. 5.25). This Euler ZYX convention uses three successive rotations: ψ_i around $\vec{z}_b$, θ_i around $\vec{y}_b'$ and ϕ_i around $\vec{x}_b''$. The position of each thruster in R_b is given by a vector $\vec{\mathbf{P}}_i = [P_{ix},\ P_{iy},\ P_{iz}]^T$. With these elements we can determine the homogeneous transformation matrix :

$$\mathbf{H}_i = \begin{bmatrix} c\theta_i\, c\psi_i & -s\psi_i\, c\phi_i + c\psi_i\, s\theta_i\, s\phi_i & s\psi_i\, s\phi_i + c\psi_i\, c\phi_i\, s\theta_i & P_{ix} \\ c\theta_i\, s\psi_i & c\psi_i\, c\phi_i + s\theta_i\, s\theta_i\, s\psi_i & -c\psi_i\, s\phi_i + s\theta_i\, s\psi_i\, c\phi_i & P_{iy} \\ -s\theta_i & c\theta_i\, s\phi_i & c\theta_i\, c\phi_i & P_{iz} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.25}$$

Where $c(\cdot)$ and $s(\cdot)$ represent trigonometric functions cos and sin respectively.

The force of each thruster follows, by convention, the $\vec{x}$ axis of its frame. This means that when using the rotation matrix (Eq. 5.25) to calculate the components of this force in R_b, only the first column of the rotation matrix is used. The orientation of the thrust vector is then independent of ϕ_i.

Taking this into account, each thruster is represented in $\mathbf{B}$, as follows:

$$\underbrace{\begin{bmatrix} X \\ Y \\ Z \\ M \\ N \\ K \end{bmatrix}}_{\boldsymbol{\tau}_a} = \underbrace{\begin{bmatrix} \dots & c\theta_i c\psi_i & \dots \\ \dots & c\theta_i s\psi_i & \dots \\ \dots & -s\theta_i & \dots \\ \dots & -s\theta_i\, P_{iy} - c\theta_i s\psi_i\, P_{iz} & \dots \\ \dots & c\theta_i c\psi_i\, P_{iz} + s\theta_i\, P_{ix} & \dots \\ \dots & c\theta_i s\psi_i\, P_{ix} - c\theta_i c\psi_i\, P_{iy} & \dots \end{bmatrix}}_{\mathbf{B}} \underbrace{\begin{bmatrix} \vdots \\ u_i \\ \vdots \end{bmatrix}}_{\mathbf{u}_p} \tag{5.26}$$

From the above equation, we see that if for some reason one of the lines of the thrusters contributions are zero, the propulsion system will not be able to actuate the corresponding degree of freedom. Such a situation may arise when the orientation of the thruster is perpendicular to the degree of freedom in question or when the moment arm is inexistent.

Clearly, the fact that one thruster can generate components in all six degrees of freedom does not mean it can effectively control them. Indeed, more important than generating forces in a degree of freedom is the capability of independently controlling it. In order to generate independently several elements of $\boldsymbol{\tau}_a$ we need to add several thrusters (columns) as well. These thrusters must be carefully positioned and orientated to avoid creating unwanted redundancies and to obtain an efficient contribution to the creation of $\boldsymbol{\tau}_a$.

As seen from the creation of $\mathbf{T}$ and $\mathbf{B}$, the initialization of these matrices could be a complicated task. From $\mathbf{T}$, the positioning of the tracking point e determines the correct calculation of the kinematic law. From $\mathbf{B}$, the quantity, positioning and orientation of the thrusters will dictate if the dynamic control law is applicable or not. This is due to the fact that, after the space reduction, the (pseudo) inverse of $\mathbf{B}_r$ must exist. If it does, then $\mathbf{u}_p$ can be calculated to generate $\boldsymbol{\tau}_a$, otherwise the robot is not capable of generating $\boldsymbol{\nu}_{ad}$ and therefore can not follow $\boldsymbol{\eta}_{ed}$.

Additionally, even for viable matrices, it is a hard task to clearly determine which configuration will be the more efficient to follow the desired trajectory. Indeed, in order to correctly set up a robot, we need to know where to place e, how many thrusters we need to use, where to position them and how to orientate then. Given the immense number of possibilities, finding the optimal configuration of a robot (for a given task) is a daunting task (likely to be the Non-Polynomial complete kind of problems). The exhaustive research of all the possibilities not being practical, a methodological cost-effective approach is needed.

5.3 Kalman filter

Once the control method has been established, we need to find a way to properly feed it with the information it needs to operate. Indeed, in order to work, the computer torque method needs to know the values of positions and velocities η_1, η_2, ν_1 and ν_2. This information is, however, not always directly available.

The RSM robot, for instance, is equipped with a depth sensor and an inertial measurement unit (IMU). These two sensors provide the following data: $\dot{\nu}_1$, ν_2 and η_2 (but not $\boldsymbol{\eta}_1$ and $\boldsymbol{\nu}_1$), which is not sufficient to the controller. Since the control method needs the complete state of the robot to properly work, the remaining data needs to be at least estimated. The chosen method of estimation is the widely used Kalman filtering.

5.3.1 Principle

A Kalman filter is a recursive algorithm that, using measurements over time and a dynamic model, generates an optimal estimation, from a statistical point of view, of unknown variables. The application of the Kalman filter assumes that both the system and the measurements are affected by Gaussian noise and that the state representation of the system is linear [124]. Our problem being non-linear, we have to use an algorithm adapted to our situation. Said algorithm is the Extended Kalman Filter (EKF) [124].

5.3.2 Extended Kalman filter

The extended Kalman filter is a version of the Kalman filter which extends its application to systems in which the state model and/or the observation model are non-linear.

In order to use this filter to estimate the state vector, the state model is described in state-space notation by a non-linear vector:

$$\dot{x}(t) = f[x(t), u(t), v(t), t] \tag{5.27}$$

Where $f[\cdot, \cdot, \cdot, \cdot]$ is a non-linear mapping of state and control input into state transition. $x(t)$ is the state vector, $u(t)$ is the control input, $v(t)$ is a variable describing the uncertainty in the evolution of the state caused by non modeled phenomena (Gaussian noise) and t is the time.

Similarly, the observation model is described in state-space notation as follows:

$$z(t) = h[x(t), u(t), w(t), t]$$

Where $h[\cdot, \cdot, \cdot, \cdot]$ is also a non-linear mapping of state and control input to observations and $w(t)$ is a variable describing uncertainty in the observation caused by sensor limitations (Gaussian noise).

This filter is, however, mostly used in discrete-time, which requires a discrete-time version of the models:

$$x(k) = f[x(k-1), u(k), v(k), k] \tag{5.28}$$
$$z(k) = h[x(k), w(k)] \tag{5.29}$$

Where k is the instant of calculation. The estimation of the state made by the EKF proceeds in two stages:

Prediction. In which a prediction $\hat{x}(k|k-1)$ of the state at the instant k and its covariance $P(k|k-1)$ are calculated. These predictions are based on all the information until the time $k-1$ and are calculated in the following way:

$$\begin{aligned}\hat{x}(k|k-1) &= f[\hat{x}(k-1|k-1), u(k)] \\ P(k|k-1) &= \nabla f_x(k-1)P(k-1|k-1)\nabla^T f_x(k-1) \\ &\quad + Q(k-1)\end{aligned}$$

where $\hat{x}(k-1|k-1)$ and $P(k-1|k-1)$ are an estimate of the state $x(k-1)$ and covariance $P(k-1)$ at the time $k-1$ based on all the observations made up to and including $k-1$. $Q(k-1)$ is a known covariance and ∇f_x is the Jacobian of f.

Update. In which an observation $z(k)$ is made at the time k and the updated estimate $\hat{x}(k|k)$ of the state $x(k)$ along with the updated estimate covariance $P(k|k)$ is calculated as follows:

$$\begin{aligned}\hat{x}(k|k) &= \hat{x}(k|k-1) + W(k)\{z(k) - h[\hat{x}(k|k-1)]\} \\ P(k|k) &= P(k|k-1) - W(k)S(k)W^T\end{aligned}$$

where

$$W(k) = P(k|k-1)\nabla^T h_x(k)S^{-1}(k)$$

and

$$\begin{aligned}S(k) &= R(k) \\ &\quad + \nabla h_x(k)P(k|k-1)\nabla^T h_x(k)\end{aligned}$$

with ∇h_x the Jacobian of h and R the known covariance of the measurement noise.

5.3.3 Application

To apply this estimation method to our robot, we take into account that Eq. 5.29 can also be written as:

$$x(k) = f[x(k-1), u(k), k] + v_k$$
$$z(k) = h[x(k)] + w_k$$

where $f[\cdot,\cdot,\cdot]$, is the discretized model of the underwater robot. v_k and w_k are the process noise and the measurement noise (both Gaussian) and $z(k)$ is the measurement affected by the noise w_k.

For our robot, the discretized model can be calculated as:

$$f[x(k-1), u(k), k] = \boldsymbol{\nu}_{k-1} + Te\left\{\mathbf{M}^{-1}\left[\boldsymbol{\tau}_{k-1} - (\mathbf{C}+\mathbf{D})\,\boldsymbol{\nu}_{k-1} - \mathbf{G}\right]\right\} \quad (5.30)$$

where T_e is the sample time, and $k-1$ indicates the instant of time prior to the calculation instant k. Matrices $\mathbf{C}$ and $\mathbf{D}$ are a function of $\boldsymbol{\nu}_{k-1}$, but for the sake of clarity in the expression, this has not been written.

The observation model $z(k)$ has the elements given by the IMU, and depth sensor. Given that the IMU measures $\boldsymbol{\eta}_2$, $\boldsymbol{\nu}_2$ and $\dot{\boldsymbol{\nu}}_1$, we can integrate this last vector in order to obtain $\boldsymbol{\nu}_1$:

$$\boldsymbol{\nu}_1 = \boldsymbol{\nu}_{1\,(k-1)} + Te\,\dot{\boldsymbol{\nu}}_{1\,(k-1)} \quad (5.31)$$

The observation model will be then:

$$z(k) = \begin{bmatrix} u_k \\ v_k \\ w_k \\ z_k \\ p_k \\ q_k \\ r_k \\ \phi_k \\ \theta_k \\ \psi_k \end{bmatrix} + w_k \quad (5.32)$$

The Jacobians used in the calculation of the prediction and update stage are calculated using a complex step differentiation method, as described in [165, 166].

In order to test the kalman filter in our robot, we will ask the controller to follow a spiral trajectory:

$$\begin{cases} x_e(k) &= 0.5\,\cos(v_{spiral}\cdot k) \\ y_e(k) &= 0.5\,\cos(v_{spiral}\cdot k) \\ z_e(k) &= 1 + v_{spiral}\cdot k \end{cases} \quad (5.33)$$

Where $v_{spiral} = 0.5\,m/s$ is a constant speed, used as a parameter to calculate the reference trajectory. $x_e(k)$, $y_e(k)$ and $z_e(k)$ are the coordinates of the spiral in R_0 at the instant k.

The RSM robot, equipped with the fixed thruster topology will be used. After applying the corresponding space reduction to the control laws, the controller will use the values of $\hat{x}(k|k)$ to determine the thrust vector $\mathbf{u}_p$.

Figure 5.10 shows the results of the estimation. In this figure, we can see the components of the state ν_1, ν_2, η_1 and η_2 as well as their estimated counterparts, designated with the symbol $(\hat{\cdot})$.

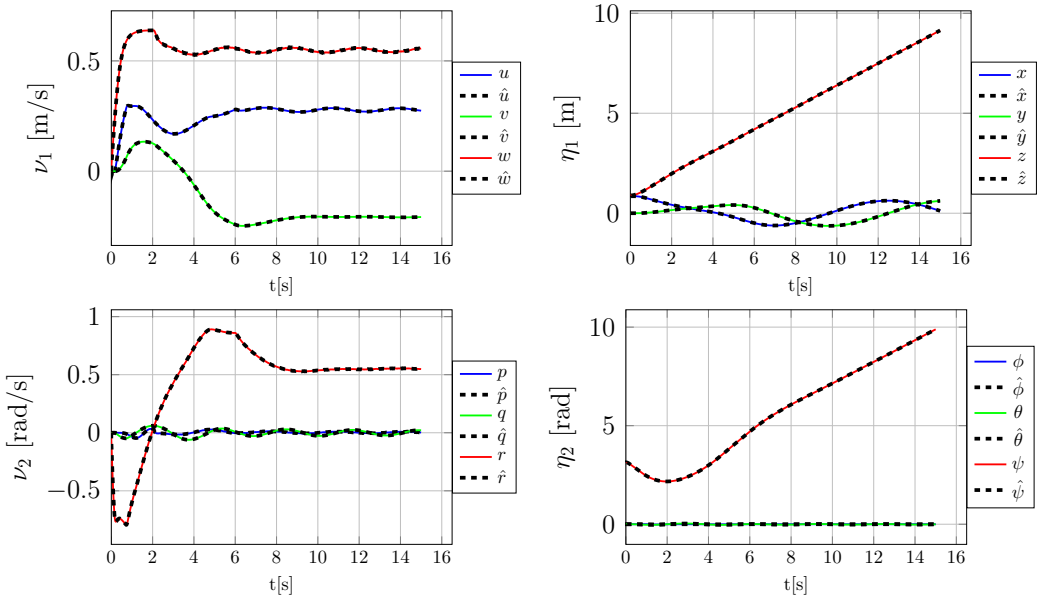

Figure 5.10: Extended Kalman filter estimation. Solid lines represent the real states and dotted ones represent the estimation.

We can see how during the time of the mission, the estimated values match the ones of the state. This allows the controller to converge towards the desired trajectory (Fig. 5.11) based solely in the estimated state, which validates the efficacy of the estimation method as well as its pairing with the controller.

5.4 Validation

Once we have determined the techniques to control and estimate the degrees of freedom of the robot, we need to verify if they work properly on the vehicle. Indeed, even if the techniques work in simulations using higher level

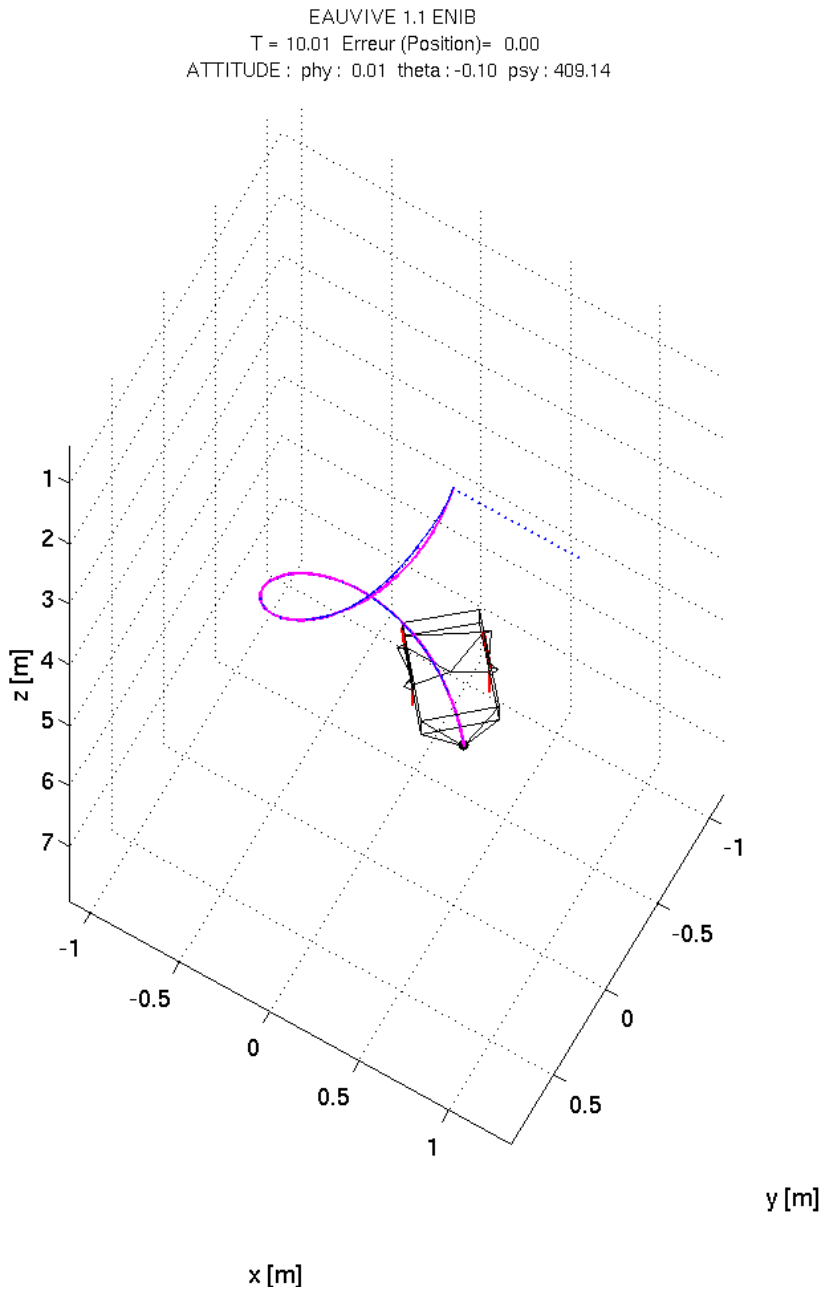

Figure 5.11: Test trajectory for EKF (refence trajectory: magenta, actual trajectory: blue).

programming languages such as MATLAB, we need to test it on the real hardware of the robot. This will allow us to validate the feasibility of these techniques in more realistic conditions.

5.4.1 ROS programming

The system used to program the control and estimations techniques is ROS (*Robot Operating System*) [167]. ROS consists of a series of software frameworks that facilitate the creation of software for robots. This system provides capabilities typical of operating systems such as software abstraction, libraries, package managing and device drivers.

Programming using the ROS framework can be done through C++ and Python languages. This is a great advantage, since C++ is a language renowned for its efficiency, which is convenient for the not so powerful hardware of our robot.

The programs created within the ROS framework, as well as the data flux are grouped in the two following categories:

- Nodes: these are the software written by the user. These programs typically take data from the robot sensors (or other nodes), process it and produce new useful data.
- Topics: they transport the information created by nodes. The nodes asynchronously publish (write) and retrieve data from these entities.

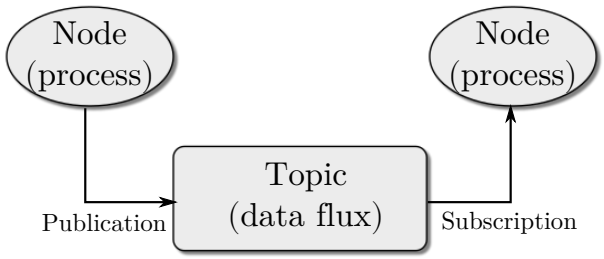

Figure 5.12: ROS minimal structure example.

5.4.2 Control architecture

The control architecture programmed using ROS is the one shown in Fig. 5.13. It consist on several nodes (represented by an ellipse) which are in

charge of manipulating data. On the left we can see two nodes called *Submarine*, these nodes are interchangeable and represent a simulation of the underwater vehicle or the real robot. In each case, the data created by these nodes are the ones of an (IMU) and a depth sensor: $\dot{u}$, $\dot{v}$, $\dot{w}$, p, q, r, ϕ, θ, ψ and z.

The IMU data and the depth of the robot are used by a node called *Submarine_Localization*, in charge of using the Kalman filter in order to estimate η_1, η_2, ν_1 and ν_2. A node called *Path planner_Controller*, calculates the reference trajectory to follow. It determines in each moment where the robot should be.

The data created by *Localization* and *Path planner* are fed to a node called *Submarine_Controller*, which is in charge of calculating the force on each thruster in order to follow the desired trajectory (applying the torque computation method of section 5.1). This node contains both the computer torque controller and thrust allocation methods.

Finally, the calculated forces on each thruster are taken by the *Submarine* node. Which applies the forces to the simulation or on the real robot.

The last node, called *UWSim* is there only to provide a 3D visualization of the robot, therefore, it does not intervene in other processes of the robot.

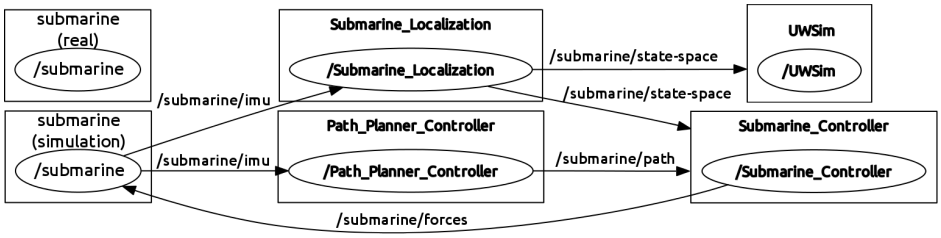

Figure 5.13: Control architecture in ROS of the RSM Robot.

5.4.3 Software-in-the-loop

Before applying the control method, a series of steps need to be taken. These steps, called "*-in-the-loop*" methods, are necessary to test and assure the viability of the control technique before using it in the real vehicle.

The preliminary method we obtained with Matlab simulation in the previous sections is sometimes called "Model in the Loop (MIL)" but it is not a true test loop since it uses high-level language programs, not representative of what happens in embedded systems. Indeed, in the process of adapting the control method from a high level language as Matlab to the real robot

there might be errors. These errors, due to specificities of the programming language or the hardware need to be addressed in a systematic way.

The first of these methods, called "*software-in-the-loop*" (SIL) allows to test and debug the code written for the microprocessor of the robot. It uses a simulator of the real vehicle and the controller written in the target language, both running on the development (not embedded) computer (Fig. 5.14).

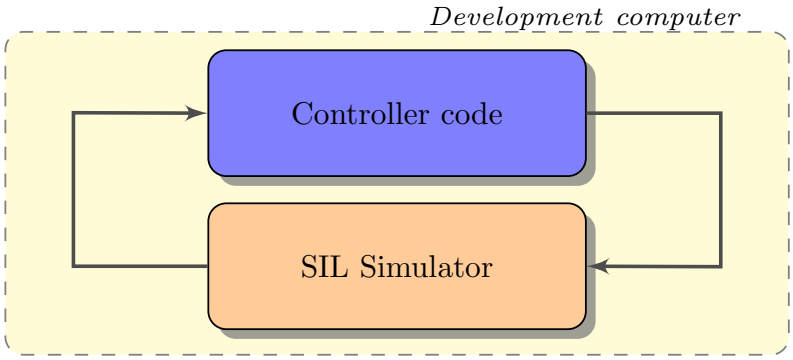

Figure 5.14: *Software-in-the-loop* schematic representation.

Using this method, we program the control algorithm on the development computer exactly as it will be programmed on the robot hardware. This is useful to emulate the working conditions of the robot in the computer, in order to prevent missing dependencies.

The *software-in-the-loop* method allows to test the controller code in order to find and repair errors in the algorithm or in the method itself. Additionally, it is possible to test the controller for critic scenarios.

5.4.4 Processor-in-the-loop

In the *processor-in-the-loop* method, the controller algorithm is compiled to be executed in the processor of the robot. The simulation, as in the previous method, still runs on the development computer. The fact that ROS can be installed in different platforms with virtually no operating differences allows to easily compile the SI-validated source code in the processor.

The control algorithm, using this method, can interact with the rest of the system using the input and outputs of the microprocessor. The rest of the system (as its low-level electronics), is simulated in the development computer. The inputs and outputs of the this computer are adapted in order to make the microprocessor act as if it was working with the real robot.

This method allows to detect errors related to the microprocessor operational conditions (compilation, optimization, available memory, calculation

power, frequency, latencies, etc).

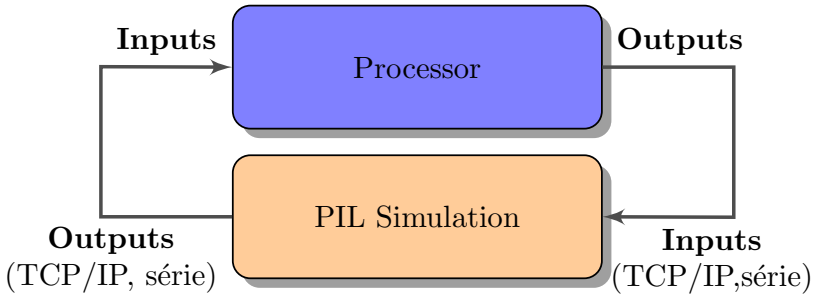

Figure 5.15: *Processor-in-the-loop* schematic representation.

Both SIL and PIL techniques have been used to validate the robot controller. In figure 5.16 we can see a simulation using UWSim [168] as a graphic output in which we test these two methods. For the SIL method the controller code and the simulator (both written in C++) run on the development computer. In the PIL method the simulation runs in the development computer while the controller runs in the RSM robot CPU board. Despite the differences in computing power and frequency (development computer far more powerful than the RSM robot CPU), both SIL and PIL simulations exhibit a similar behavior. This comparison between the two techniques validates the implantation of the code on the RSM robot embedded high level electronics.

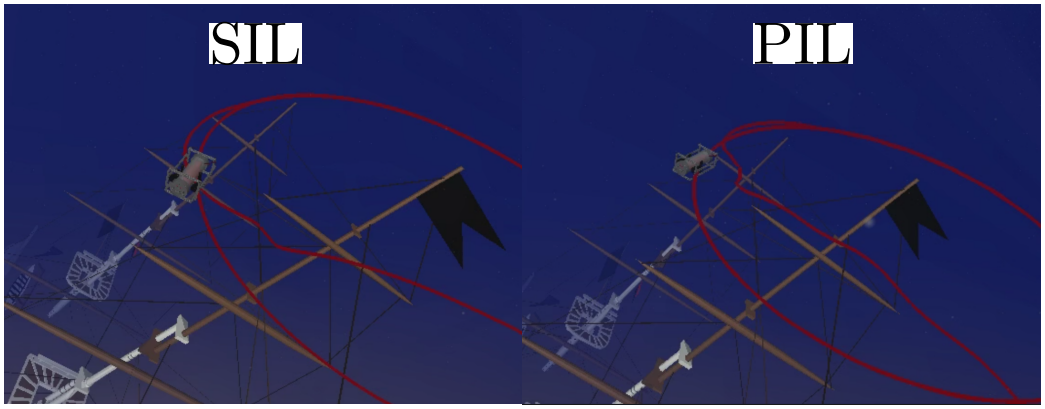

Figure 5.16: Simulations using SIL and PIL techniques.

An "*-in-the-loop*" method has not yet been applied, the *hardware-in-the-loop* or HIL (ongoing collaboration work with partners of UNICAMP, state of Sao Paulo, Brazil). This technique also uses the low level electronics of the

robot and allows to test entirely the acquisition and control lines. However, it needs a real time computer simulation in order to test the operations of the system. This type of simulation is done by special hardware with real-time capabilities (dSpace its used by our partners in Brazil).

5.5 Conclusion

In this chapter a nonlinear control method was introduced and a way of adapting it to different topologies was developed. Additionally, considerations about the applicability of this method (controllability over the trajectory) depending on the robot configuration were given.

A comparison between two different topologies using the control method was presented. In this comparison we observe how the same task can be performed with a different strategy depending on the propulsive topology. It also allows us to see how, depending on the chosen strategy, the performance in terms of convergence toward the desired trajectory and energy consumption can vary. This comparison confirms the importance of the propulsive topology for the performance of the robot. *How to choose the propulsive topology in order to maximize the performance?* is then the question raised by this work. The selected method is presented in the next chapter.

The Kalman filter estimation method and its implementation in the simulation of the RSM robot has been presented and discussed. Additionally, the controller has been programmed in C++ for its implantation on the real robot CPU. The target-compiled code has been validated using "*-in-the-loop*" techniques. While not yet integrated in the optimization of the AUV propulsion, these are important steps in the development of the RSM robot as an experimental platform.

Chapter 6

Genetic optimization

6.1 Introduction

The models, control technique and concepts presented in the previous chapters show the many parameters and sub-systems that a designer needs to address to successfully create an underwater robot propulsion system. Parameters and sub-systems as the gains of a controller and the propulsive topology of the robot are essential to create a functional robot. Moreover, even for a functional underwater robot, a slight change on these design elements can cause great variations of the robot overall performance as it can be seen even in simulation.

When designing a robot to perform precise missions, said sub-systems and parameters must be taken into account in order to create a robot adapted to the user needs. The designer, faced with a blank sheet of paper at the beginning of the project, needs to imagine the missions the robot will perform and come up with an appropriate system. This system will be a carefully chosen combination of the design elements previously discussed.

Given the large amount of design elements, and their heterogeneity, finding a suitable configuration is a challenging task. If additionally, we are looking for a highly performing configuration, the design process becomes extraordinarily complex and no known systematic method exists.

To overcome such a complexity, the designer can choose from the following strategies:

- Use a descriptive approach, which is the classical "human" approach. The designer uses previous information and novel ideas in order to solve the problem. The ideas are transformed iteratively in the mind of the designer until a satisfactory solution is created. Since this process occurs in a semi-empirical way, the final design is as good as the

experience and capacity of the designer.

- Use a digital approach, which consists on giving to a computer the tasks of creating a system or analyzing a previously designed one. This approach provides several advantages. It can be used to automatically find the design elements of a system that optimizes a certain characteristic. Computer aided design (CAD) techniques, take precise instructions about the features to optimize and, using an iterative process, find the most fitted solution(s).

Taking into account the number of parameters and sub-systems to determine before properly designing a robot propulsion, it is clear that such a difficult task would greatly benefit from a computer-based optimization design method. Namely, the design problem discussed in this research work, can be formulated as a global optimization problem.

6.1.1 Global optimization problem

The goal of this kind of problem is given in the following general definition:

Definition 1. *Global minimum*
Given a function $f : M \subseteq \mathbb{R}^n \to \mathbb{R}$, $M \neq \emptyset$, for $\vec{x}^* \in M$ the value $f^* := f(\vec{x}^*) > -\infty$ is called a global minimum, iff

$$\forall \vec{x} \in M : \quad f(\vec{x}^*) \leq f(\vec{x}) \tag{6.1}$$

Then, $\vec{x}^*$ is a global minimum point, f is called objective function, and the set M is called the feasible region. The problem of determining a global minimum point is called global optimization problem.

In the case of the design of a robot, the global minimum would correspond to the best set of parameters that minimizes (or maximizes) the objective function f. It would correspond to the best robot, based on these parameters.

Generally, the objective function has several *minima* of different values. The optimization can be attracted to follow one of these minima, called *local minima*, even if its value is mediocre compared to other *local minima* or to the *global minima*. We can define mathematically these *local minima* as:

Definition 2. *Local minimum*
For $\hat{\vec{x}} \in M$ te value $\hat{f} := f(\hat{\vec{x}})$ is called a local minima, iff

$$\exists \varepsilon \in \mathbb{R}^{+*} : \forall \vec{x} \in M : ||\vec{x} - \hat{\vec{x}}|| < \varepsilon \Longrightarrow \hat{f} \leq f(\vec{x}) \tag{6.2}$$

A local minima would correspond to a version of the robot which is locally better than its neighboring robots, but not the best of all search space M..

6.1.1.1 Different types of methods

There are several classifications of the different global optimization methods. Here, the classification given by Bäck [169] will be used. Indeed, we will separate these methods in two large groups:

- **Volume oriented**: These methods are based on an exhaustive search throughout the entire feasible configurations. This, of course, requires a finite search space. Examples of methods using this paradigm are Monte-Carlo strategies and cluster algorithms
- **Path oriented**: These methods follow a path in the feasible space, starting from a point that can be placed whether randomly or using previous knowledge. Path oriented methods can be divided into two sub-groups:
  - **Prediction methods**: these use an explicit model of the objective function to predict the best path to follow. Tunneling methods and gradient descent are examples of these type of approach.
  - **Exploration methods**: these try different paths and discard the ones that are not useful or successful. Examples of this approach are pattern search, rotating coordinates method and, to a great extent (they have traits corresponding to volume oriented and prediction methods), evolutionary algorithms.

For this work, the evolutionary approach will be used to solve the global optimization problem. This choice is based on the many successful applications of evolutionary algorithms to system design [170–173].

6.1.1.2 Evolutionary algorithms

Evolutionary algorithms are computational techniques based on the model of natural evolution formulated by Darwin [174]. They model the evolution process of a population of individuals using simplified processes of genetic mechanisms as described by Mendel.

Evolutionary algorithms manipulate the genotype of a population of individuals using genetic operators, this population undergoes a fitness-based selection process, in which the fitness of each individual depends on its adaptation to its "environment". With the proper definition of "genotype", "fitness" and "environment" we can apply these optimization algorithms to a myriad of problems, including robot design optimization [173].

From a mathematical point of view, an evolutionary algorithm can be defined as follows [169]:

Definition 3. *An Evolutionary algorithm (EA) is defined as an 8-tuple*

$$EA = (I, \Phi, \Omega, \Psi, s, \iota, \mu, \lambda) \tag{6.3}$$

where $I = A_x \times A_s$ is the space of individuals, and A_x, A_s denote arbitrary sets; $\Phi : I \to \mathbb{R}$ denotes a fitness function assigning real values to individuals.

$$\Omega = \{\omega_{\Theta_1}, ..., \omega_{\Theta_z} | \omega_{\Theta_i} : I^\lambda \to I^\lambda\} \cup \{\omega_{\Theta_0} : I^\lambda\} \tag{6.4}$$

is a set of probabilistic genetic operators ω_{Θ_i}, each of which is controlled by specific parameters summarized in the sets $\Theta_i \subset \mathbb{R}$.

$$s_{\Theta_s} : (I^\lambda \cup I^{\mu+\lambda}) \to I^\mu \tag{6.5}$$

denotes the selection operator, which may change the number of individuals from λ or $\lambda + \mu$ to μ, where $\mu, \lambda \in \mathbb{N}$ and $\mu = \lambda$ is permitted. An additional set Θ_s of parameters may be used by the selection operator. μ is the number of parent individuals, while λ denotes the number of offspring individuals. Finally, $\iota : I^\mu \to \{true, false\}$ is a termination criterion for the EA, and the generation transition function $\Psi : I^\mu \to I^\mu$ describes te complete process of transforming a population P into a subsequent one by applying genetic operators and selection:

$$\Psi = s \circ \omega_{\Theta_{i_1}} \circ ... \circ \omega_{\Theta_{i_j}} \circ \omega_{\Theta_0} \tag{6.6}$$

$$\Psi(P) = s_{\Theta_s}(Q \cup \omega_{\Theta_{i_1}}(...(\omega_{\Theta_{i_j}}(\omega_{\Theta_0}(P)))...)) \tag{6.7}$$

Here $\{i_1, ..., i_j\} \subseteq \{1, ..., z\}$, and $Q \in \{\emptyset, P\}$

Several algorithms use the paradigms of evolutionary optimization, we name here a few:

- **Genetic algorithms:** The most used technique within the evolutionary algorithms. Typically, the individuals are coded using binary encoding and they use crossover and mutation as operators [175].

- **Evolution strategies:** It was born to solve hydrodynamic problems such as shape optimization and drag reduction [176, 177]. In this technique the individuals are coded as vectors of real numbers. It uses primarily mutation, which can be *self-adapting*, depending on the objective function. Genetic operators manipulate random variables and are based on probabilistic formalism.

- **Evolutionary programing:** This technique evolves a population of computer programs to solve an algorithmic problem. The solutions to the optimization problems given by this technique are, then, computer programs [178].

Taking into account their ease of use, the simplicity of their implementation and their popularity (which gives many examples of applicability), the genetic algorithm method will be used from now on to solve the optimization problems in this work. Indeed, task-based propulsive solutions will be created writing the design of an AUV as genetic algorithm.

6.2 Genetic algorithms

6.2.1 Principle

Being a type of evolutionary algorithm, genetic algorithms (GA) are also based on the evolution theory of Darwing. Genetic algorithms use individuals, genes, populations, a fitness function and operators such as selection, crossover and mutation [175].

On a typical genetic algorithm application, the method follows these steps:

1. A population of individuals is created. This population can be created randomly or using previous information.

2. The individuals of the population are evaluated and given a fitness according to their performance based on the objective function.

3. Using the fitness of each individual as information. Genetic operators are applied to create new individuals:

   (a) Selection: The best individuals (best fitnesses) are selected from the population. This selection can be deterministic or stochastic.

   (b) Crossover: Two individuals (parents) mix their genes in order to create two new individuals (children).

   (c) Mutation: A change of value of the genes (alleles) of the individuals is applied. The change (from 0 to 1 or conversely) is decided by stochastic methods (each gene has a tiny probability of being mutated)

   (d) Population replacement: The new generation created by the previous operators replaces completely the old generation.

4. Go to step 2 if solutions are not yet satisfactory. Otherwise, finish the algorithm

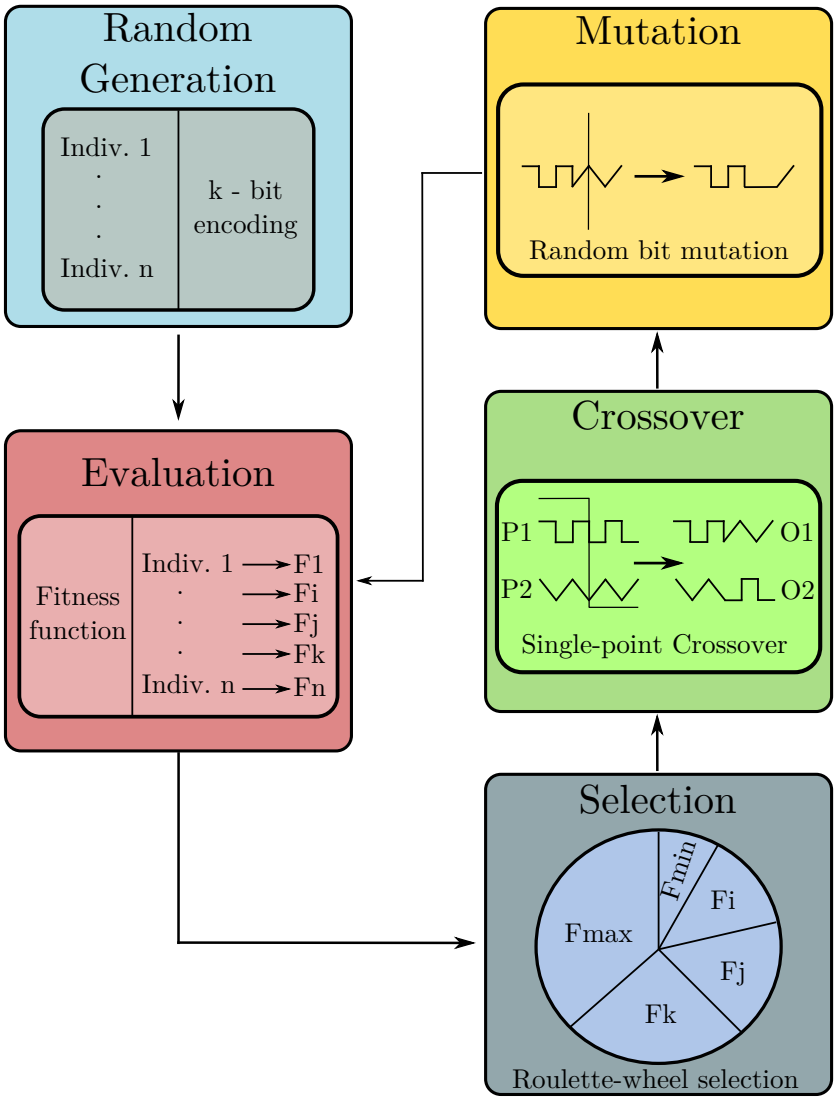

Figure 6.1: Genetic Algorithm.

6.2.2 Components

6.2.2.1 Population initialization

The first step using a genetic algorithms is the creation of an original population. This population will contain the first individuals, that later on, will

be evaluated and affected by genetic operators.

The μ individuals of a population are typically coded using k binary genes. The creation of these binary genes is made, most of the time, randomly. Indeed, a binary random variable is sampled $k \cdot \mu$ times in order to create every single gene of all the individuals of the population. The process is equivalent to a coin toss. The result is the start population $P(0) = \{\vec{a}_1(0), ..., \vec{a}_\mu(0)\}$.

Other techniques of population initialization exist. Some of them place individuals in zones of the population were the optimal solution is likely to be found. This type of strategy requires previous knowledge about the optimization problem and may lead to local optimization if mishandled.

6.2.2.2 Evaluation

Once created, the individuals of the population are evaluated. In order to do this, each individual must be decoded from binary to decimal base first.

The decoding is made taking the integer value from 0 to $2^k - 1$ of each individual and then mapping it to a real interval $[u_i, v_i]$. An example of this conversion is given here:

$$\Upsilon^i : I\!B^{k_x} \to [u_i, v_i]\Upsilon^i(a_{i1}, ..., a_{ik_x}) = u_i + \frac{v_i - u_i}{2^{k_x} - 1} \left(\sum_{j=0}^{k_x - 1} a_{1(k_x - j)} 2^j \right) \quad (6.8)$$

The decoded individuals are then, one by one, evaluated using an objective function $f(\Upsilon(\vec{x}))$ based on the characteristics to enhance/decrease. This objective function is scaled to the domain of the positive real numbers, since the selection operator needs positive fitness to work. This is done with a scaling function δ:

$$\delta : I\!R \times \Theta_\delta \to I\!R^+ \quad (6.9)$$

There are several scaling methods [175] (linear, logarithmic, exponential, etc). The choice of the right method depends on the application.

The scaling function gives to each individual in the population a *fitness* F, which describes their adaptation to the environment and it is useful to apply the operators effectively .

6.2.2.3 Selection

Selection is the genetic mechanism from which the best individuals of the population are saved from extinction.

The selection is based on the survival of the fittest rule of Darwin. In this type of selection, called *proportional selection*, each individual has a *selection probability* (also called *survival probability*) proportional to its relative fitness. It means that the better fitted an individuals is, with regards to the rest of the population, the bigger the chances of its survival are.

Mathematically we can determine this probability for the individual $\vec{a}_i(t) \in P(t) = \{\vec{a}_1(t), ..., \vec{a}_\mu(t)\}$ as:

$$p_s(\vec{a}_i(t)) = \frac{F(\vec{a}_i)}{\sum_{j=1}^{\mu} F(\vec{a}_j(t)} \tag{6.10}$$

Evidently this calculation only works if the values of the individual fitness are positive.

Using the above defined probability and the number μ of individuals in the population, we can calculate the expected number of occurrences of individuals $\vec{a}_i \in P(t)$ in the population after the selection:

$$\eta(\vec{a}_i(t) = \mu\, p_s(\vec{a}_i(t)) \tag{6.11}$$

The proportional selection mechanism is materialized by a method called the "roulette wheel". In this digital roulette we assign to each individual a slice of a size proportional to its relative fitness. Then a random number is generated and the individual whose segment spans the random number is selected (Fig. 6.1). This process is repeated for each individual of the population.

6.2.2.4 Crossover

The crossover, considered as the most important operator [169], is the operator by which new individuals on a population are created from parent individuals of the previous population. Indeed, new individuals can be yield from useful elements of different parents. These new individuals have the combination of the characteristics of their ancestors.

The crossover operator selects two parents from the population and recombines their genes (binary data) to produce two new individuals. A pair of individuals has a probability p_c of being crossed. The value of this probability depends on the application. In the literature we can find authors proposing p_c values ranging from 0.5 to 0.95.

Another topic of discussion is the crossover technique, that is, the way in which the elements from the parents are combined to form their children.

Several techniques exist [169, 179]. In figure 6.1 we can see the most basic (and popularly used) technique: the single point crossover.

The single point crossover method, proposed by [180], uses a crossover position $\chi \in \{1, ...k-1\}$ which is selected randomly. The two resulting children are then created exchanging the bits to the right of χ between the parents. From two individuals $\vec{s}$ and $\vec{v}$ the resulting children are:

$$\vec{s'} = (s_1, ..., s_{\chi-1}, s_\chi, v_{\chi+1}, ..., v_k) \tag{6.12}$$
$$\vec{v'} = (v_1, ..., v_{\chi-1}, v_\chi, s_{\chi+1}, ..., s_k) \tag{6.13}$$

6.2.2.5 Mutation

The mutation operator, just like in nature, changes the genes of the individuals. In genetic algorithms, this operator will invert the bits (genes) of the individual. The probability of a bit from an individual to be mutated is given by a probability p_m. Several p_m values are proposed depending of the author(e.g.$p_m = 0.001$ for [181], $p_m = 0.01$ for [182]). In this work we will use a rule in which we will calculate p_m in order to have 50% of mutants expected per generation.

Mathematically, the individual after mutation is calculated in the following way:

$$a'_i = \begin{cases} a_i & , \text{ if } \chi_i > p_m \\ 1 - a_i & , \text{ if } \chi_i \leq p_m \end{cases} \tag{6.14}$$

where $\chi \in [0, 1]$, as in the crossover operator, is a uniform random variable which value is recreated for each bit.

6.2.2.6 Replacement

After applying selection, crossover, and mutation to the individuals of a population P_0, we obtain a new population P_1. This new generation will replace completely the old generation. The new individuals will take the place of the old individuals, which will be extinct. New generations will be created using this operation iteratively until the termination criteria is reached.

The termination criteria depends on the application. It can be a simple maximum number of generations or it can be associated to a specific fitness value for the best individual or average fitness of the entire population.

6.2.3 Application example

6.2.3.1 Rastringin function

Before applying the genetic algorithm to our particular problem, we need to verify its efficacy. This verification is necessary from a theoretical and practical point of view. Theoretically, it will allow us to confirm the pertinence of these type of algorithm and will improve our grasp over the method. From a practical point of view, it will allow us to understand the logic behind our code.

These type of verification are typically done using a known multimodal multidimensional function, known as the test function. These test functions have the property of having a great number of *local minima* and a known *global maxima*, which makes them a challenging problem, perfect to test the genetic algorithm.

Among the several functions found in the literature [183], in this work we will use the Rastringin function. This function is multidimensional and highly multimodal, with a large amount of *local minima* surrounding the *global maxima*. Mathematically we can define this function as follows:

$$f(x) = 10\,n + \sum_{i=1}^{n} [x_i^2 - 10\cos(2\,\pi\,x_i)] \tag{6.15}$$

The limits of this function are restricted to the hypercube $-5.12 \geq x_i \leq 5.12\,, i = 1, ..., n$. The *global minima* of this function is $f(x) = 0$, which is the result of $x_i = 0\,,\ i = 1, ..., n$.

In Fig. 6.2 we can see a plot for the Rastringin function with $n = 2$.

For our test,we will use a Rastringin function with $n = 20$. The configuration of the genetic algorithm is given next.

6.2.3.2 Parameters

In order to use the genetic algorithm, first we have to configure it correctly. The configuration of the algorithm implies the definition of the following parameters:

- *pc*: crossover probability
- *pm*: mutation probability
- μ : number of individuals per generation
- g : number of maximum generations

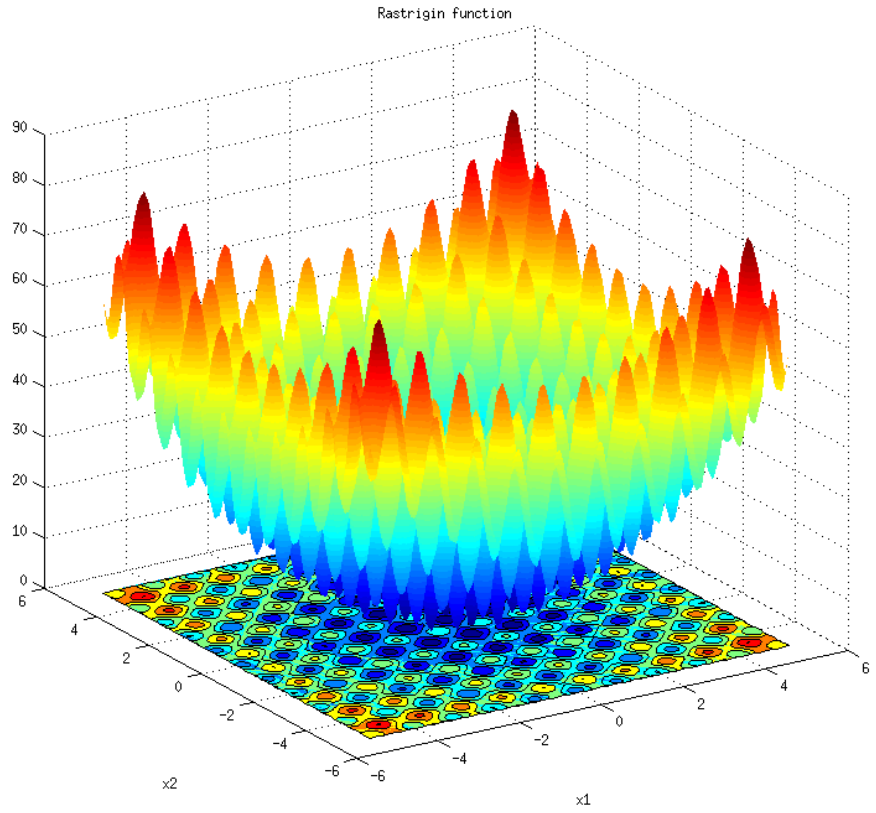

Figure 6.2: Rasringin function with $n = 2$ (2D).

The determination of these parameters depends on the application and a general method for their determination can not be stated.

For this particular problem, the chosen crossover probability will be the one proposed by [175], which is $pm = 50\%$.

The evolution is very sensible to the mutation effect, since a large mutation probability would prevent the algorithm from finding suitable solutions. For instance, a probability $pm = 50\%$ would be equivalent to a random walk.

In order to define the mutation probability, we will use the method used in [171]. Using these method, the mutation probability is inversely proportional to the number of bits of a chromosome (gene arrangement). If we want at least 50% of the total population in average to mutate, then the mutation probability is given by:

$$pm = \frac{1}{2\,n\,k} \tag{6.16}$$

with n the number of variables to optimize and k the number of bits coding these variables.

The number of individuals per generation (μ) in a population must be sufficiently large so the selection operator can pick from a great variety of individuals. Just like in nature, selection is more efficient if it has several type of individuals to chose from. This can lead to make the algorithm be able to "explore" more solutions and "discover" areas of the feasible landscape M that otherwise wold have been hidden.

A way of choosing the value μ can be based on the fact that it should allow the possibility of having the full range of values attainable by the fitness function F:

$$\Omega(F(M)) = k_{min} = 1 + \frac{1}{F_e} \tag{6.17}$$

with F_e the smallest possible variation on the fitness function, i.e. the variation on the fitness function given by a change of the last significant bit in a variable.

To determine the number of generations g we need to take into account that it has to be sufficiently large so the evolution has enough time to find suitable solutions and to improve the fitness of the population. We can chose the number of generations in order to give time to the mutation of modifying the complete chromosome of an individual. Given that the expected number of mutations per generation over an individual is 0.5, the minimum number of necessary generations is :

$$g_{min} = 2\,n\,k \tag{6.18}$$

Lastly, we can use the previous parameters to check the number of total evaluations on our genetic algorithm with regard to the cardinality of the optimization problem. Indeed, we can evaluate the result of the following equation:

$$\frac{\mu\,g}{\Omega(M)} \tag{6.19}$$

with M the set of solution candidates and $\Omega(M)$ the cardinal of the set M. The smaller the result of this equation, the bigger the difference between the evaluations made by the GA and the number of possible solutions of the optimization problem.

For our test, we will use $n = 20$ (number of variables) and $k = 14$ (bit for the encoding of each variable). This allows us to create the table 6.1.

Table 6.1: Evolutionary parameters for Rastringin optimization test.

Parameter	Symbol	Numerical value
Crossover probability	pc	0.5
Mutation probability	pm	1.786×10^{-3}
Population size	μ	1290
Number of generations	g	560
Number of parameters	n	20
Enconding bits per parameter	k	14

6.2.3.3 Results

Using the genetic algorithm with the given configuration we find the following results. In Fig. 6.3 we can see the evolution of both the best and average fitness for the population. It is clear how the average fitness curve increases along the best fitness curve. This shows how the genetic algorithm takes the information from the parents in order to create better and more fitted children as the generations go by. The gap between the two curves, created by the difference in grow rate, is an evidence of the variety of solutions the algorithm maintains during the evolution. Even if some of these solutions are not as good as the best one, they may contain useful information that, thanks to the crossover operator, can lead to improve the best fitness later on. This gap, mostly maintained by the mutation taking place on the individuals of the population, is very healthy for the evolution. Indeed, in its absence, the population converges to μ copies of a single individual. At this point, fitness improvement would become rather difficult and unlikely, since it is only driven by mutation.

Figure 6.4 shows the evolution of the parameters of the best individual. We can see how in the beginning the best solution has parameter values far from zero and how, as the evolution progresses, the values of the variables decrease as new areas of M are explored. The process continues, passing through parameters corresponding to better *local minima*, until the *global minima* configuration is found. At this point no further improvement is possible, but the genetic algorithm continues its research until the stop condition is attained.

We can also evaluate the efficiency of the GA by calculating :

$$\frac{\mu\, g}{\Omega(M)} = 3.7186 \times 10^{-19} \tag{6.20}$$

which means that the number of evaluations made by the algorithm was negligible when compared with the possibilities of the research space M.

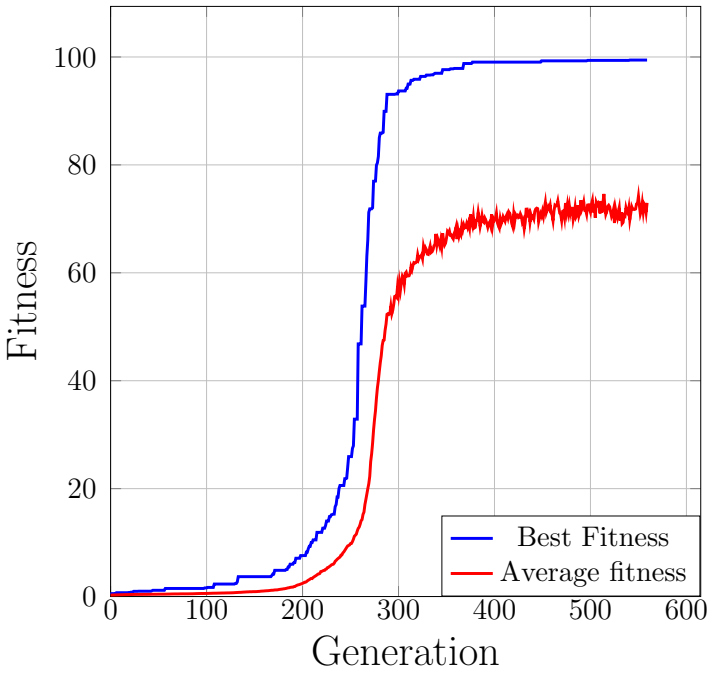

Figure 6.3: Best and average fitness evolution for Rastringin optimization problem.

6.3 Adaptation to AUV propulsion design

An underwater robot, regardless of its category, is a complex system affected by various and heterogeneous parameters. Indeed, a small change in these parameters can greatly influence the overall performance of the robot. They can even be the difference between a working and a defective robot. A non exhaustive list of some of these important design parameters was given in section 3.1

Considering this, the use of a genetic algorithms to find appropriate parameters to create a suitable robot for a given task seems reasonable.

In this section the adaptation of genetic algorithms to our specific underwater robot optimization problem will be discussed. Descriptions about the configuration of the evolution (probabilities, number of generations, number of individuals, etc) and of the design parameters (type, encoding, initialization, etc) will be given.

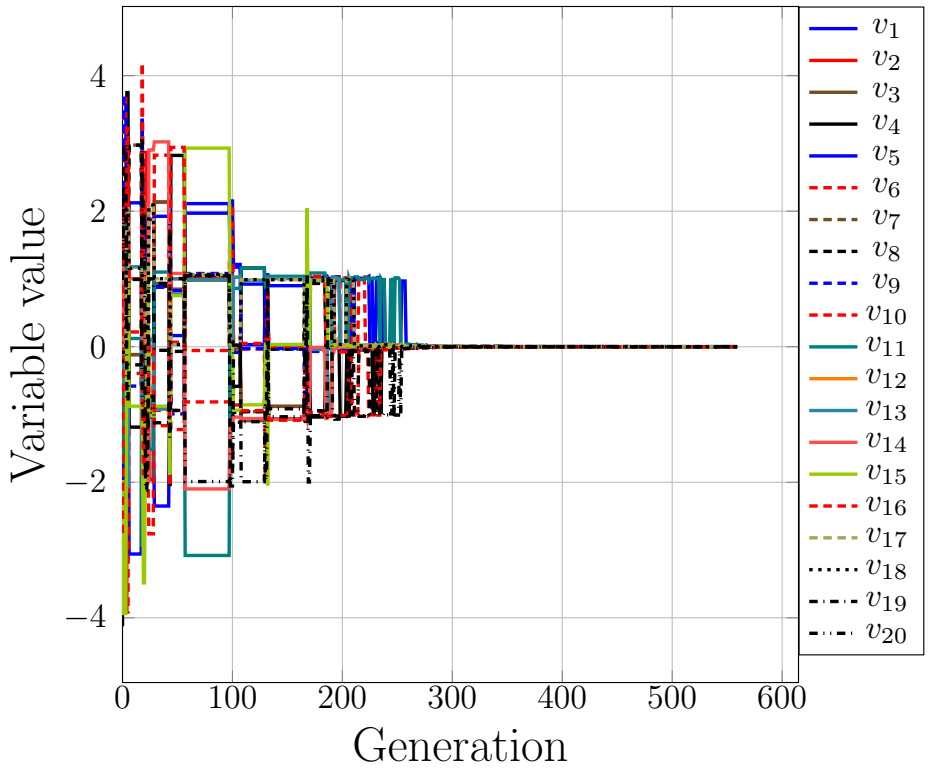

Figure 6.4: Evolution of the n variables of the Rastingin optimization problem.

6.3.1 Design parameters

Among the many design elements of an AUV, this work will focus on the ones regarding the propulsion and the control. Namely:

- Propulsion parameters
  - Number of thrusters: n
  - Thruster position: P_{ix}, P_{iy}, P_{iz}
  - Thruster Orientation: θ_i, ψ_i
- Controller FBLN parameters
  - Control gains: Λ, K_p
  - Tracking point: P_{ex}, P_{ey}, P_{ez}

Where i represents the i^{th} thruster in the topology.

6.3.1.1 Propulsion parameters

Propulsion parameters have a deep influence on the operation of underwater robots. The three propulsion parameters listed above will determine the controllability of the underwater vehicle. Indeed, as seen in section 5.2, each new propeller will contribute to the robot controllability with a propulsive wrench. The wrench contribution depends on the position and orientation of the thruster, since these parameters will dictate what DOF will be actuated by the thruster. For each additional thruster we use on the robot, we will either expand its propulsive capabilities (controllability) or simply add a redundancy on its DOF. For a robot with few thrusters, each new thruster can easily improve the robot controllability. However, for a robot with many thrusters each new thruster will have a decreasing contribution on the controllability.

To follow a given trajectory, the necessary DOF can be determined using kinematic considerations. In our genetic algorithm, this will be used to know the minimum propulsive capabilities needed by a robot. This pre-analysis provides an answer to *What* the robot needs in terms of DOF, while the genetic algorithm is in charge of finding *How* to provide them.

The propulsion parameters also have an influence over the energetic performance of the robot. The larger the number of thrusters, the heavier, cumbersome and costly the robot will be. In our genetic algorithm, each additional thruster penalizes the objective function, in order to account for this effect.

Likewise, if the thrusters of the robot are badly placed and oriented, the creation of the propulsive wrench will not be optimal, or worse inefficient, even with an appropriate number of thrusters. A defective propulsion topology needs to generate a greater propulsive wrench in order to follow a defined trajectory. Since our simulation saturates the force on the thrusters (by using the electrodynamical thruster model with bounded voltage), a poorly configured robot could not be as effective to follow said trajectory than a properly configured one. This will translate on bigger tracking errors (and energy consumption) throughout the evaluation.

6.3.1.2 Controller parameters

Controller parameters also play an important role on the performance of the robot. These parameters have to be precisely tuned in order to make the robot follow the desired trajectory efficiently.

The first parameter, the tracking point e, intervenes in the kinematic part of the controller. It determines the rank of the $\mathbf{T}$ matrix, which represents

the rigid body relation between the velocity of the tracking point $\boldsymbol{\nu}_e$ and the velocity on the mobile frame origin $\boldsymbol{\nu}$. The $\mathbf{T}$ matrix has to be inverted in order to calculate the necessary velocity of the robot in order to follow the desired trajectory. This means that the position of the tracking point has to guarantee the invertibility of the $\mathbf{T}$ matrix after the space reduction. Additionally, since ultimately it is the tracking point that follows the desired trajectory, its position will determine as well the convergence seed. Indeed, a tracking point far from the origin of the mobile frame will need less thrust power to correct its trajectory. This is due to the fact that small changes on the robot angular velocity result in big changes on the tracking point linear velocities. The position of this point relatively to the robot may be dictated as well by the task itself.

Lastly, the gain parameters of both the kinematic and dynamic control layers will also influence the robot ability to follow the desired trajectory. A fine tuning of these values is necessary to obtain a rapid and efficient convergence to the reference trajectory. On one hand, both gains need to be sufficiently large to guarantee a rapid convergence to the followed path. However, on the other hand, the kinematic gain also needs to be small enough to create a feasible reference velocity for the dynamic control layer (because of acceleration limitations). Moreover, the dynamic gain need to be small enough to avoid creating chattering on the propellers during the dynamic controller correction.

6.3.2 Representation of solutions

To properly implement the genetic algorithm, we need to precisely define the design parameters. These parameters are the building blocks used by the genetic algorithm to find adequate solutions to our problem, and they will become the solution at the end of the evolution process.

The parameters being of different nature (we have parameters of position, orientation and others representing controller gains), we need to determine each one on its own domain.

6.3.2.1 Number of thrusters

The topology of the underwater robot must be optimized, this means that the number of thrusters is variable in the search. To account for this characteristic we will use an activation bit. The activation bit will be in charge of activating or deactivating a thruster, i.e. including or excluding the thruster from the topology.

6.3.2.2 Thruster position

The thruster position parameter is in reality a position vector, composed of three coordinates expressed in the robot frame: P_{ix}, P_{iy}, P_{iz} (with i the number of the thruster).

Since the robot has a fixed shape and size, these three coordinates have different limits: P_{ix} can take values from $[-L,\, L]$ and P_{iy} and P_{iz} take values from $[-r_c,\, r_c]$. The volume described by these three coordinates is a cuboid, this corresponds with the real RSM Robot (Fig. 6.5) with its metallic frame. To avoid this heterogeneity we can make these parameters dimensionless:

$$\begin{aligned} \tilde{P}_i = (\tilde{x}_{pi}, \tilde{y}_{pi}, \tilde{z}_{pi}) \in I\!R^3_{[-1,1]} \\ \tilde{x}_{pi} = \frac{2\,P_{ix}}{L} \\ \tilde{y}_{pi} = \frac{2\,P_{iy}}{r_c} \\ \tilde{z}_{pi} = \frac{2\,P_{iz}}{r_c} \end{aligned} \tag{6.21}$$

with L and r_c the length and radius respectively.

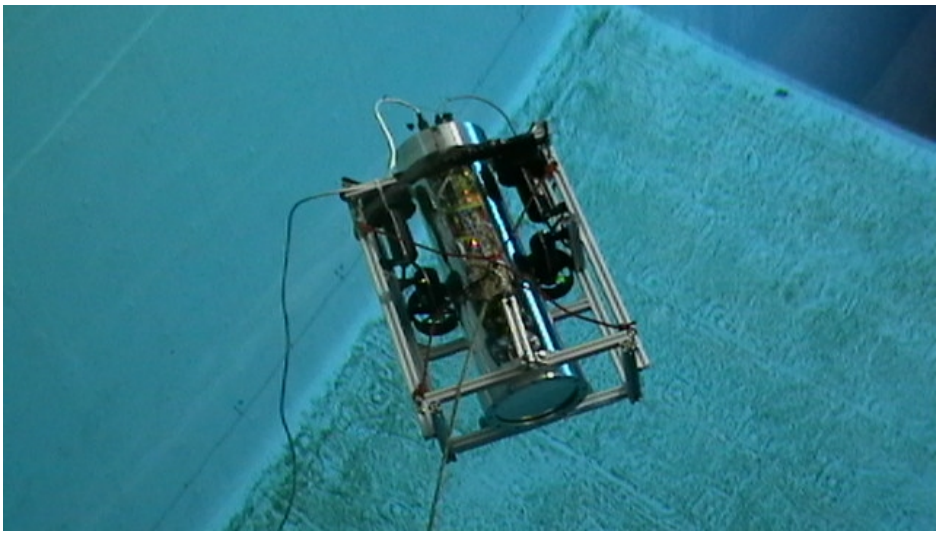

Figure 6.5: Structure of RSM robot (IRDL-ENIB, Brest Ifrement bassin).

Each one of these coordinates will be represented by a binary word of 5 bits. This representation will allow to identify 32 points on each axis of the robot, as shown on Fig. 6.6, which allows to have a resolution of 3.23% of the

total length. The choice of the discretization is a trade-off between accuracy and calculation time. Indeed, a small resolution gives a good precision but also increases the search space M, increasing the time necessary to solve the problem.

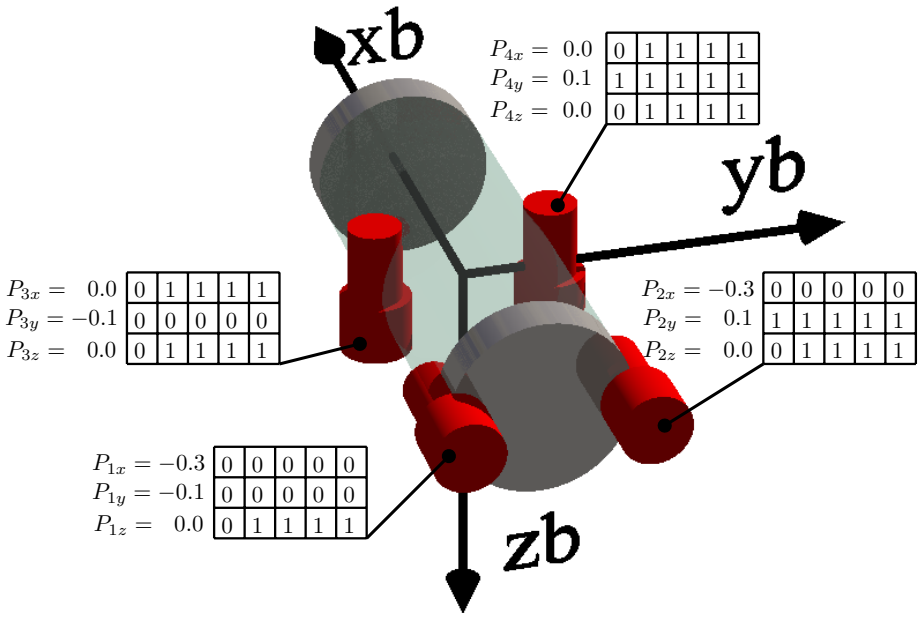

Figure 6.6: Thrusters position encoding (example of RSM robot).

6.3.2.3 Thruster orientation

Thruster orientation is a design parameter that contains two sub-parameters: *pitch* angle θ_i and *yaw* angle ψ_i which were defined in section 5.2.2.

These angles can vary within a $[-\pi/2, \pi/2]$ interval. The variation interval have been shortened since we consider the thruster capable of generating the same forces forward and backwards.

As we did with the position of the thruster, we can make these sub-parameters dimensionless:

$$\begin{aligned} \tilde{O}_i = (\tilde{\theta}_i, \tilde{\psi}_i) &\in I\!R^2_{[-1,1]} \\ \tilde{\theta}_i &= \frac{\theta_i}{\pi/2} \\ \tilde{\psi}_i &= \frac{\psi_i}{\pi/2} \end{aligned} \tag{6.22}$$

The encoding of these sub-parameters are made using 7 bits, which allows us to identify 128 possible orientations for each angle, as shown in Fig. 6.7. The encoding, with a resolution of 0.79% of the range, is also a trade-off between accuracy and time of calculation.

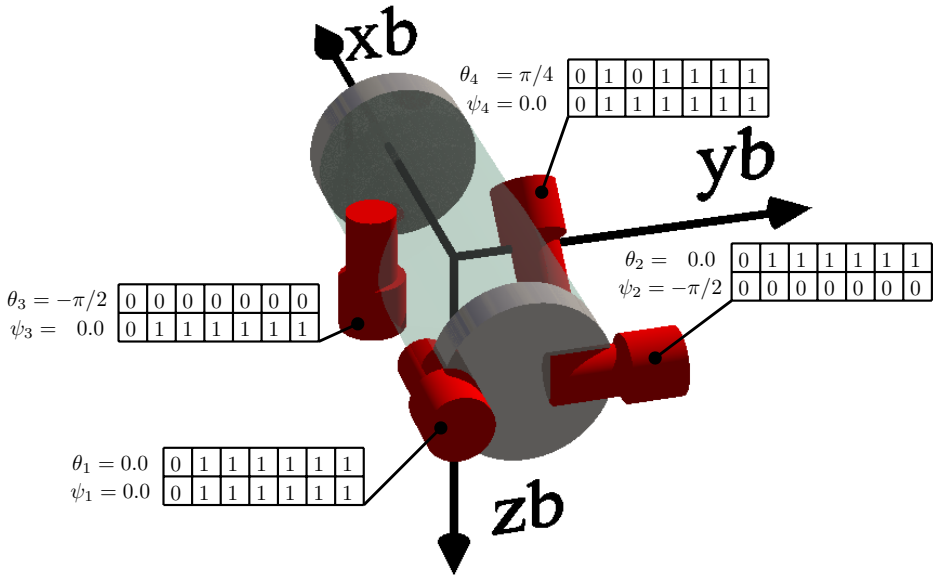

Figure 6.7: Thruster orientation encoding.

6.3.2.4 Control gains

The control method uses two gains, Λ for kinematic layer and K_p for the dynamic one. For the implementation of the GA we have decided to limit the values of these two sub-parameters to an interval $[0, 63.75]$. This choice was made based on previous knowledge about the values of λ and K_p, and because it is practical to encode each parameter with 8-bits, which allow us to discretize the variation range in segments of 0.25.

The definition of these two sub-parameters is the following:

$$\tilde{G} = (\Lambda, K_p) \in I\!R_{[0,63.75]} \tag{6.23}$$

6.3.2.5 Tracking point position

Just as the thruster position, this tracking point position parameter contains three coordinates expressed in the robot frame (with the same boundaries): P_{ex}, P_{ey} and P_{ez}.

The tracking point is the point that the controller will steer to follow the desired trajectory. Typically, it must be located within the boundaries of the robot (cylinder plus its metallic frame), because it should correspond to the position of one of the robot equipments (IMU, SONAR, sensors or camera). In our application, as we did for the thruster position parameters, we will make these sub-parameters dimensionless.

To encode these sub-parameters we will use 5-bits, which will allow us to obtain good precision (resolution of 32.3% on each axis) on the location of the tracking point.

Mathematically, this design parameters is defined as:

$$\begin{aligned} \tilde{P}_e = (\tilde{x}_e, \tilde{y}_e, \tilde{z}_e) &\in I\!R^3_{[-1,1]} \\ \tilde{x}_e &= \frac{2\,P_{ex}}{L} \\ \tilde{y}_e &= \frac{2\,P_{ey}}{r_c} \\ \tilde{z}_e &= \frac{2\,P_{ez}}{r_c} \end{aligned} \tag{6.24}$$

Table 6.2 sums-up the design parameters and their configuration.

6.3.3 Optimization problem

As it was introduced earlier, our optimization problem "create a suitable robot propulsion system capable of performing a given mission" is not very well defined. Evidently, a mathematical and concise definition of our problem will allow us to correctly design and implement the genetic algorithm.

The controller calculates the force to generate on the thrusters of a specific robot in order to make it follow a desired trajectory. We can also consider that a high level mission is a succession of properly tunned trajectories chained one after the other. Taking into account these two statements we

Table 6.2: Configuration of design parameters.

Name	Symbol	Enc. bits	Range	Resolution
Activation bit	a	1	[0, 1]	-
Thruster pos.(x)	P_{ix}	5	[-0.3 m,0.3 m]	$0.019\,m$
Thruster pos.(y)	P_{iy}	5	[-0.1 m,0.1 m]	$0.006\,m$
Thruster pos.(z)	P_{iz}	5	[-0.1 m,0.1 m]	$0.006\,m$
Thruster or. ($\vec{y}_b$)	θ_i	7	$[-\pi/2, \pi/2]$	0.025 rad
Thruster or.($\vec{z}_b$)	ψ_i	7	$[-\pi/2, \pi/2]$	0.025 rad
Kin. gain	Λ	8	[0, 63,75]	0.25
Dyn. gain	K_p	8	[0, 63,75]	0.25

can conclude that our controller, by making the robot follow the desired trajectory, also makes the robot comply with a given complex mission.

The optimal robot for the optimization problem must be one that follows precisely the desired trajectory given to the controller. This means that the distance from the tracking point to the desired trajectory must reach to zero and stay like that. We can rephrase the last statement saying that the integral of the trajectory error must be as close to zero as possible. Our optimization problem is then, the minimization of an objective function defined as the integral of the kinematic error ϵ_{kin} of the controller. This can be put mathematically as follows:

$$\text{Find } \mathbf{x}^\star \in I\!R|_{[-1,1]}^{5n+5} : (\forall\, \mathbf{x} \in I\!R|_{[-1,1]}^{5n+5},\ g(\mathbf{x}^\star) \leq g(\mathbf{x}) \tag{6.25}$$

with:

- $\mathbf{x}^\star$ The most suitable individual formed by $5n + 5$ parameters. Its is searched in a discrete subspace of $I\!R|_{[-1,1]}^{5n+5}$
- $g(x)$ The objective function to minimize, defined as the integral of the kinematic error of the controller over the time of the mission.

As seen in the previous equation, the number of parameters forming an individual can change. Each propeller has 5 parameters : 3 position coordinates(x_{pi}, y_{pi}, z_{pi}) and 2 orientations (θ_{pi}, ψ_{pi}), all of these defined with regard to the robot frame. Given that in the general optimization problem we want to find not only the best position and orientation for the thrusters but also an adequate propulsive topology, the number of thrusters, n, is a variable.

Since the robot has only one main controller, we have 5 parameters to optimize. The parameters of the controller are: 2 gains (for the kinematic

and dynamic control layers) and 3 position coordinates (x_e, y_e, z_e), for the position of the tracking point e.

6.3.4 Evaluation

Once the optimization problem and the design parameters have been defined, we need to establish how the genetic algorithm will determine whether an individual is fitted or not. This is done through an evaluation phase, in which the performance of an individual is determined by an objective function. The result of the objective function will be then scaled to obtain a fitness function, that will be used by the selection operator of the genetic algorithm.

The evaluation phase is done typically using a dynamic simulation of the real system using a numerical model. This cost effective method allows a great quantity of evaluations in a relatively short period of time. Additionally, it allows to evaluate solutions that might be harmful for the robot in the real world. This type of evaluation method has, however, a very important draw back: the *reality gap*, an issue by which the solutions found using the GA are not efficiently transfered to the real system [184]. This *reality gap* is often created by inaccuracies in the model, which the GA can use to obtain cheated higher fitness.

In the literature we find several attempts to cross the *reality gap*, here we name a few:

6.3.4.1 Simulation based

Since one of the main reasons of the existence of the *reality gap* are the inaccuracies of the simulations, a straightforward way of solving this problem would be creating more precise simulations. However, the more accurate the simulation is, the more computation-intensive it gets, which increases the time needed to find proper solutions. Additionally, model accuracy improvement has a limit, specially for systems involving fluid dynamics.

Other simulation-based methods propose to prioritize the evolution of solutions presenting a robust behavior. This can be done by means of a reduced simulation taking into account only the meaningful phenomena and dismissing the complex behavior of the real system [185]. As a consequence of this, the genetic algorithm is not able to exploit the inaccuracies of the system, since they are hidden. Robustness is also obtained by evaluating solutions on different simulation environments and initial conditions [186] or by using adaptive mechanisms, such as neural network controllers [187].

However, despite the drawbacks linked to the inaccuracies, the used models generally describe the studied system in a proper way. Most models

contain both accurate and inaccurate parts, the task of the GA user is to interpret whether the outcome of these models is appropriate or not.

In our application of genetic algorithms, we will use simulation as a evaluation mean. Our simulator *EAUVIVE*, far from being perfect (lacking of fluid dynamics and experimental validation of added mass terms), offers an acceptable degree of accuracy considering our goal: finding clues that could aid to the design of underwater robots.

6.3.4.2 Reality based

Needless to say, the simplest way of avoiding the *reality gap* problem is not having it. The reality-based approaches are based on this premise, as they consist on using the real system to evaluate the solutions. Evidently, a direct application of this approach is very time consuming since each evaluation is done in real time on the robot [187–189].

Other methods based on this approach have been tested. Namely a two step method, in which simulation is used first to determine a set of well-fitted solutions and then the evolution is completed on the real system [190, 191].

Since these approaches are based on the use of simulation to narrow down the research, they are subject to the same considerations of the simulation-based methods.

6.3.4.3 Robot-in-the-loop based

This approach also uses a co-evolutionary process, in which the algorithm evolves both the simulator and the controllers. Using this method, observed real data has to be correctly described by the simulator, then the best controller used on these simulators is implanted on the real system. This generates a new set of observed real data, which is what the simulator will try to describe on the next iteration [192, 193]. Similar methods [194] based on co-evolution try to reduce the fitness variation between the simulation and the reality.

These methods are based under the assumption that a simulator can be sufficiently improved to allow a perfect transfer between simulation and reality.

Other methods based on the robot-in-the-loop approach use a transferability function [184]. This method, based on fixed non-evolving simulations, makes the evolution aware of the limits of the model. It uses transferability measures that compares the simulated behavior with the real one and calculate its closeness. The transferability measures, as they are not possible to calculate explicitly for the complete research space, is interpolated from a

few selected solutions. With this method the fitness and the transferability objective are optimized.

6.4 Global optimization of underwater robots

In this section, the results of the application of the genetic algorithm to the underwater robot optimization problem are given. The results are ordered increasing the complexity of the problem. This allows us to better understand the complexity of the algorithm itself and to learn from the intermediate solutions we find.

6.4.1 Separate optimization of sub-problems

Before applying the genetic algorithm to the general optimization problem (find the underwater robot parameters that minimize the controller error), we will use it to solve a series of sub-problems. This will teach us important lessons about the design parameters relative importance in the design of the robot and about the configuration of the genetic algorithm (number of individuals, generations, encoding, etc).

For these sub-problems, the RSM robot will follow a straightforward task: a circle. The circle (in meters) is defined as follows:

$$\begin{cases} x &= 15 + 2\,\cos(v_{circle} \cdot t) \\ y &= 15 + 2\,\cos(v_{circle} \cdot t) \\ z &= 3 \end{cases} \tag{6.26}$$

with $v_{circle} = 0.3\,m/s$ a constant speed used to calculate, using the time t, the turning ratio.

The controller of the robot will be asked to control the position in space of the tracking point (x_e, y_e, z_e) and to keep the roll angle $\phi = 0$.

The fitness calculation for the sub-problems studied here will be done using the following equation:

$$F(x) = \frac{100}{1 + g(x)} \tag{6.27}$$

Where $g(x)$ is the objective function calculated as the addition of the normalized integral of the kinematic error e_{kin} (position) and the normalized energy consumption :

$$g(x) = \frac{\int_0^t e_k\,dt}{N_{e\,kin}} + \frac{\int_0^t P\,dt}{N_{energ}} \tag{6.28}$$

with N_{ekin} and N_{energ} the normalizing factors corresponding to the kinematic error integral and the energy consumption, respectively. The first factor is calculated as the integral of the kinematic error for a static robot (not using at all its thrusters), its numerical value is $19\,m\,s$. The second factor is the energy consumed by the robot when all the four thrusters are saturated for the duration of the task, the numeric value is $4728\,J$.

For each subproblem, a predefined configuration of the RSM robot will be taken. The evolution will only act on one parameter concerned by the subproblem, leaving the rest with the *standard* configuration. Said *standard* configuration is shown in table 6.3 and it has a calculated fitness of 15.66.

Table 6.3: *Standard* configuration for the RSM robot.

Parameter	**Symbol**	**Numerical value**
Kinematic controller gain	Λ	1.0
Dynamic controller gain	K_p	1.0
Point e position	r_e	$[0.3, 0.0, 0.1]^T$
Number of thrusters	n	4
Thruster 1 position	P_1	$P_{1x} = -0.3, P_{1y} = -0.1, P_{1z} = 0.0$
Thruster 1 orientation	O_1	$\theta_1 = 0.0, \psi_1 = 0.0$
Thruster 2 position	P_2	$P_{2x} = -0.3, P_{2y} = 0.1, P_{2z} = 0.0$
Thruster 2 orientation	O_2	$\theta_2 = 0.0, \psi_2 = 0.0$
Thruster 3 position	P_3	$P_{3x} = 0.0, P_{3y} = -0.1, P_{3z} = 0.0$
Thruster 3 orientation	O_3	$\theta_3 = -\pi/2, \psi_3 = 0.0$
Thruster 4 position	P_4	$P_{4x} = 0.0, P_{4y} = 0.1, P_{4z} = 0.0$
Thruster 4 orientation	O_4	$\theta_4 = -\pi/2, \psi_4 = 0.0$

6.4.1.1 Optimization of control gains

The first sub-problem to address concerns the definition of the most suited controller gains. In this case, $\mathbf{x}$ will contain only the elements of $\tilde{G} = (\Lambda, K_p) \in [0, 63.75]$.

Given the reduced nature of the problem, the GA optimization parameters need to be adapted. Taking into account the given procedure to select these parameters (Section 6.2.3), we obtain the values of Table 6.4

The population size parameter, as for the crossover and mutation operator, depends on the application. Unlike the Rastringin example, here F_e gives little information due to the fact that it corresponds to the numerical precision of the computer. In the literature we find several propositions for the population size, however a general rule for its determination is not avail-

Table 6.4: Evolutionary parameters for control only sub-problem.

Parameter	Symbol	Numerical value
Population size	μ	20
Number of generations	g	35
Number of parameters	n_p	2
Number of coding bits	k	8
Size of genotype	s_{gen}	$2 * 8 = 16$
Crossover probability	pc	0.5
Mutation probability	pm	31.25×10^{-3}

able. In our work, the determination of μ, the size of the population, is based on trial and error.

Results The evolution of the best and average fitness is given in Fig. 6.8, it shows the rapid increase of the maximal fitness from a fairly good value found in the first generation (lucky start). The size of the population was sufficiently large to provide a good dispersion of individuals over the researchable space E. We can see as well how in the end of the evolution the fitness reached 21.60, which is 2.5 times bigger than the fitness for the *standard* RSM Robot. This puts in evidence the ability of the genetic algorithm to find suitable solutions for underwater robots controller design by use of dynamic simulation.

The average fitness follows the maximum fitness curve. The average fitness presents some variations due to the mutation operator, which creates unpredictably individuals of assorted fitness levels (the search is still at work till the end).

Figure 6.9 shows the searched parameters of the best individual for each generation. In the first generations, the algorithm finds the fittest individuals when it increases the value of the kinematic gains up to 3.5. Afterwards, it finds even fitter individuals decreasing the value of Λ. Finally, it finds a set of values ($\Lambda = 1.5$ and $K_p = 0.25$) that generates an individual with a fitness unbeatable by the rest of the individuals of the evolution. For clarity, the best individual found by this optimization is given in Table 6.5.

Table 6.5: Controller configuration parameters for the evolution best individual

Parameter	Symbol	Numerical value
Kinematic gain	λ	1.506
Dynamic gain	K_p	0.25

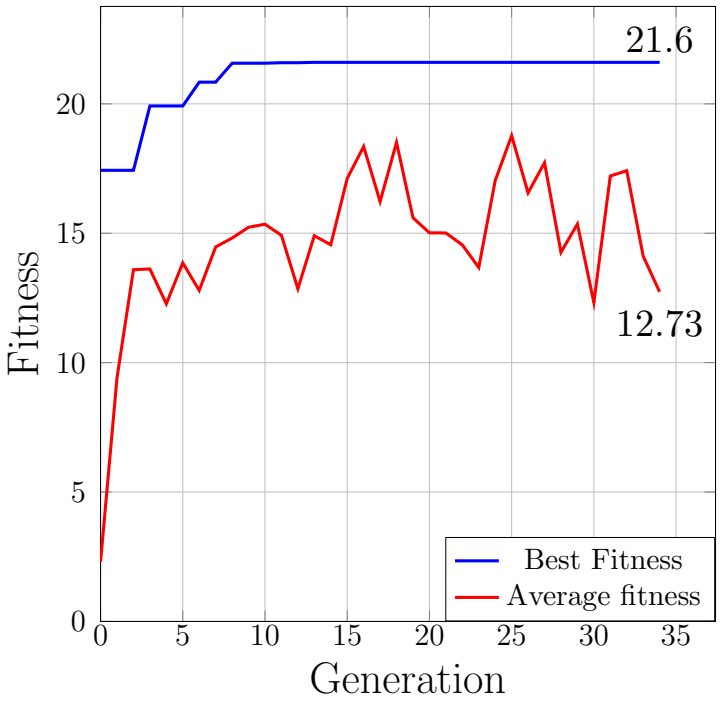

Figure 6.8: Maximum and average fitness of population for the optimized controller research.

We can observe how the value of the dynamic gain K_p for the best found individual is small, making the dynamic control loop less sensitive to the error in the desired robot velocity ν_{ad}. This can be explained by the fact that the dynamic controller ignores the dynamic of the thrusters. In consequence, it calculates a vector of forces $\vec{\mathbf{u}}_p$ that, later on, is not generated instantly by the thrusters. The delay in the establishment of the forces, introduced by the thrusters dynamics, leads to an *overshoot* of the dynamic controller.

Evidently, in order to fight this overshoot, the controller must create a control force $\vec{\mathbf{u}}_p$ of opposite direction, which in turn, *overshoots* again around the desired robot velocity. The solution found by the genetic algorithm is to give to the controller a less sensitive K_p, which masks the delay created by the thrusters dynamics and reduces the *overshoot*. A smaller overshoot reduces the error in position and reduces the energy consumption on the thrusters (less acceleration/deceleration cycles), which explains the good fitness of this individual.

In this sub-problem we see how the GA is capable of adapting to a defect in the controller model. The thrusters dynamic, not modeled in the controller, is seen as a perturbation by the GA. In order to reject this per-

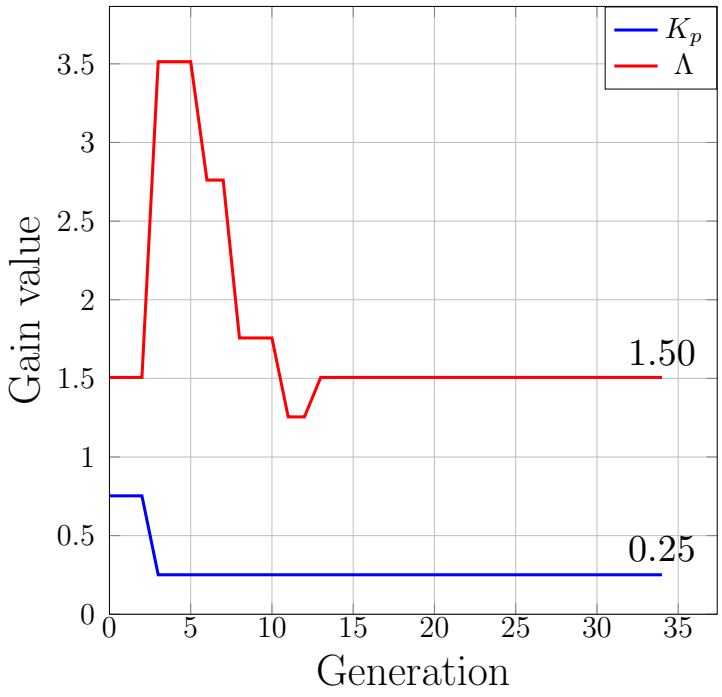

Figure 6.9: Evolution of parameters of the best individual of each generation.

turbation, the algorithm adapts the solution, making it less susceptible to this effect.

6.4.1.2 Optimization of point e position

In this separate subproblem optimization, the same circle task (Eq. 6.26) with the *standard* RSM robot (Table 6.3) will be used to optimize the position of the tracking point e.

Even if the position of the e point can have an impact in the calculation of the fitness function, its position is often fixed by the type of tasks we are trying to accomplish. For instance, in the case of a mission in which the robot must use a video camera to record, we would put the point e on the camera, in order to control and stabilize its position. However, as long as the point e is a virtual one, it could be used to improve the convergence of the robot to the desired task. In particular, it could be useful to help the robot to converge faster to a new trajectory after finishing a previous one. Indeed, changing the position of the point e, we can induce the robot to take different paths between trajectories. The dynamic positioning of the point e would be, in that case, a trajectory planning problem that is not addressed in this work.

Given that the point e can play different roles (it can be positioned arbitrarily for a specific task or used as a mean for the convergence). The goal of the evolution of this point can be defined differently depending on the case. Here, even if the position of the point e is evolved in order to minimize the kinematic error, the underlying goal of this optimization is to learn about our system. The configuration of this evolutions is given in Table 6.6.

Table 6.6: Evolutionary parameters for e position sub-problem.

Parameter	**Symbol**	**Numerical value**
Population size	μ	20
Number of generations	g	30
Number of parameters	n_p	3 (x_e, y_e, z_e)
Number of coding bits	k	5
Size of genotype	s_{gen}	$3 * 5 = 15$
Crossover probability	pc	0.5
Mutation probability	pm	32×10^{-3}

In figure 6.10 we can see the evolution of the maximum and average fitness curves. The final robot is about 1.25 times fitter than the *standard* version of the RSM Robot. This shows the effect of the positioning of this point in the performance of the robot. The increment of the performance thanks to the change in the point e position is, however, less important than the increase of performance thanks to the choice of the controller gains.

The evolution of the coordinates of e, x_e,y_e and z_e, are shown in figure 6.11. For the last generation, which contains the best configuration found throughout the evolution, the coordinates of e are the following:

$$r_e = [0.1548,\ 0.0194,\ 0.1]^T \tag{6.29}$$

The point e is placed halfway forward and slightly on the right (Fig. 6.12). The forwardly position can be explained in terms of stability. Indeed, a point placed further on the x_b axis would be affected by stronger lateral forces (along y_b axis) created by the horizontal thrusters action. The controller, not being aware of the thruster dynamics, must control the position of the point e correcting the overshoot created by the dynamic behavior of these actuators. The overshoot increases with the distance along x_b of the point r_e. Conversely, the closer the point e is to the origin of R_b, the larger is the thrust necessary to control it (shorter lever arm). The halfway forward positioning is then a trade-off between these two effects.

The small offset along y_b could be explained by the fact that the circle does not change of direction. Given that to follow the circle the robot must

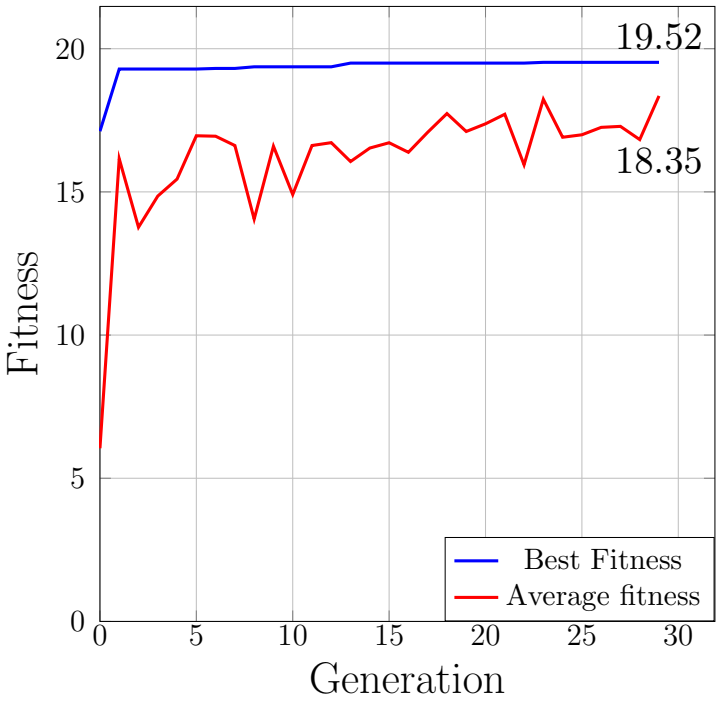

Figure 6.10: Maximum and average fitness of population for the optimized position of the point e.

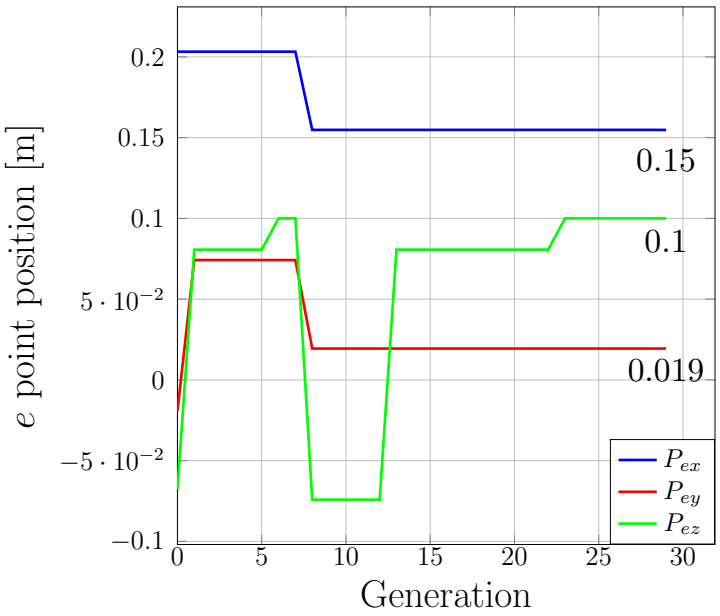

Figure 6.11: Evolution of the coordinates of e.

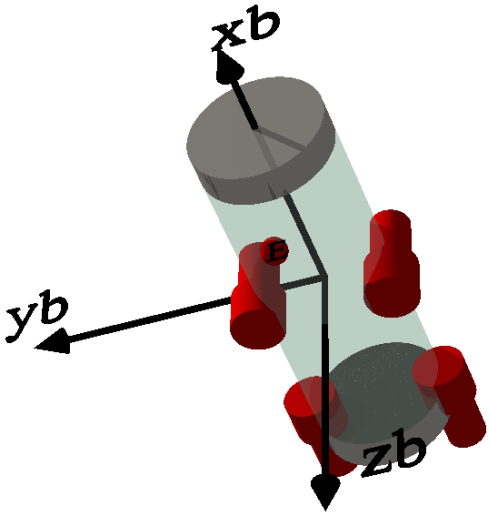

Figure 6.12: Position of e on the robot.

always turn to the same direction, the point e found by the GA leans closer to the circle. Evidently, for a task with a change on the direction of the circle, this bias would disappear.

The position of the point e along the z_b axis can be explained similarly to the position along the x_b axis: in order to control the point, the vertical thrusters benefit from a lever arm, allowing them to perform small corrections on v_e (as explained in section 5.2.2). It decreases the power consumption since less forces are necessary to create the movement. Evidently, as discussed before, a great lever arm can be counter-productive, but the small diameter of the robot (0.1 m) prevents this to happen.

6.4.1.3 Optimization of thrusters position

For this separate optimization, the parameters of the *standard* RSM robot will be taken and only the position of its four thrusters will be optimized. This means that $\mathbf{x}$ will be composed only of the elements of $\tilde{P} = (\tilde{x}_{pi}, \tilde{y}_{pi}, \tilde{z}_{pi}) \in \mathbb{R}^3_{[-1,1]}$.

The parameters of the evolution have been recalculated and are displayed on Table 6.7. Here, the population has been increased to account for the increase of the number of parameters to find (3 coordinate values per thruster). Again, this value has been selected based on trial and error.

Table 6.7: Evolutionary parameters for thruster position sub-problem.

Parameter	**Symbol**	**Numerical value**
Population size	μ	40
Number of generations	g	120
Number of parameters	n_p	$4 * 3 = 12$ (P_{ix}, P_{iy}, P_{iz})
Number of coding bits	k	5
Size of genotype	s_{gen}	$4 * 3 * 5 = 60$
Crossover probability	pc	0.5
Mutation probability	pm	8.35×10^{-3}

Results In figure 6.13 we can see how the sole optimization of the thrusters position increases the fitness of the robot. The final fitness is 1.27 times bigger than the fitness of the standard robot, however, best final fitness is not as high as the one found with the optimization of the controller gains. The figure also shows how, out of luck, the algorithm found an individual with a fitness of 11.5 within the first generation. We observe as well how the curves rapidly stabilize, which means that the optimization was very efficient and the number of generation too large. For its part, the gap between the best and average fitness curves shows that we have a good research in terms of exploration/utilization of the individuals information.

Figure 6.14 shows the position of the thruster for the best fitted individual found by the AG. Next to each truster we can see a curve showing the "way" followed by the thruster before finding its last position. As we can see, most of the thrusters had to endure several changes of position throughout the optimization, specially along the x_b axis.

Since the rest of the parameters are the same as the *standard* RSM robot, the AG had to find a suitable solution for the tracking point instability caused by the unoptimized parameters of the controller. The solution found by the algorithm was to place three thrusters next to the tracking point.

The forces generated by three thrusters placed in the front of the robot are more effective controlling the tracking point. In part, this is due to the fact that they are now closer to e, which reduces the effect of the *overshoot* created by the controller-ignored dynamics of the thrusters. Additionally, the position of the thrusters provide a lever arm from which the controller benefits, controlling e with reduced power consumption (larger forces). A last thruster is placed in the rear of the robot, it enhances the stability of the robot throughout the mission thanks to its *pitch* action.

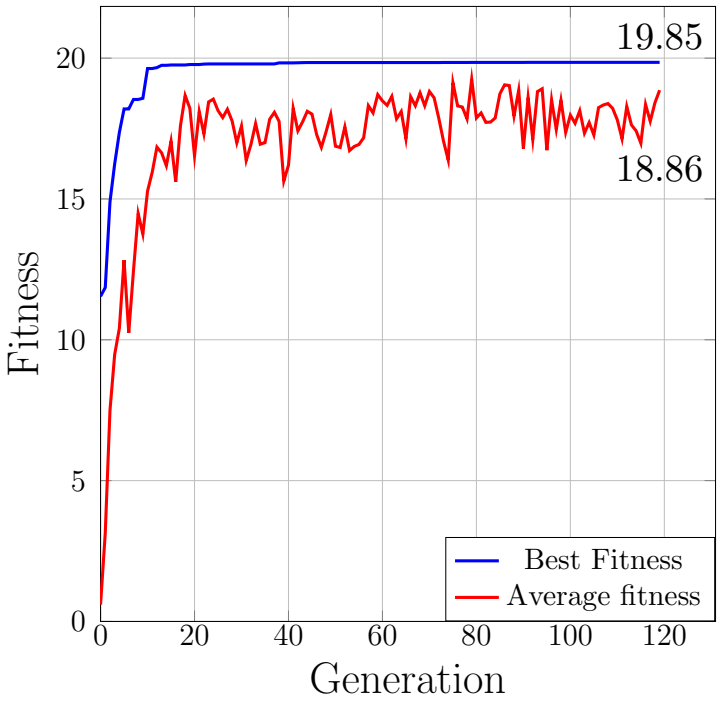

Figure 6.13: Evolution of the position of the thrusters of the RSM robot.

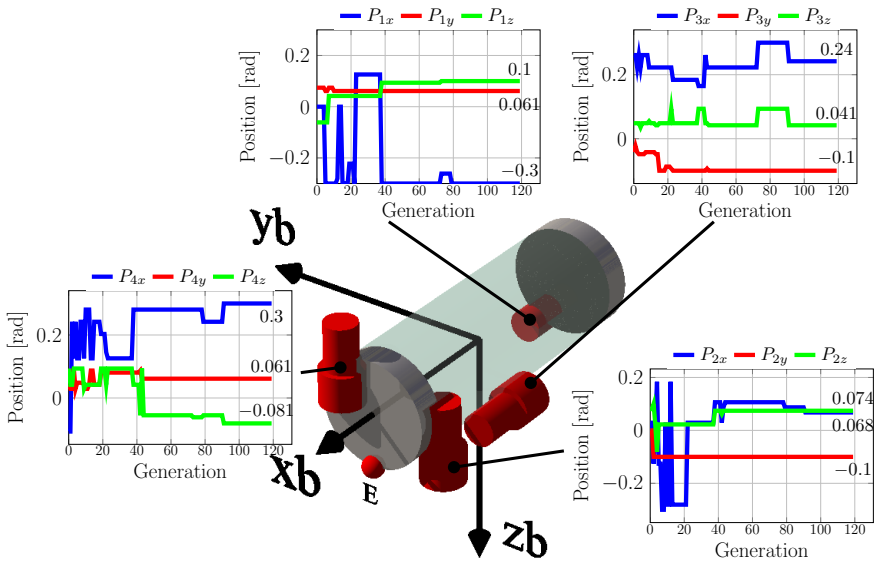

Figure 6.14: Position of thrusters on the robot.

6.4.1.4 Fixed thrusters orientation

In this separate optimization, the GA will only be able to modify the orientation (θ_i and ψ_i) of each thruster. It will keep the rest of the parameters of the *standard* RSM robot, in order to follow the circle trajectory. Mathematically, the genotype $\mathbf{x}$ will be composed only by elements of $\tilde{O}_i = (\tilde{\theta}_i, \tilde{\psi}_i) \in I\!R^2_{[-1,1]}$. The parameters of the evolution are given in Table 6.8. For this optimization, the size of the population is still 40 but the number of generations has been increased. The increase was motivated by improvements on the fitness value occurring even in the last generations of the evolution (on trial optimizations).

Table 6.8: Evolutionary parameters for thruster orientation sub-problem.

Parameter	**Symbol**	**Numerical value**
Population size	μ	40
Number of generations	g	160
Number of parameters	n_p	$4 * 2 = 8$ (θ_i, ψ_i)
Number of coding bits	k	7
Size of genotype	s_{gen}	$4 * 2 * 7 = 56$
Crossover probability	pc	0.5
Mutation probability	pm	8.93×10^{-3}

The average and best fitness for each generation are shown in Figure 6.15. As in the previous optimization, during the first generation the algorithm already found a good fitness (10). Then it rapidly improved the fitness of the best individual until it reaches 20, after that point the performance increase was slower. We can see how the end fitness is about 1.28 times bigger than the *standard* RSM robot. Considering that this fitness is achieved only by changing the orientation of the thrusters. We can conclude that, for a given task, using a canonical orientation of the thrusters (i.e. along the axis of the robot) can be detrimental for the performance of the underwater robot.

In figure 6.16 we can see the thrusters orientation for the best fitted individual found by the AG. Next to each thruster we can see the evolution of the orientations throughout the generations. In order to find the most suitable individual, the algorithm has spend the first 20 generations intensely changing the configuration of the best individual, increasing greatly the fitness in each iteration. After that point, the fitness improvements were due to small changes applied to the configuration found previously.

For this optimization, explaining the final individuals is a harder task. The thrusters have to improve the control over the point e and reduce the power consumption only by changing their orientation. We can see in figure

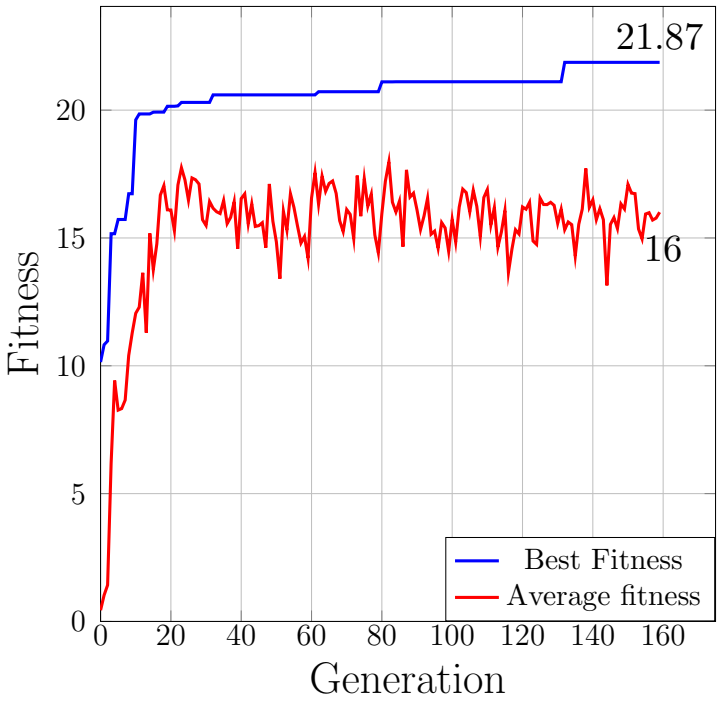

Figure 6.15: Best and average fitness for the optimization of RSM robot thrusters orientation.

6.16 that the rear thrusters have opposed orientations around the x_b axis. When the robot turns, this configuration will add up the components on y_b of these thrusters force (forces on opposite directions). The same components on y_b will be canceled out when the robot performs a surge movement. The end result is an additional component of force on the y_b axis during the mission. The orientation of the fourth thruster (almost perpendicular to the body) suggest that its main purpose is to create a force along the y_b axis. This thruster will compensate the forces on y_b created by the rear thrusters arrangement, improving the control of the point e over the y_b axis. Lastly, thruster number three will compensate the moment over the x_b axis introduced by the other thrusters and will generate the necessary forces to keep the robot at the desired depth.

6.4.2 Global vs. sequential optimization

Clearly, for a given mission, the optimization of the underwater robot should be applied to all of the design parameters and not individually. What needs to be determined is the way of doing it, given that we can identify two approaches. The first is to optimize groups of parameters sequentially, i.e.

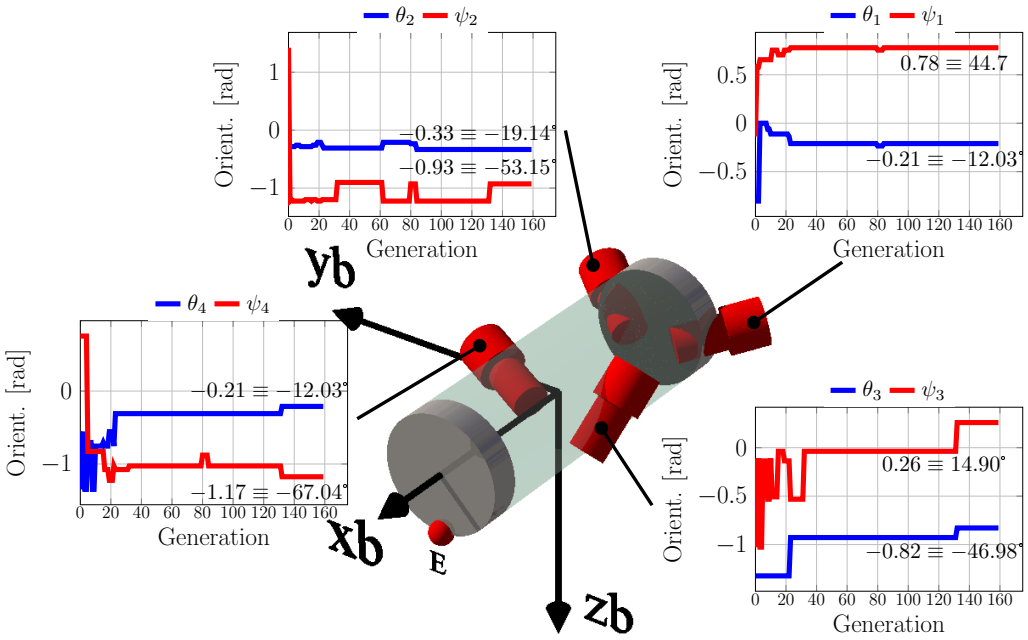

Figure 6.16: Orientation of thrusters on the RSM robot.

using the best parameters found in a previous optimization to configure the optimization of the next set of parameters. This allows us to incrementally generate better individuals at the end of each sequential evolution.

The second alternative is to perform the evolution globally (i.e. simultaneously). Using this method, all the parameters are evolved at the same time. Given that the research space is larger than in the sequential optimization, a greater number of individuals per generation must be used in order to obtain a good distribution over M. This will lead to a harder and longer optimization, but with the guarantee to have a genuinely global search (avoiding local minima)

6.4.2.1 Sequential optimization

The first tried approach will be the sequential optimization. For this method we will will rely on previous optimizations analyses. Consequently, we begin by the optimization of thrusters position. After that, with the found positions, we will optimize the orientation of the thrusters. Finally, for the given propulsion topology, we will optimize the control gains and tracking point position. The configuration parameters of the evolutions are given in Table 6.9.

Figure 6.17 shows the fitness evolution of the three sequential optimiza-

Table 6.9: Evolutionary parameters for sequential optimization.

	Parameter	**Symbol**	**Num. value**
Position	Population size	μ	40
	Number of generations	g	120
	Number of parameters	n_p	12
	Number of coding bits	k	5
	Size of genotype	s_{gen}	60
	Crossover probability	pc	0.5
	Mutation probability	pm	8.35×10^{-3}
Orientation	Crossover probability	pc	0.5
	Population size	μ	40
	Number of generations	g	160
	Number of parameters	n_p	8
	Number of coding bits	k	7
	Size of genotype	s_{gen}	56
	Crossover probability	pc	0.5
	Mutation probability	pm	8.93×10^{-3}
Controller	Crossover probability	pc	0.5
	Population size	μ	40
	Number of generations	g	65
	Number of parameters	n_p	5
	Number of coding bits (gains)	k_{gains}	8
	Number of coding bits (e)	k_{e}	5
	Size of genotype	s_{gen}	31
	Crossover probability	pc	0.5
	Mutation probability	pm	16.1×10^{-3}

tions. The first plot, the one of the thrusters position optimization, corresponds to the one already discussed in section 6.13. The central plot, shows the evolution of the orientation of the thrusters using the optimized positions found in the first evolution. The last plot shows the evolution of the control parameters (controller gains and tracking point position). This last evolution has been done using the found thrusters positions and orientation. Looking at these curves we can note how the *standard* configuration of the RSM robot, albeit effective, is not at all optimized for the task we are trying to perform. It is clear that by following canonical design procedures, depending on the task, we could be wasting most of the potential of the robot.

Figure 6.18 shows the simulation of the found propulsive solution while it performs the desired task. Additionally we can see the evolution of the kinematic error e_{kin}, which slowly converges towards the trajectory.

Figure 6.17: Optimization of the RSM robot parameters using sequential evolution.

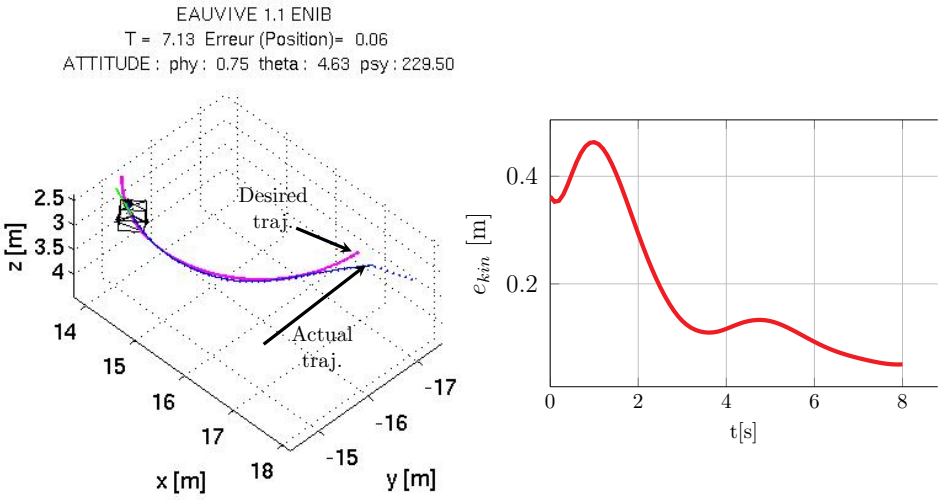

Figure 6.18: EAUVIVE simulation and error evolution for circle task (sequential optimization).

The resulting robot can be seen on figure 6.19. After accepting the curious arrangement of the solution found by the AG, we can see how the solution partly neglects the *surge movement*. Indeed, we see how only thruster one is in charge or driving the robot forward while thruster three is there only to provide *yaw*. Thruster two generates a component of force along x_b but its main action is to generate heave and roll motions along with thruster four.

The controller parameters have also changed, the gains of the controller are $\Lambda = 1.5$ and $K_p = 1.0$. The controller can be more demanding in its reactions, it can allow itself to react rapidly to errors. This is due to the

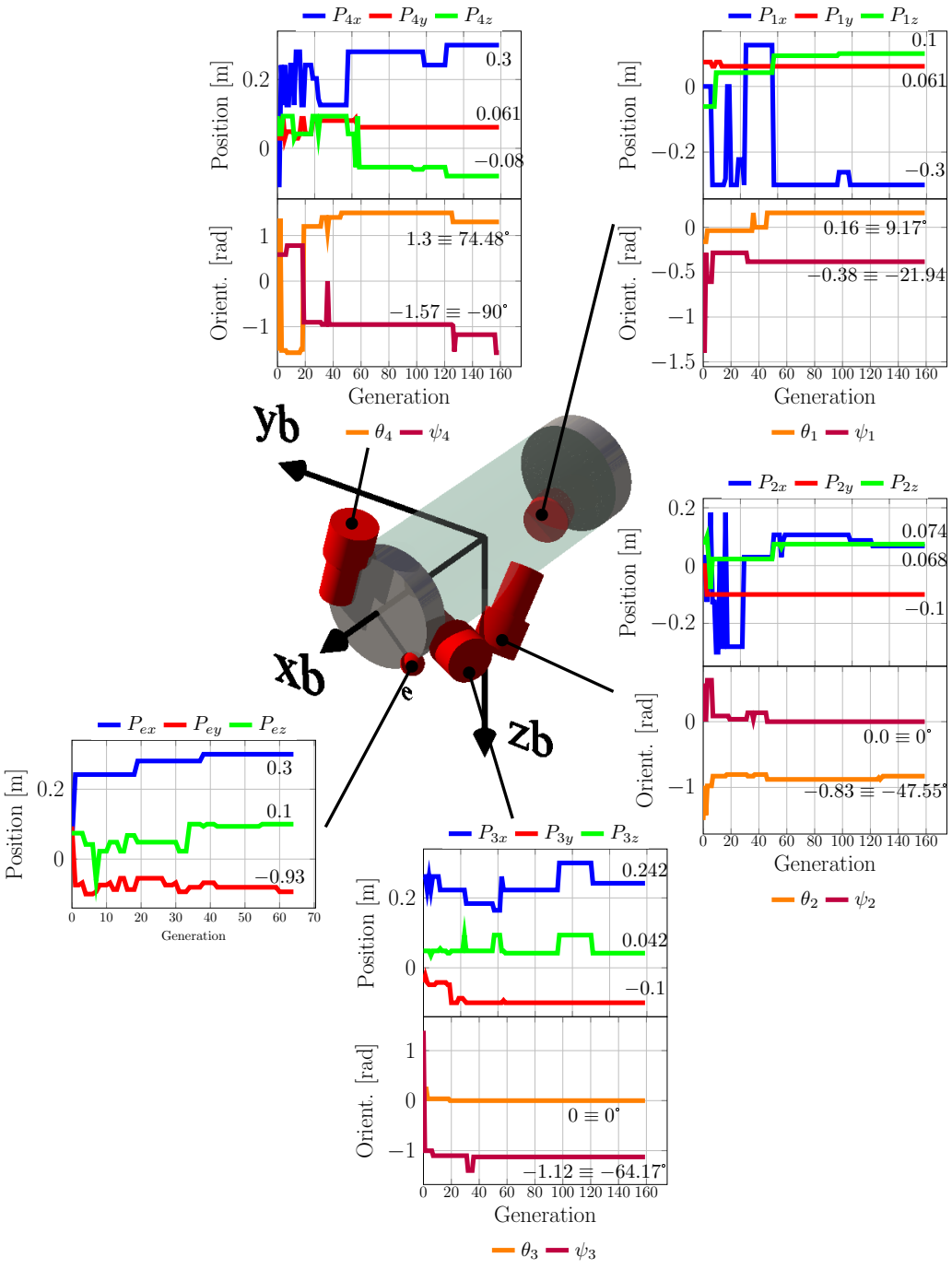

Figure 6.19: Resulting robot for the sequential optimization.

improved stability brought by the point e, since the position and orientation of the thrusters have been optimized to reduce the *overshoot* generated by thrusters dynamics.

Point e presents an interesting location as well, by positioning itself on the lower left edge of the robot it can benefit by the improved stability of that position (closer to the thrusters). Additionally, that point is somehow aligned with the direction of the force created by thruster one, which generates a surge motion on e.

Figure 6.20 shows the forces on the thruster and the total consumed propulsive power throughout the task. As we can see the thrusters one and two (horizontal) are the ones doing most of the work driving the robot forward, the other two thrusters are only in charge of stabilizing the robot. The consumed power shows a peak in the first instants of the simulation, which is caused by the controller suddenly asking the thrusters to give maximum force in order to accelerate toward the task. The high value of the peak is due to the integration step of the simulation, which is adapted to the dynamics of the AUV rather than the dynamics of the thrusters (faster). The high peak is present only in one step of the simulation (10ms) and does not affect significantly the integration of the consumed power (i.e. energy). After the peak, the power decreases rapidly until it stabilizes when the robot reaches the proximity of the desired trajectory.

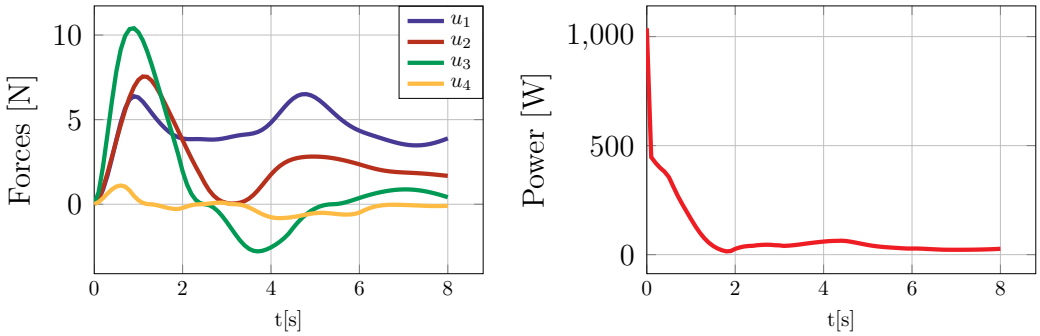

Figure 6.20: Forces and power consumption during the circle mission (sequential opt.).

6.4.2.2 Global optimization

The second approach to evolve the design parameters of the robots is the global one. The evolution will optimize all the parameters at the same time: for each one of the four thrusters it will search the optimal position and orientation and it will, at the same time, look for the best controller gains and

tracking point position. In order to keep the comparison between the two methods fair, we chose to give the same number of evaluations to both approaches (to assign the same CPU and time resources). Taking into account this, we have configured the evolution as shown in Table 6.10.

Table 6.10: Evolutionary parameters for global optimization.

Parameter	**Symbol**	**Numerical value**
Population size	μ	60
Number of generations	g	230
Number of parameters	n_p	25
Number of coding bits (thrust. or.)	k_{pos}	5
Number of coding bits (thrust. pos.)	k_{or}	7
Number of coding bits (gains)	k_{gains}	8
Number of coding bits (*e.* pos.)	k_{e}	5
Size of genotype	s_{gen}	147
Crossover probability	pc	0.5
Mutation probability	pm	3.4×10^{-3}

The fitness evolution can be seen in figure 6.21, from which we can observe two interesting characteristics. The first one concerns the final value of the maximum fitness, which is higher than with the sequential approach. The second is the speed of the fitness curve. Indeed, given that the algorithm is able to evolve all the parameters at the same time, it is capable of finding better individuals faster than the previous approach, making the research process more efficient. These two points make the global evolutionary research objectively better and preferable than the sequential evolution.

The simulation and kinematic error e_{king} are given in figure 6.22.

The end result of the evolution is shown in figure 6.23. The first that we notice when we see the robot is its resemblance with the resulting robot of the sequential evolution. We see how the thrusters have tendency to gather around the tracking point e. As discussed before, this masks the thrusters dynamic effects while providing enough lever arm to easily control the position of e.

If we analyze the positions and orientations of the four thrusters, we can see that their contributions to the robot wrench are strongly coupled. Thrusters one, two and three are mainly in charge of *yaw* and *surge* generation, they have also influence in the generation of *roll* due to their ψ angles (components of force along y_b). Thruster number one, being so close to the center of the robot, generates mostly roll and forces along y_b and x_b. Thruster four, for its part, is the only one with a strong vertical orientation,

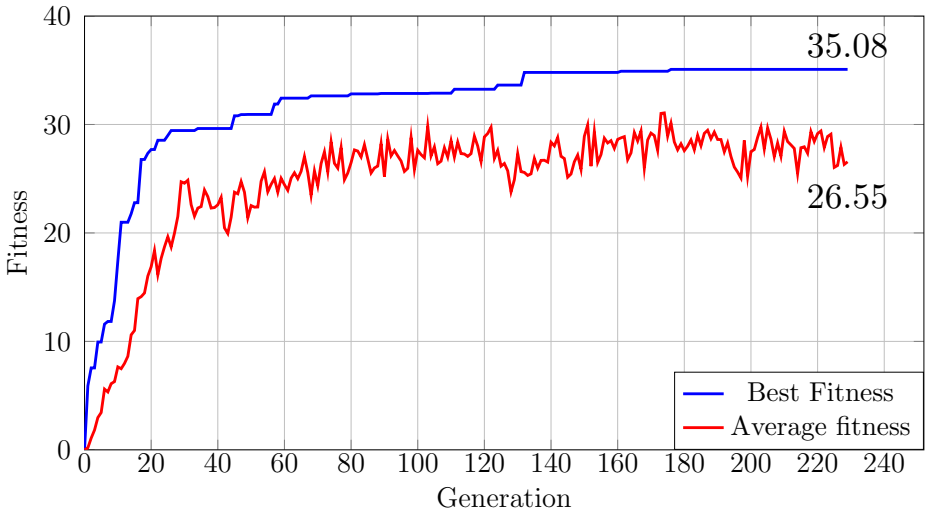

Figure 6.21: Fitness evolution for the global optimization of the RSM robot.

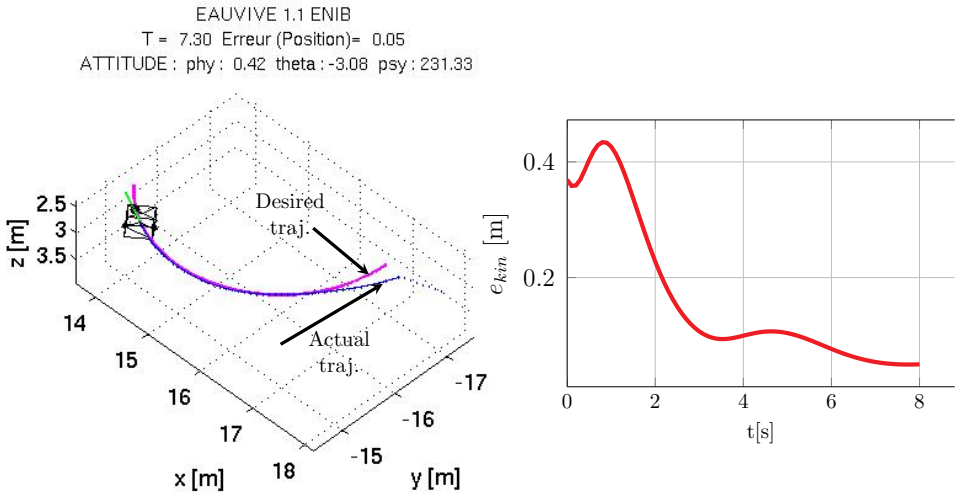

Figure 6.22: EAUVIVE simulation and kinematic error evolution for circle task (global opt.).

this is the thruster taking in charge the depth control of the vehicle. The point e has been placed in front, over the lower left corner of the vehicle, a similar position that the one found with the sequential method. Finally, the controller gains have been modified as well, now the kinematic gain is higher ($\Lambda = 2.76$), allowing the kinematic controller to be more demanding thanks to improved propulsion. However, the value of the dynamic gain remains rather low ($K_p = 0.75$), in order to mitigate the unknown (for the dynamic

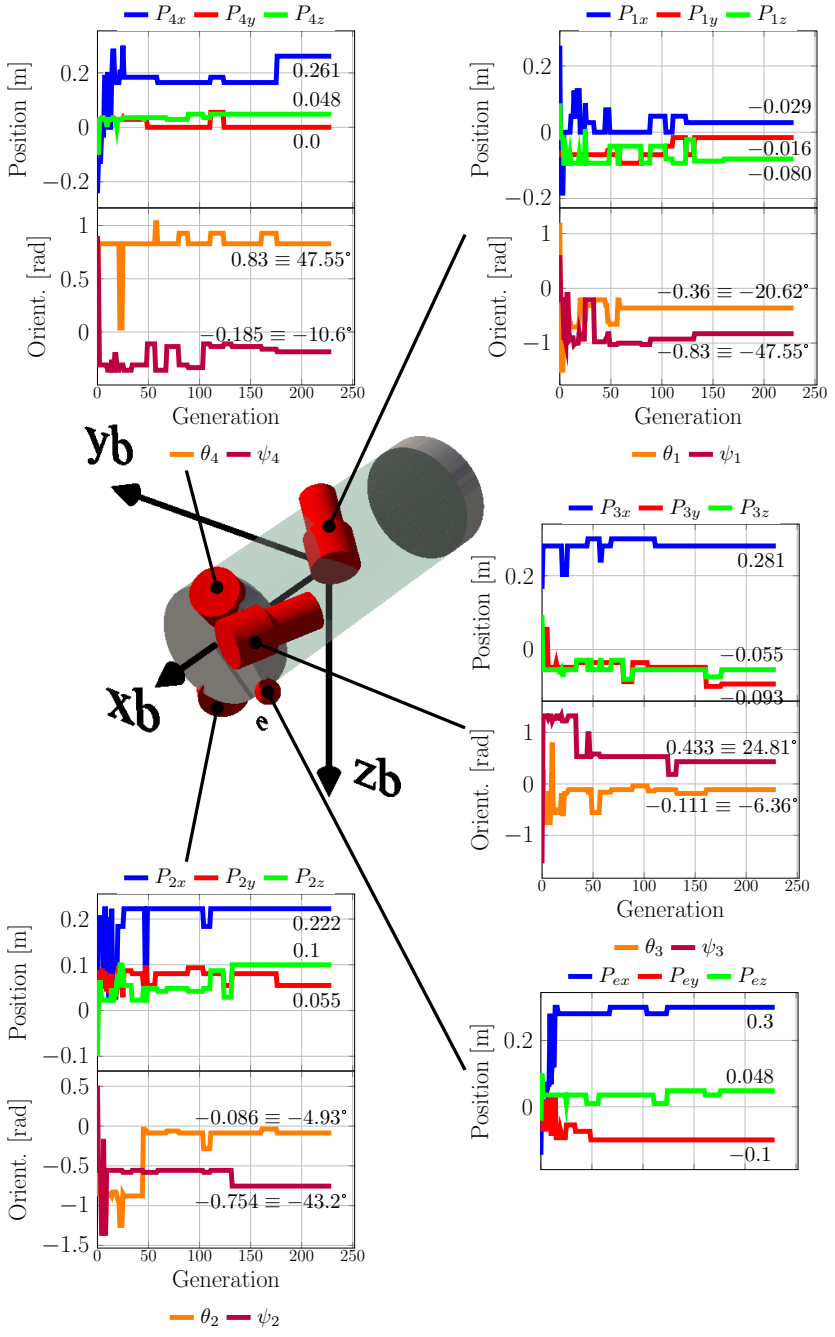

Figure 6.23: Final configuration of the RSM robot after global optimization.

controller) effects of the thruster dynamics.

Figure 6.24 shows thrusters force and the total propulsive power consumption of the robot. We can see how thrusters one and three are the most

used thrusters thanks to their ability to generate *surge* and *yaw*. The other two thrusters are in charge of stabilizing the robot and controlling its depth, judging by their produced force, these actions are not demanding. As for the sequential optimization, the total propulsive power peaks in the first step of simulation and then decreases until it stabilizes when the robot is in the vicinity of the desired trajectory.

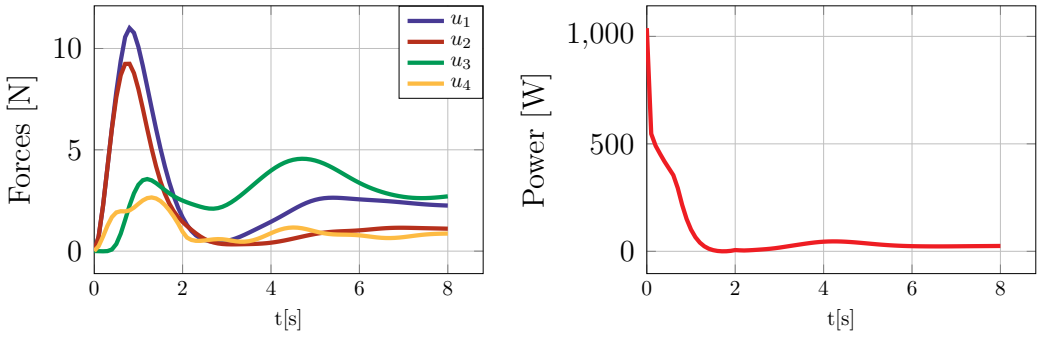

Figure 6.24: Forces and power consumption during the circle mission (global opt.).

6.4.3 Topology optimization

Comparing the results of the sequential and global optimization approaches we can see that the former is outperformed by the latter. Global optimization not only found a better solution (fitness 35 vs 32.5) but also matched the fitness of the sequential method in a third of the generations. Taking into account these two facts, it seems clear that the global approach should be used to carry on the next stage in our study.

In this stage of the optimization of underwater robots design, we will address the problem of determining the complete topology of the underwater vehicle in order to perform a given complete robotic mission. The optimization of the number of thrusters (up to twelve thrusters), position and orientation as well as the controller design parameters will be undertaken.

Unlike the previous mission, in which we asked the robot to follow a circle, we will now use a mission with a more realistic goal. The mission in question will be the inspection of a marine current turbine. The reason to chose such a mission is based on the fact that these devices, in addition to their installation costs, are very expensive in terms of maintenance. These turbines, as with any other structure installed in the sea, are exposed to aggressive degradation factors such as corrosion and biofouling (among others). In order to verify

the turbine condition, a team of divers check for signs of deterioration, which increases the effort and cost of maintenance. Based on the current trends for this type of operation, the use of underwater robots seems appropriate and inevitable.

Evidently, given the complexity of a mission of these characteristics, a good autonomy is needed. As discussed before, autonomy comes as a compromise between energy consumption and trajectory control.

The water turbine inspection mission (Fig. 6.25) can be divided in three phases:

- **Scanning of the seabed for the turbine:** After activating this phase, the robot follows a survey trajectory in order to find the position of the water turbine by scanning the seabed. The underwater vehicle can use multi-beam sonar information to scan the seabed, from near surface depth, until it detects the wanted target.
- **Diving toward the turbine:** Once the robot has found its target, it starts a diving phase. In this phase the robot will try to approach (but not too close) the turbine as fast as possible, trying to minimize the energy consumption at the same time.
- **Tomography (inspection) of the water turbine:** When the robot reaches its target, it starts the inspection of the energy generator. This phase is usually executed using a camera to record images of the turbine from all azimuths. In order to correctly capture images, the robot needs to encircle the device ensuring that it keep a constant distance from the target and its x_b axis always points the center of the circle.

We will apply our optimization algorithm to each one of these three phases in order to determine a suitable topology and controller.

For the following optimizations, we will calculate the fitness using a *topology factor*, K_t:

$$F(x) = \frac{100}{1 + g(x)/K_t} \tag{6.30}$$

This factor accounts for the number of thrusters in the topology, and can be calculated as follows:

$$\begin{cases} K_t &= 1 \text{ for } N_t \leq 3 \\ K_t &= 1.2252 - 0.0813\, N_t \text{ for } N_t > 3 \end{cases} \tag{6.31}$$

with N_t the number of thrusters in the topology. Assuming that in order to follow a trajectory in the space we need at least three actuators, we have decided to penalize the inclusion of thrusters only beyond the third one.

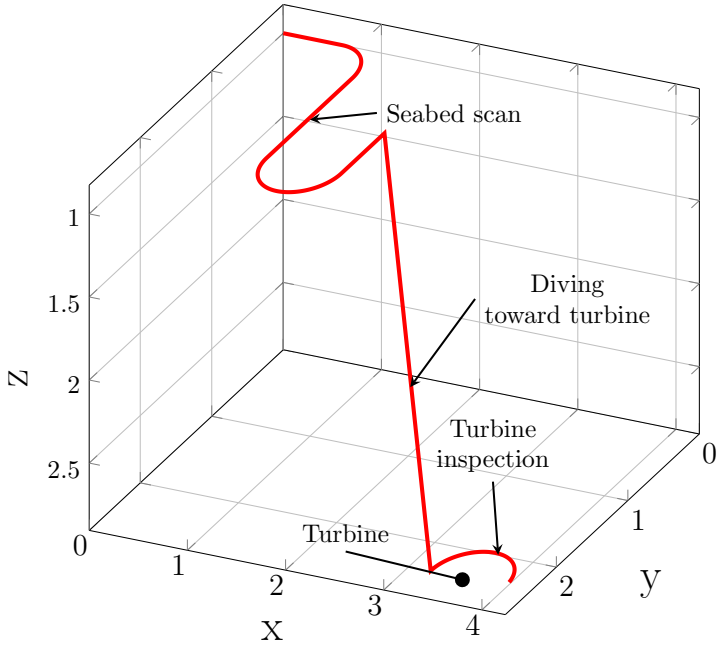

Figure 6.25: Trajectory for water turbine inspection.

The evolution parameters for the three phases described here are shown in Table 6.11

Table 6.11: Evolutionary parameters for topology research.

Parameter	**Symbol**	**Numerical value**
Population size	μ	50
Number of generations	g	max 600
Number of parameters	n_p	77
Number of coding bits (thrust. or.)	k_{pos}	5
Number of coding bits (thrust. pos.)	k_{or}	7
Number of coding bits (gains)	k_{gains}	8
Number of coding bits (e. pos.)	k_{e}	5
Size of genotype	s_{gen}	391
Crossover probability	pc	0.5
Mutation probability	pm	1.27×10^{-3}

Given that for these tasks the position of e is desirable to be in the front of the vehicle and along the x_b axis (it corresponds to the positioning of the eventual instruments and the advance direction), the position of this tracking point will be limited to the positive section of x_b.

6.4.3.1 Optimization for scanning of the seabed

In this part of the inspection mission, the robot goes from idle (or from a different mission) to scanning the seabed in order to find the water turbine.

The scan phase, also called survey mission, consists on exhaustively sweeping an area until the robot is capable of determining were the target is located. To perform such a task, the robot needs to control its position in space (*surge*, *sway* and *heave*). In addition, the detection process could benefit from a stabilized *roll* motion, so this degree of freedom will be also required (although not included in the fitness calculation). Finally, since this task usually involves going across great distances, the robot must consume the least amount of energy possible.

Figure 6.26 shows the fitness evolution. We can see how, unlike the previous optimizations, the algorithm takes longer to converge. The ascension phase of the curve is fast in the beginning but it decelerates rapidly as well. After one hundred generations the genetic algorithm keeps looking for suitable solutions but with a small pay-off per generation, however, this small increments add up and represent around 10% of the total fitness. This behavior is expected since the genetic algorithm is trying to evolve far more complex individuals than before.

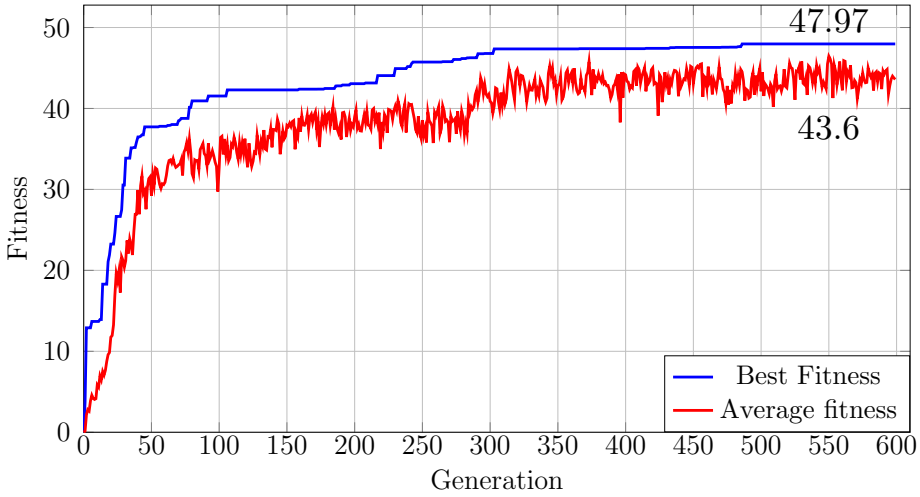

Figure 6.26: Global optimization fitness for seabed scanning phase.

A snapshot of the simulation and the curve describing the position error are shown in figure 6.27. We can observe how despite the difficulty of the task, the robot maintains an error inferior to 10cm throughout the task.

The solution to the optimization problem found by the algorithm is shown in Fig. 6.28. In order to perform this mission, the robot actuates four degrees

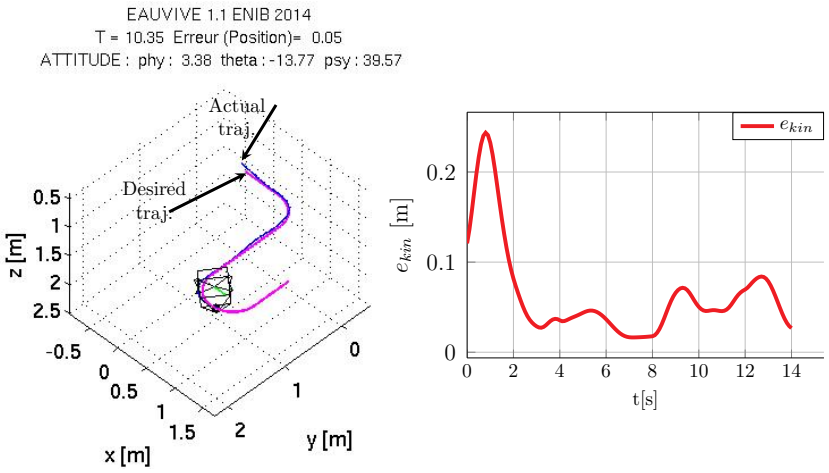

Figure 6.27: EAUVIVE simulation and kinematic error evolution for seabed scanning.

of freedom in R_b: *surge*, *heave*, *roll* and *yaw* using only 3 thrusters (actuating space is 3-DOF). From the figure, we can observe the contribution of the three thrusters to the actuated space.

In order to correctly perform this task, the robot must be agile in the $x - y$ plane. The topology achieves this using thrusters two and three to produce *surge* and *yaw*, with these two thrust axes it can advance and correct some light lateral deviations. Given that the main movement of the robot is defined as a forward motion (along x_b) with eventual turning motions, these two thrusters are enough to control the robot in x and y. Thruster one also contains a component of its force in the horizontal plane, which is used to increase the control on the lateral displacement.

The task requires as well a control over the depth of the vehicle. This is why all three thrusters have components over z_b. Given the small buoyancy of the vehicle (0.5% of the weight), a small control effort is needed to keep the robot at a certain depth. Said control effort is created by a linear combination of the vertical components of the thrusters. The vertical components of thrusters three and two are also used to correct the *roll* motion. This degree of freedom is mechanically stabilized thanks to the position of the center of gravity and buoyancy, reducing the need of a intensive control effort.

In what concerns the controller solution, the point e is located on the front of the robot, since advancing is the predominant motion of the mission. The distance to the thrusters gives a chance to control the tracking point thanks to the lever arm, which is long enough to favor control and to reduce the *overshoot*. The values of the gains are $\Lambda = 4.01$ and $K_p = 0.75$, the intricate

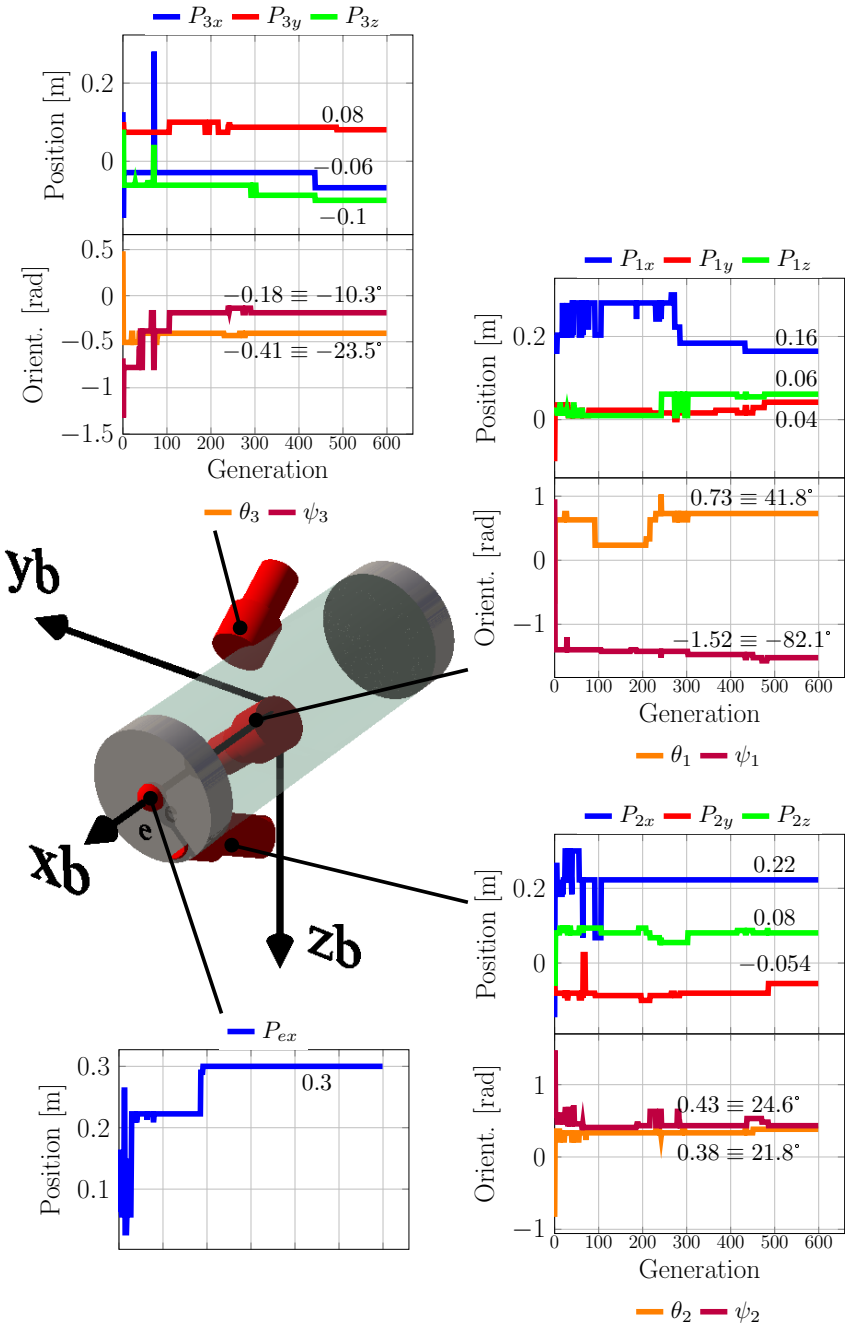

Figure 6.28: Final propulsion system for seabed scanning robot.

layout of the trajectory demands a tight kinematic control but, conversely, a slow cancellation of the dynamic effects (acceleration) caused by the change of direction.

The evolution of forces and propulsive power consumption are given in figure 6.29. Given that the robot has only three thrusters, all of them participate actively in the control of the vehicle. This is specially clear during the turning phases of the task ($t = 2$ and $t = 9$). Indeed, in order to comply with the change in the trajectory, the robot uses all its available means. Despite the active use of the thrusters we can see how the controller manages to follow the desired trajectory with small thrust forces.

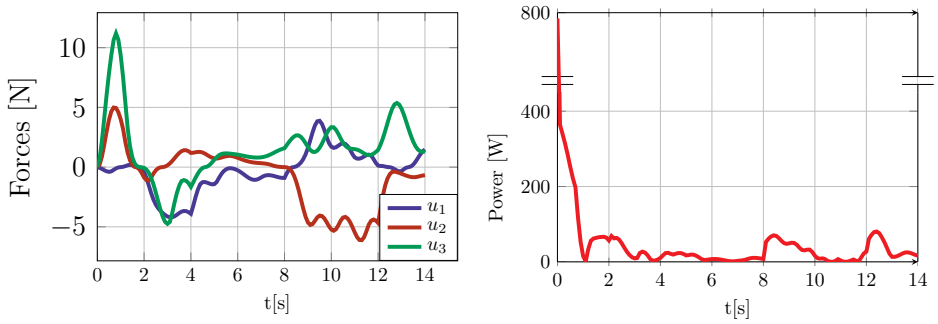

Figure 6.29: Forces and power consumption during the seabed scanning task.

6.4.3.2 Optimization for diving toward water turbine

After finding the location of the water turbine, the robot needs to reach it to start the inspection. To do so, it will start a diving phase, in which the main goal is to get closer to the target (at a given relative position, let say 1m before it).

The diving task consists simply on following a descending straight line. All the resources of the robot are devoted to correctly follow this trajectory. The task relies on the control of three degrees of freedom: *surge*, *sway* and *heave*. Knowing that the drag forces are greater while diving horizontally, we will also ask the controller to keep a *pitch* angle equal to the diving angle, since this could improve the overall performance of the robot. However, since we want to give the GA the liberty of choosing the diving method, this factor will be controlled but not included in the performance and fitness calculation. Lastly, the power consumption will also be included in the performance calculation.

In Fig. 6.30 we can see the evolution of fitness along the generations. As in the previous phase, the fitness convergence takes longer, around 100 generations to begin to stabilize. After that point, the increments are even smaller per generation, giving place to a long plateaus.

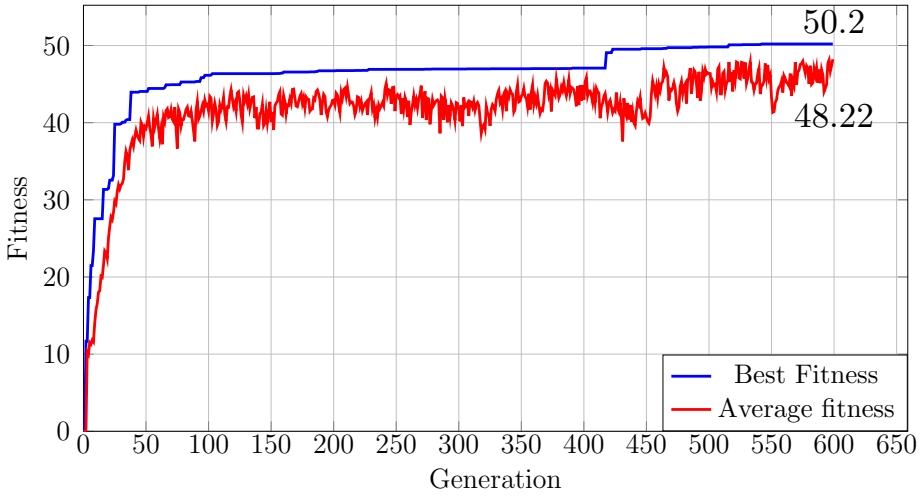

Figure 6.30: Diving phase global fitness evolution.

Figure 6.31 shows the simulation of the task and the evolution of the kinematic error. In this figure we see how the robot follows the trajectory diving horizontally (not using *pitch*). Indeed, the GA found that with this diving strategy it can maintain a kinematic error inferior to 10cm.

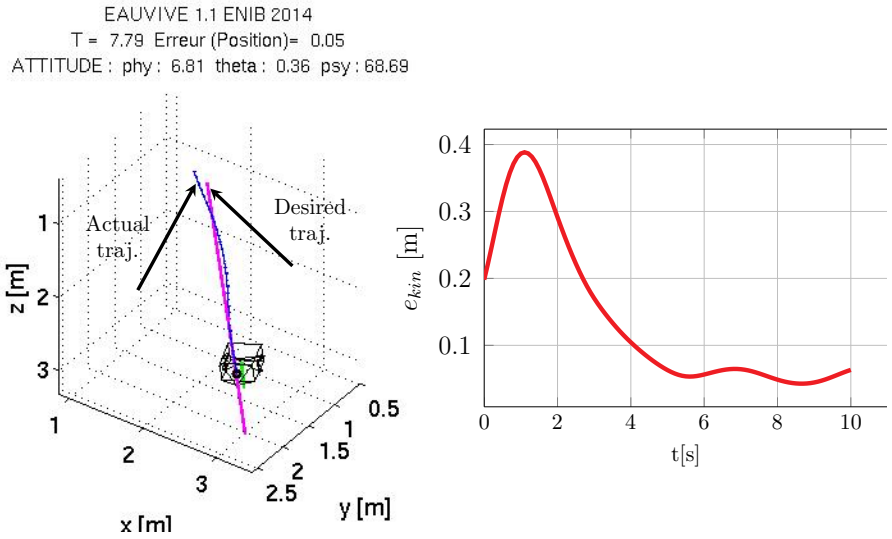

Figure 6.31: EAUVIVE simulation and kinematic error evolution for diving phase.

The found topology is shown in Fig. 6.32 where we can see how the thrusters are located in the front and center of the robot.

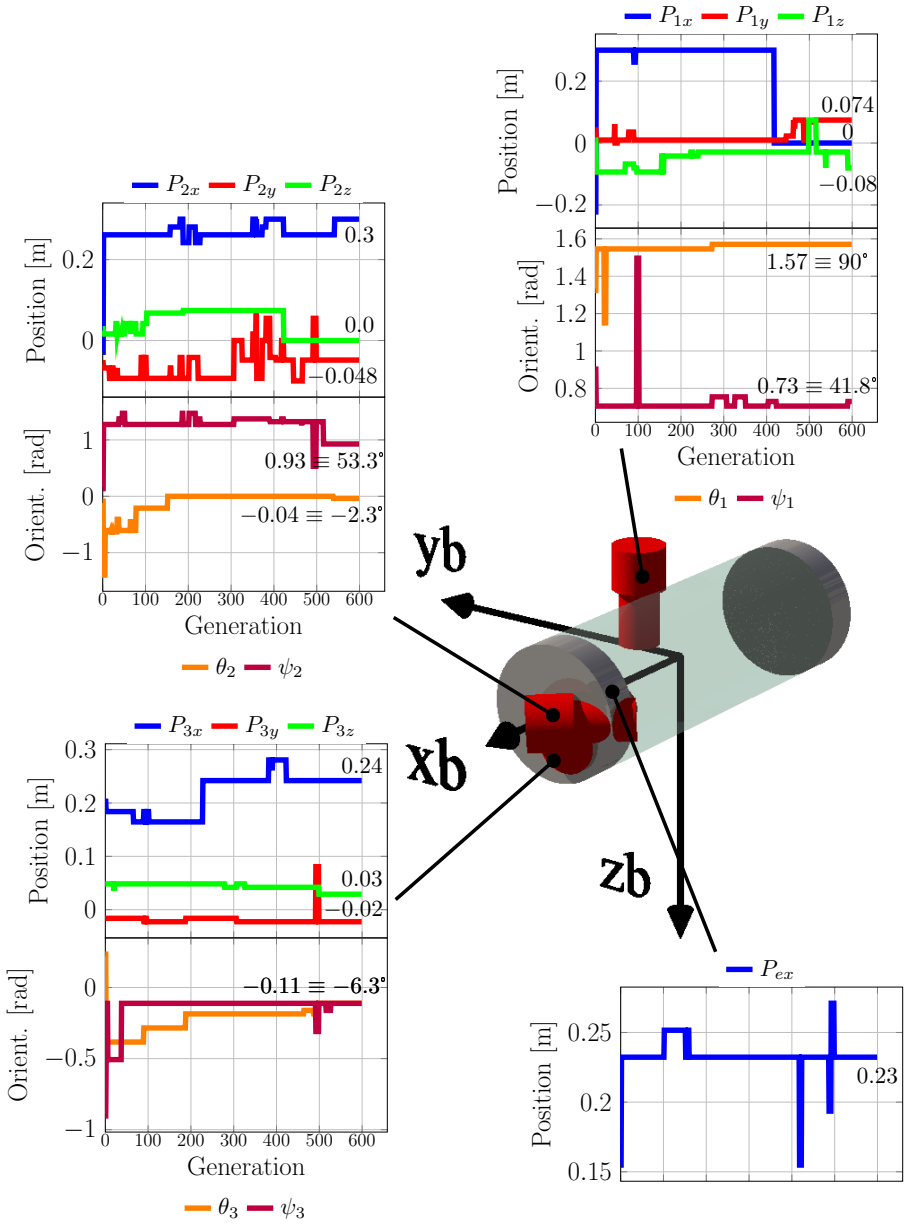

Figure 6.32: Final configuration for diving phase after global optimization.

Given that the success of the mission depends on following the position of the straight line, the genetic algorithm only care about the x, y and z position. The found solution is one that neglects the creation of *pitch*. Even if a *pitch* angle dive reduces the hydrodynamic forces over the robot, the control found by the algorithm still manages to follow the trajectory with a

reduced power consumption.

Based on the geometry of the tasks, the two most important degrees of freedom to control are *surge* and *heave*. We can see how the first degree of freedom is controlled mainly by thruster three. Thruster two can generate a component of force on x_b but its main contribution is the correction over y_b thanks to the generation of *yaw*.

Finally, thruster number one is only there to control *heave*. Its position creates a small moment arm to generate *roll*, but its vertical orientation shows that its main contribution to the topology is the control over the vertical displacement.

The e point is placed in the front of the robot, showing the preponderant direction of motion. From this position it benefits of the *yaw* created by thruster number two and three in the front. Lastly, the gain values are $\Lambda = 2.0$ and $K_p = 0.50$, making the robot react rapidly to errors in position, but loosely compensating the dynamic effects over the robot.

Figure 6.33 shows the thrusters force and the power consumed by the propulsion system throughout the task. Not surprisingly, thruster one (vertical) is the one generating most of the force. Thruster three produces the forward motion and thruster two is in charge of lateral corrections, specially in the beginning of the simulation, before the robot reaches the trajectory.

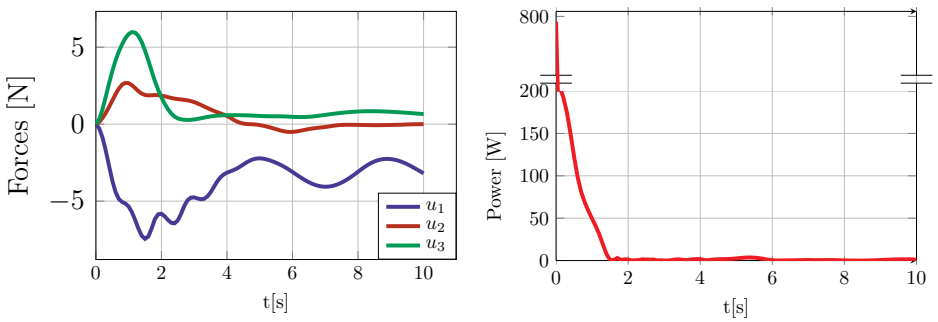

Figure 6.33: Forces and power consumption during the diving task.

6.4.3.3 Optimization for tomography of water turbine

In the last stage of the mission, the robot arrives to the water turbine. At this point it begins the phase of inspection of the energy generator. In order to detect signs of deterioration, it needs to turn around the turbine making sure that the camera is constantly pointing toward the center of the circle (target position) and its distance from the target is constant.

For this specific phase, not only the position of the robot is important, but also its yaw orientation ψ. If the robot does not head toward the turbine, the inspection could be unfruitful. This is why we have added a term related to the *yaw* orientation of the robot in the fitness function. Now, the performance of the robot will be calculated taking into account the position and yaw angle error of the robot. Mathematically, it means that the objective function now will add a term with the normalized integration of the *yaw* angle:

$$g(x) = \frac{\int_0^t e_k \, dt}{N_{e\,kin}} + \frac{\int_0^t e_\psi \, dt}{N_\psi} + \frac{\int_0^t P \, dt}{N_{energ}} \tag{6.32}$$

Where e_ψ is the *yaw* error and N_ψ the normalizing factor. This factor is calculated as the integral of the *yaw* error for a static robot (starting position).

Figure 6.34 shows the evolution of the fitness value. We can see that despite the new added constraint (yaw angle error), the algorithm effectively finds a suitable individual for this task. However, it is also true that this time the ascending phase of the *best fitness* curve has been enlarged, with smaller increments per generation.

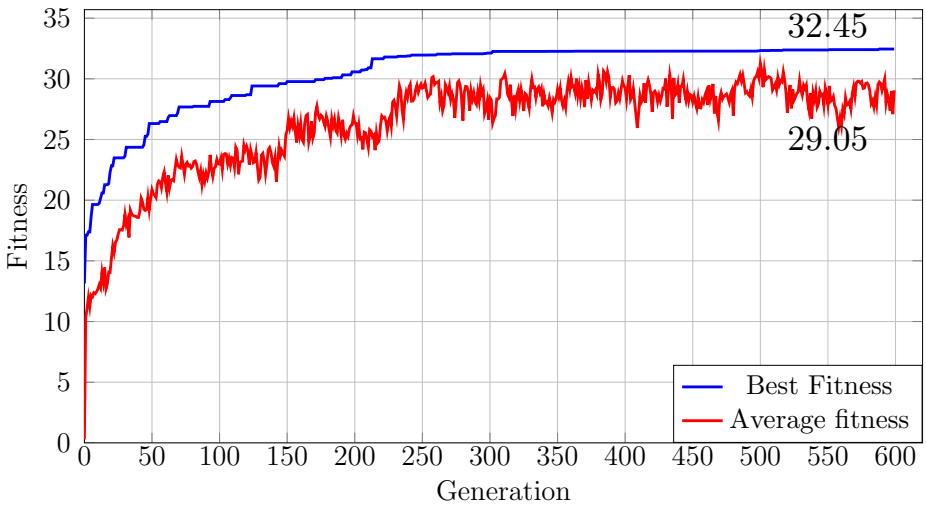

Figure 6.34: Fitness evolution for tomography phase.

Figure 6.35 shows the simulation of the task and the evolution of the kinematic and *yaw* error. We can note the slow but steady convergence of the position during the first stage of the simulation, which stabilizes with an error inferior to 10cm. The *yaw* exhibits a faster convergence, and stabilizes with an error inferior to $0.1\,rad$ (0.57°).

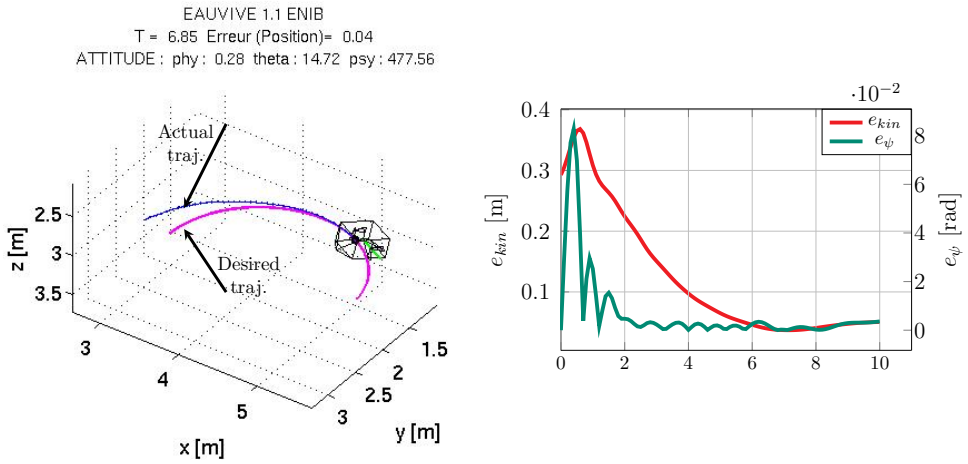

Figure 6.35: EAUVIVE simulation and kinematic error evolution for tomography phase.

The found solution is shown in figure 6.36, after a brief inspection we can observe that the topology is somehow not surprising. In order to perform this mission, the degrees of freedom needed to control the position and orientation of the e point are *surge*, *sway*, *heave* and *yaw*. From the figure we can see how these four motions are created by the thrusters.

From the position and orientation of thrusters one, three and four we can see that the motions to control are the lateral ones, *sway* and *yaw*. Indeed, all of these three thrusters have a component of the force over y_b, which allows to create *sway*. Thrusters number three and four are also able to create *yaw*, thanks to their moment arm.

Thruster number three also participates in the creation of *surge*, since this degree of freedom only requires small corrections in this task, only a component of the force over x_b is enough to control it.

Thruster two is in charge mainly of creating *heave*, as with *surge*, the necessary corrections are small which makes one thruster enough to control it. This thruster also participates in the creation of *surge*.

The position of the point e confirms the importance of *yaw* in this phase of the mission. Indeed, the fact that it is positioned close to the center, shows how *yaw* motion is one of the dominant degrees of freedom of this task. The closeness to the center of the coordinate frame is used by the controller to create a pure *yaw* motion without spurious coupled thrusts.

Finally, the controller gains are $\Lambda = 0.25$ and $K_p = 16.06$, which makes the robot to slowly converge to the desired trajectory but with a heavy compensation of the dynamic effects (arisen by presence of yaw rotation).

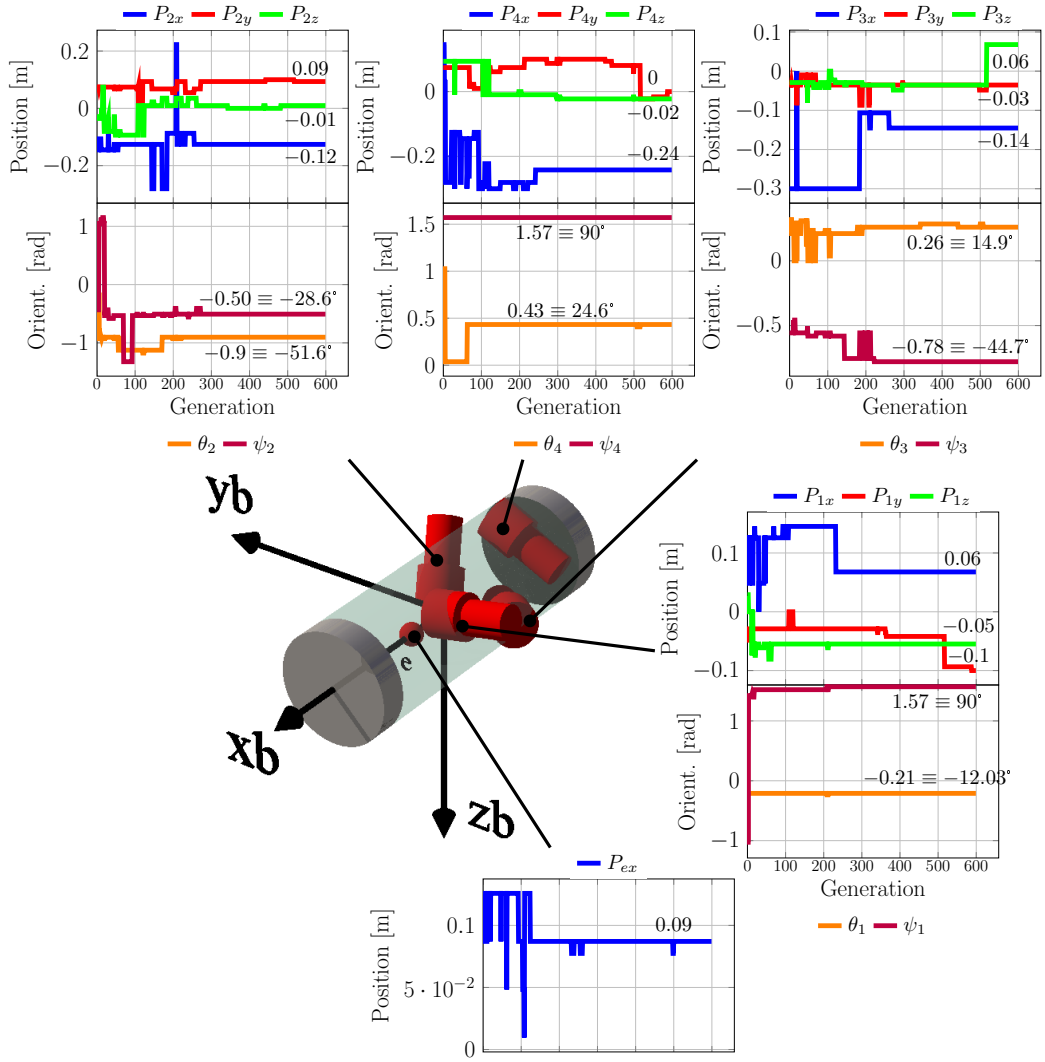

Figure 6.36: Final configuration for tomography task.

Figure 6.37 shows the propulsive forces and consumed power throughout the task. The robot having a very exigent dynamic controller, demands full force from the thrusters in the transient zone (before reaching the task), this creates peaks in a similar way as in the beginning of the simulation. As discussed before, these high peaks have a very short duration and have little effect over the dynamic of the robot and the integration of the consumed power. Given that the robot mostly uses *surge* and *yaw* in order to perform this task, it is not surprising that thrusters number one and four are the ones generating most of the force. Indeed, the actions of these thrust have a direct impact in the concerned degreed of freedom (thruster one on *sway*

and thruster four on *yaw*). The other two thrusters are in charge of small corrections and therefore do not generate great forces.

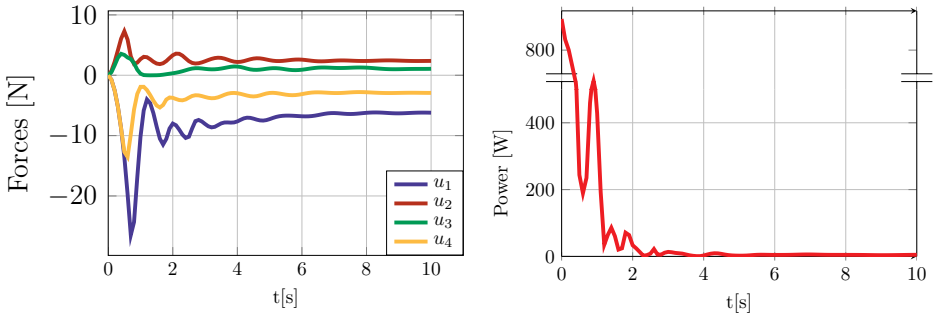

Figure 6.37: Forces and power consumption during the tomography task.

6.5 Dynamic configuration

The global optimization method allows us to obtain promising results in terms of performance. We can find a suitable robot for a given task, which is our main goal. However, giving the GA the complete liberty of searching the best configuration for each single task could generate impractical solutions (the same robot must perform several task in the same mission). Predicting and optimizing all possible tasks appear to be impossible. The other option is to reconfigure dynamically the robot, which means it will change its propulsion configuration during the mission, while it is under the sea.

Nowadays, propulsion technology is not capable of providing full reconfiguration capabilities. Reconfiguration of the orientation of the thrust seems to be giving its first steps out of infancy as seen in section 3.1.3.2, but dynamic (during mission) reconfiguration of the position of the thrusters has not yet received interest.

Taking into account these restrictions, it seems interesting to test the algorithm in a more *realistic* scenario. To do this, we will try to find suitable solutions for the same inspection mission but without moving thrusters (positions are fixed) or changing their number (we keep RSM parameters, i.e. 4 fixed thrusters). Only the thrusters orientations and control parameters will be optimized. This solution is realistic because it can be implemented using vectorial thrusters such as the RMCT studied in section 4.2.3. We will continue using the *topology factor*, in order to compare the fitness of the dynamic configuration to the one of the global optimization. For these optimizations the evolution parameters are shown in Table 6.12.

Table 6.12: Evolutionary parameters for dynamic configuration.

Parameter	Symbol	Numerical value
Population size	μ	40
Number of generations	g	max 500
Number of parameters	n_p	25
Number of coding bits (thrust. or.)	k_{pos}	5
Number of coding bits (thrust. pos.)	k_{or}	7
Number of coding bits (gains)	k_{gains}	8
Number of coding bits (*e.* pos.)	k_{e}	5
Size of genotype	s_{gen}	147
Crossover probability	pc	0.5
Mutation probability	pm	5.74×10^{-3}

6.5.1 Seabed inspection

This time, the achievement of the optimization will be greatly limited since the GA will only be able to change the orientation of the four thrusters and the control parameters.

However, despite the constraints, the algorithms finds a solution comparable in performance with the global one. The final fitness value is 35.52, as seen in Fig. 6.38. This reduction could be due, evidently, to the fact that less parameters can be changed, but also due to the fact that the number of thrusters is set to four, changing the topology factor to 0.9.

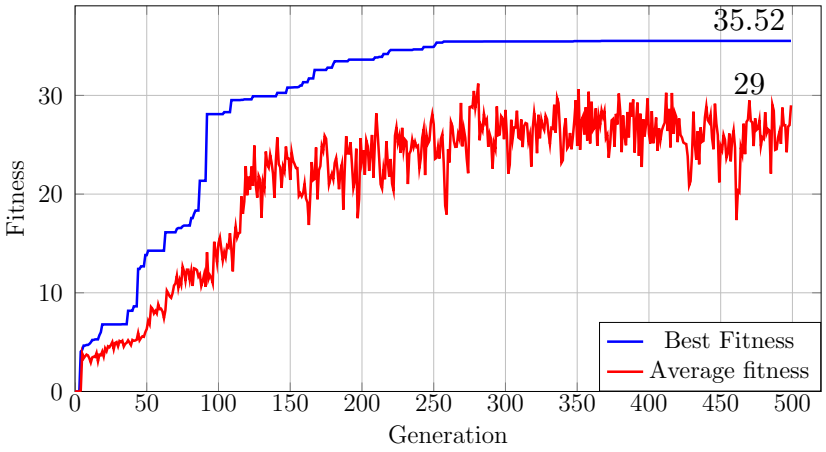

Figure 6.38: RSM robot fitness for seabed inspection (dynamic conf.).

Figure 6.39 shows the simulation and the evolution of the kinematic error. The complexity of the tasks is observed in the evolution of the error. Indeed,

we can see how, despite being small throughout the simulation, the error has strong variations during the turning phases of the task.

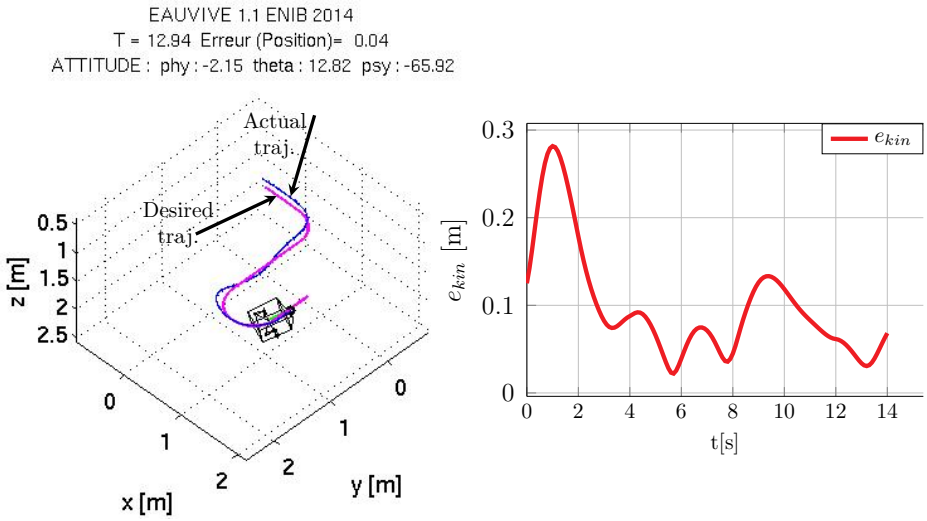

Figure 6.39: EAUVIVE simulation and kinematic error evolution for seabed scanning phase (Dyn. conf.).

Figure 6.40 shows the configuration of the robot. We can see that thrusters one, two and four are able to create *yaw*. Which is important to be able to steer the vehicle during the turning phases. We see as well how *surge* is now created by thrusters two and four. Even if they are on the same side, the *yaw* perturbation gets compensated by the orientation of thruster two. Additionally, thruster three gets to act over y_b in order to strengthen the lateral trajectory control.

All four thrusters have a component over z_b, allowing them to intervene in the control of *heave* and *roll*. The lack of dedicated thrusters in order to fulfill this duty is due to the small control effort needed to correct these two degrees of freedom for this robot.

The position of e is not surprising. The main movement of the robot is to advance, so it is expected to be located in the front of the robot. Allowing to profit from the moment arm from its position to the one of the thrusters.

As with the global controller gains, this time the GA has found a somehow loose kinematic gain ($\Lambda = 1$) and a tighter control over the the dynamic effects on the robot ($K_p = 1.5$).

Figure 6.41 shows that, as in the global optimization for this task, all thrusters have a very active participation in the control of the vehicle. Indeed, we can see how the thrust forces are constantly variating, specially during

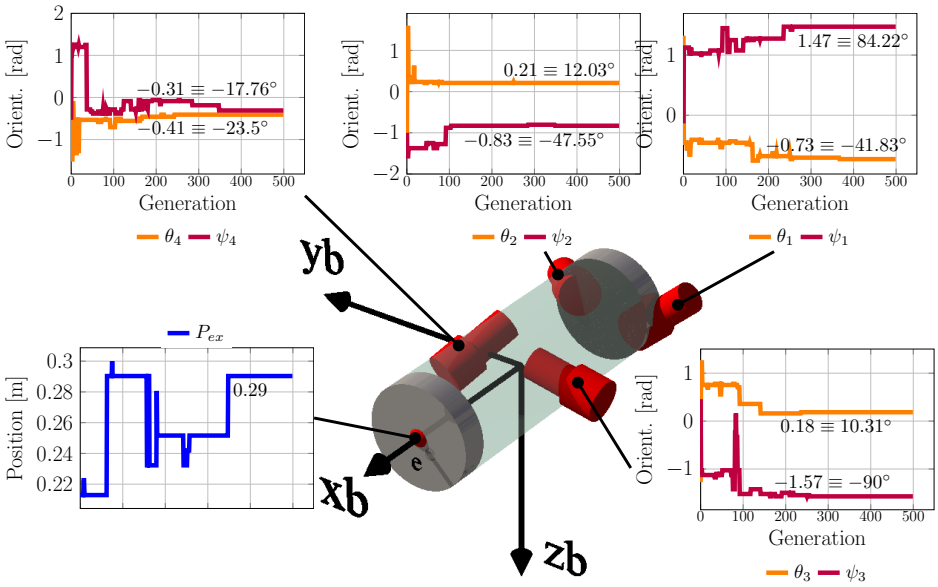

Figure 6.40: RSM robot for seabed inspection (dynamic conf.).

the turning phase, which creates a noticeable increment in the the consumed power.

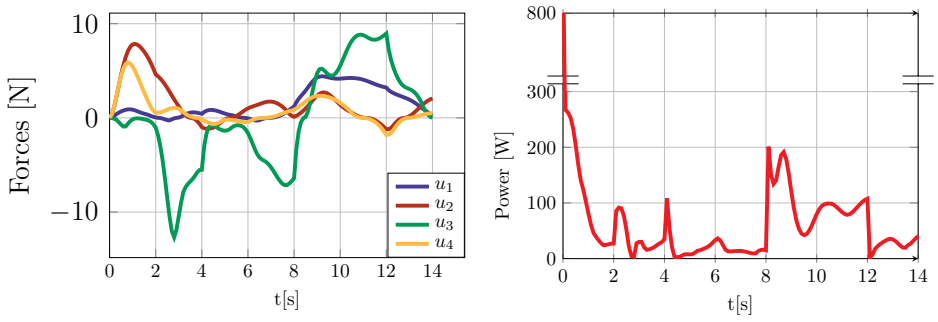

Figure 6.41: Forces and power consumption during the seabed scanning phase (dyn. conf.).

6.5.2 Diving toward water turbine

In this transition phase, the vehicle needs to dive preferably with a *pitch* angle in order to reduce drag forces. This is a type of mission that the RSM robot was not originally designed for. In the global optimization application we have seen that the found configuration involved not controlling the *pitch*

angle and just focusing in following the position of the trajectory. For this optimization we give again the possibility to the algorithm to find creative ways of performing the mission.

Figure 6.42 shows the fitness of the optimization. Despite the added difficulty of the problem (recreating the same movements than before but with less parameters to modify), we see that the algorithm does not take longer until it begins to consistently improve the individuals of the population.

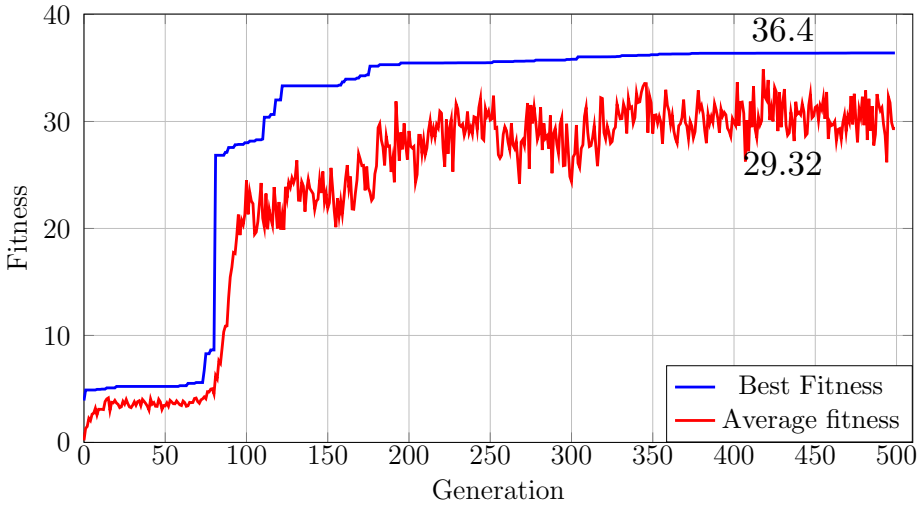

Figure 6.42: RSM robot fitness for the diving phase (dynamic conf.).

Simulation and the evolution of the kinematic error are shown in Figure 6.43. From the kinematic error evolution we can note how the task is straightforwardly executed with minimal variations in the error after the desired trajectory is reached. Additionally, we can also observe that this time the GA has found a solution using *pitch* during the diving.

In figure 6.44 we see the solution found by the GA. The four degrees of freedom needed to perform this task are *surge*, *heave*, *pitch* and *yaw*. From the figure we can deduce how the motions are created.

The *surge* and *yaw* motions are created mainly by thrusters one and four. These thrusters take in charge the advancement and lateral correction of the trajectory of e. For their part, thrusters two and three are in charge of the remaining degrees of freedom: *heave* and *pitch*, following a similar arrangement as the other two actuators. The inverted configuration of the thrusters gives stability while correctly controlling the degrees of freedom.

The point e is positioned in the front of the robot, this gives thrusters one and four a better moment arm to create *yaw*. This position also helps thrusters two and three create *pitch*. Controller gains are $\Lambda = 0.5$ and

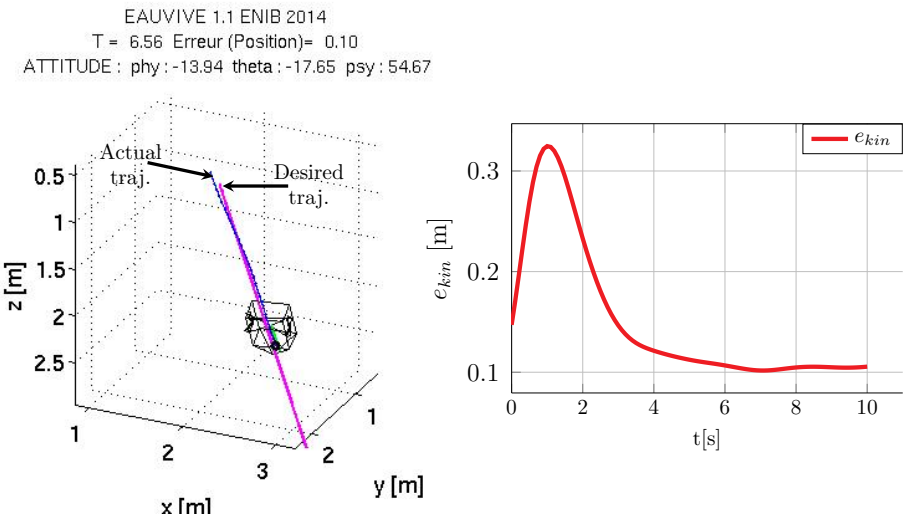

Figure 6.43: EAUVIVE simulation and kinematic error evolution for diving phase (Dyn. conf.).

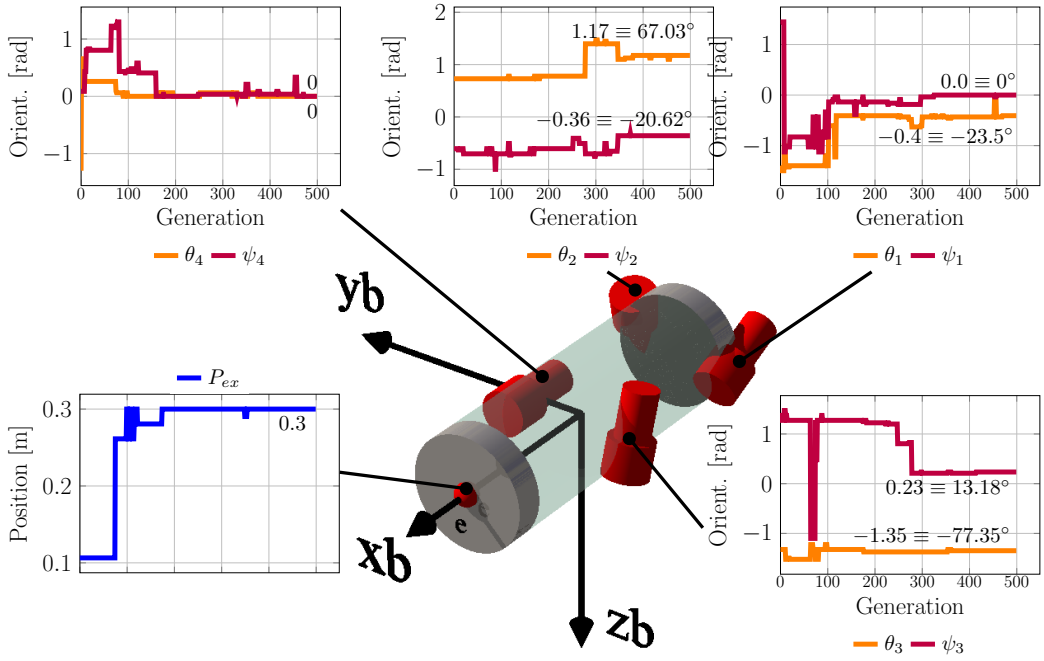

Figure 6.44: Final robot for diving phase (dynamic conf.).

$K_p = 2.76$, this means that the kinematic controller slowly reacts to the error but makes the dynamic controller rapidly compensate the dynamic effects.

Thrust forces during the simulation and the evolution of the consumed

power are shown in Figure 6.45. We can see how the creation of *pitch* by thruster number two and *heave* by thruster number three are very demanding in terms of force. Indeed, in order to create *pitch*, thruster number two needs to fight against the restoring forces, while thruster three needs to create the diving force. Thrusters one and four are used to give small lateral corrections on the trajectory.

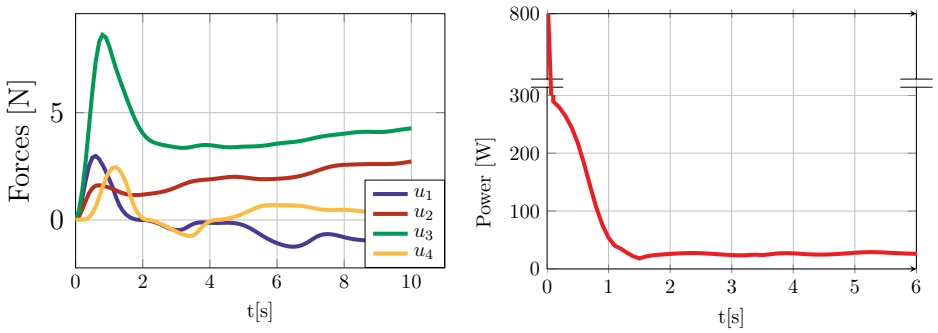

Figure 6.45: Forces and power consumption during diving phase (dyn. conf.).

6.5.3 Tomography

For the last part of the mission, the RSM robot needs to adapt its thrusters to perform a task that, as seen in the global evolution section, involves mostly *sway* and *yaw*.

Figure 6.46 shows the fitness throughout the evolution. We can see how the GA smoothly converged to a solution despite the difficulty of the problem. In fact, if we compare with the result of the global optimization for the same task, the fitness in this case is slightly superior. This is due to the search space reduction, now the genetic algorithm does not loose time evaluating unuseful topologies. In the global optimization, the GA had to first find promising topologies and then optimize their configuration. The fact that the global approach found a four thrusters topology shows how, by reducing the problem to the RSM robot configuration, we have spared the GA of great part of the search effort (less time used for exploration and more exploitation).

Figure 6.47 shows a snapshot of the simulation and the evolution of the kinematic error. As for the global optimization of this task, the robot converges slowly but with a steady pace. Once the AUV has reached the desired trajectory, the kinematic and *yaw* errors remain small ($e_{kin} < 0.1m$ and $e_{\psi} < 1.4°$).

The configuration found by the GA is shown in Fig. 6.48. As stated in the application of the global optimization, the necessary degrees of freedom

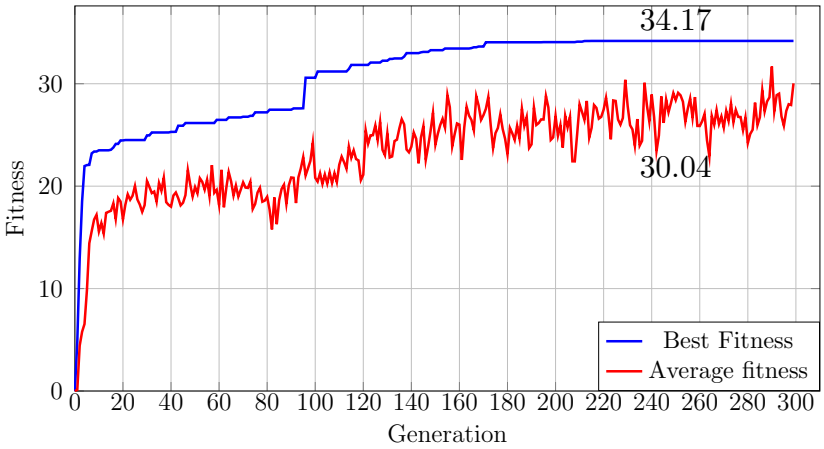

Figure 6.46: Fitness evolution for tomography phase (dynamic conf.).

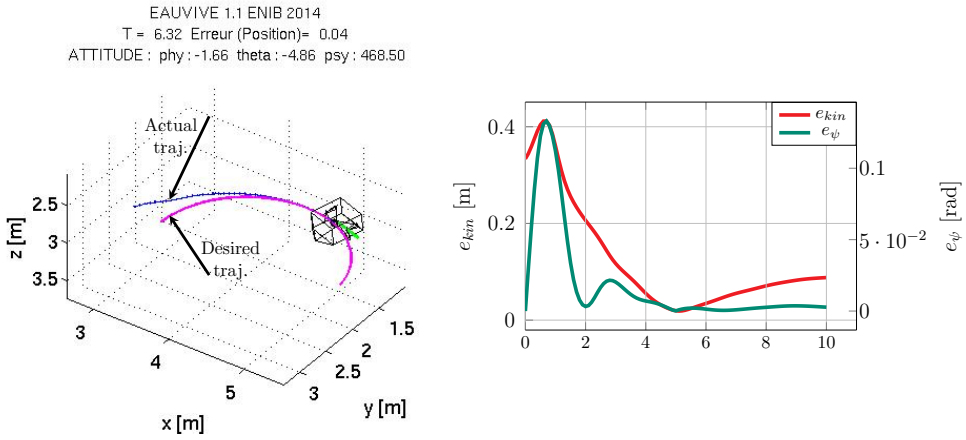

Figure 6.47: EAUVIVE simulation and kinematic error evolution for tomography phase (Dyn. conf.).

to control in order to perform this trajectory are *surge*, *sway*, *heave* and *yaw*. In the figure we see how all thrusters have a force component over the y_b axis, which is understandable since it is the main direction of the motion. Thrusters one and two are, like in the RSM robot, in charge of the *yaw* movements. However, we see that only thruster two contributes to the *surge* motion while thruster one generates *heave* and *pitch*. For its part, thruster three generates mainly a lateral force, controlling *sway*. Lastly, *heave* is generated by a composition of the forces over the z_b from thrusters one and four.

The main movements being *sway* and *yaw*, so the position of e was to

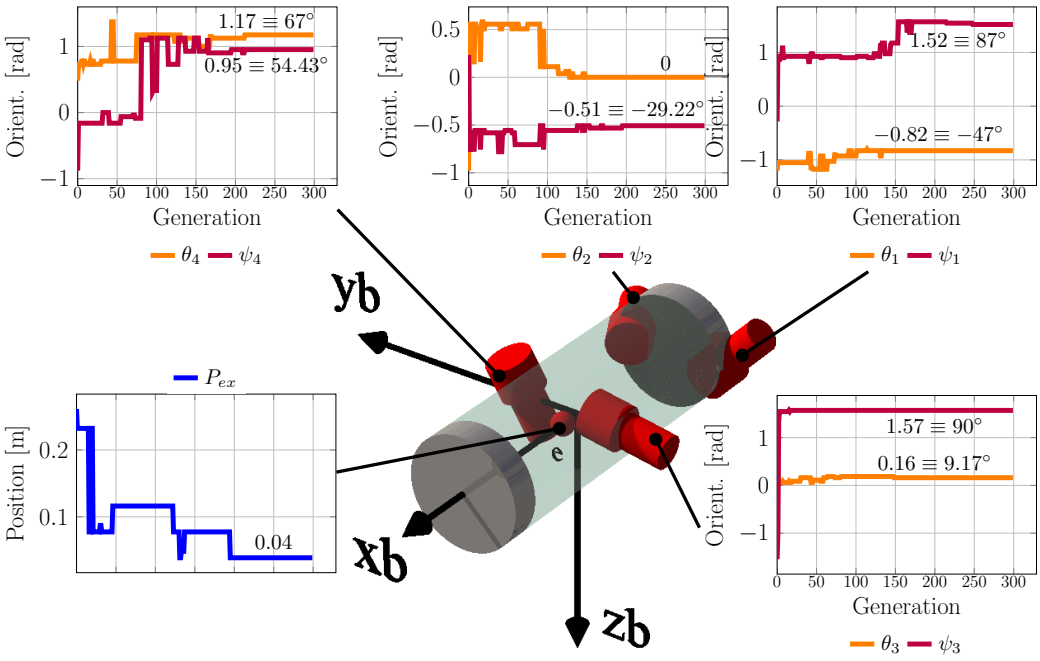

Figure 6.48: Final robot configuration for the tomography phase (dynamic conf.).

be expected. The position of the tracking point, close to the R_b origin, favors the control over *yaw*. It gives the thrusters in charge of this degree of freedom a long enough lever arm, which is also short enough to limit the *overshoot* due to propeller dynamics. The GA chose again a small kinematic gain ($\Lambda = 0.25$) making the robot slowly converge to the task, however, it rapidly compensates the dynamic effects on the robot thanks to its dynamic gain value ($K_p = 9.28$).

Figure 6.49 shows thruster forces and power consumption throughout the task. As for the global optimization of this task, the strong dynamic compensation of the controller demands full force from the thrusters, generating in the force peaks of short duration. Once the robot has reached the desired trajectory, we see how thruster number three generates the higher force in order to move the AUV laterally. Thruster number two is the second biggest force contributor, taking in charge the creation of *yaw*. Thrusters one and four have a small contribution, they are mostly in charge of controlling depth.

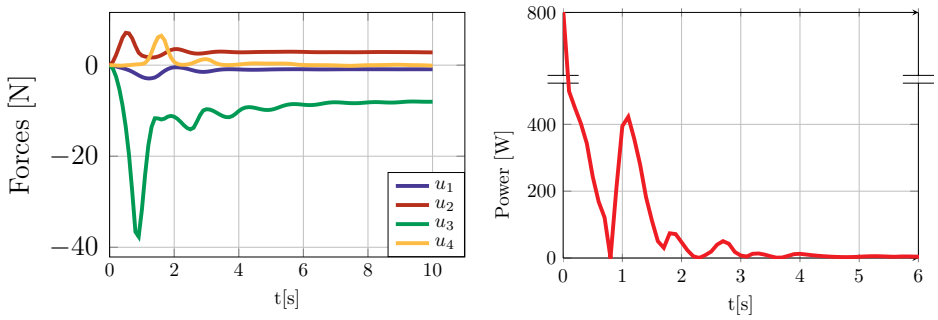

Figure 6.49: Forces and power consumption during tomography(dyn. conf.).

6.6 Conclusion

Analyzing the results of the dynamic configuration and global optimizations (Table 6.13), we can see how the global approach tends to have better results. For the seabed scanning, the fitness of the best individual found by the global approach was 1.35 times better than the one found by the dynamic configuration optimization. We obtain a similar outcome analyzing the two approaches for the diving task, the global optimization best individual is 1.38 times fitter than the one found by the dynamic configuration. However, for the tomography task, the dynamic configuration finds a better suited robot than its global counterpart (1.05 times fitter), probably due to the reduced search space and to a starting topology (RSM robot) well suited for the task.

Table 6.13: Results obtained with dyn. conf and global opt.

Approach	**Task**	**Best fitness**
Global opt.	Scanning of seabed	47.97
	Diving toward turbine	50.2
	Tomography of turbine	32.45
Dyn. configuration	Crossover probability	35.52
	Scanning of seabed	36.4
	Diving toward turbine	34.17

Solely based on the fitness of the best individual, it is clear that global optimization is to be preferred. However from a technological point of view, dynamic configuration seems to be more promising. Indeed, even if the performances of this approach are not as good as the global ones, the fact that they are technically feasible makes the dynamic configuration more useful.

In section 6.5 the RSM robot propulsion system, with predefined thruster number and positions, was optimized to follow three different tasks included

in a robotic mission. The three optimized configurations maximize the performance of the robot in each task. A version of the RSM robot capable of dynamically changing its propulsive configuration (vectorial thrusters) would be able to achieve the three tasks in a row (i.e. the mission) with the best performance. Given the current state of thrust configuration technology, this idea seems applicable in the medium-term.

The advantages of dynamic configuration are clear, it can make underactuated AUVs capable of performing complex and demanding tasks in an optimal way, enhancing the propulsive capabilities of these vehicles. Indeed, AUVs with dynamically reconfigurable thrusters, using propulsion configurations determined by this method, would be able to perform tasks with good trajectory tracking convergence and reduced power consumption. This would result in improving AUVs capabilities and therefore expanding their autonomy.

Chapter 7

Conclusions and perspectives

The design of autonomous underwater vehicles supposes the definition of a myriad of design parameters. Said parameters must be selected and properly configured in order to obtain enhanced performance and autonomy. A way of achieving autonomy in an AUV is increasing its propulsive capabilities, adapting the robot to the tasks it is expected to perform.

Nowadays, AUVs are exposed to increasingly demanding tasks and missions. Far from the survey-style missions that made these vehicles popular, the new missions demand the AUVs to have greater autonomy and efficiency.

Most of AUV design methods rely heavily on designers experience and rules of thumb, taking well known designs and adopting classical approaches. While capable of performing difficult tasks, these robots are not optimized to perform these missions causing a decrease in their overall performance and autonomy. Also their efficiency is not as high as it could be (using too many thrusters).

This work proposes a way of improving the design of AUVs through the optimization of its propulsive configuration, increasing their performances and therefore enhancing their autonomy. Indeed, *mission-adapted* propulsive configurations allow the AUV to perform otherwise difficult tasks, making them less depending of human intervention.

The optimization technique used in this work uses an AUV simulator in order to evaluate the candidate solutions. This simulator, contains the hydrodynamical model of the robot, which includes rigid body dynamics and hydrodynamic effects such as added mass and damping.

The EAUVIVE simulator also includes the electrodynamic model of a real thruster, which parameters have been identified and validated using a test rig. A vectorial thruster has been studied as well, in order to study its feasibility. This vectorial thruster offers the possibility of changing the direction of the thrust thanks to a magnetic coupling. The dynamic model

of this device was developed and validated.

In order to correctly simulate the behavior of the robot, we use a model-based nonlinear controller called torque computation. This controller uses the kinematic and hydrodynamic model of the AUV in order to compensate the hydrodynamic effects and steer the vehicle toward a desired trajectory. This control technique generates the needed wrench in order to follow the desired trajectory. An additional stage, called thrust allocation, was added in order to determine the needed forces from each thruster of the robot.

Given that the genetic algorithm evaluates different propulsive topologies during its optimization work, a way of adapting the control technique to different propulsive topologies was presented. Additionally, some considerations about the conditions to successfully apply this control adaptation method were discussed.

The extended Kalman filter estimation method and its implantation (along with the control method) into the robot control architecture was discussed. This is not yet used by the genetic algorithm optimization, but it is an unavoidable step in future robotic platform developments.

Lastly, using the aforementioned elements, the genetic algorithm was developed. It allows to, given a robot structure and a defined task, to find an optimal propulsive topology and control parameters. The found solution will be adapted for the demanded task, creating a robot with a good performance but task-specific. Given that in the real world most of the solutions found thanks to this method are difficult to apply, an alternative approach was presented. This approach, called dynamic configuration, proposes to apply the optimization method to a robot in which the position of its thrusters are predefined. Restricting the GA possibilities of research by imposing some of the design parameters can result in less efficient topologies. However, as a compensation for the performance loss, the found solutions are technically viable. Applying the dynamic configuration method with the same constrained parameters but for different tasks, allows the AUV designer to calculate the optimal topology configuration for each task. With this information, and taking into account the advances in reconfigurable thruster technology, the creation of a dynamically reconfigurable robot is possible in a near future. A robot designed using this technique could be optimally adapted to a mission including any number of different tasks.

Future work should focus on the improvement of the genetic algorithm. This can be done increasing its efficiency using more advanced techniques or by narrowing down the search space based on previous experience. This could allow to expand the number of design parameters without increasing the calculation time.

The expanded set of design parameters could include a different type of

controller technique (for instance PID, sliding mode, adaptive methods, etc). Indeed, giving the possibility to the GA of choosing the type of control used in the robot could lead to a improvement of the AUV design. The correction terms in the used computed torque control technique could be improved as well, making it use not only the proportional correction but also the integral and derivative ones.

The genetic algorithm would benefit as well by using improved models in the control technique (for instance including the thruster dynamics in computed torque method). Indeed, a more precise definition of the hydrodynamic terms in the AUV model and a fully dynamic model of the thrusters would help to reduce the reality gap, making the solutions found by the optimization easier to apply in the real world.

Lastly, real experimentations using the RSM robot with the found propulsive topologies should be carried out, in order to validate and tune the solutions found by the artificial evolution.

References

[1] K. Wong, "Wealth of the Oceans," *Open Hydrol. J.*, 2015.

[2] T. P. Barnett, D. W. Pierce, K. M. Achutarao, P. J. Gleckler, B. D. Santer, J. M. Gregory, and W. M. Washington, "Penetration of human-induced warming into the world's oceans." *Science*, vol. 309, no. 5732, pp. 284–7, jul 2005.

[3] G. Antonelli, *Underwater Robots: Motion and Force Control of Vehicle-Manipulator Systems (Springer Tracts in Advanced Robotics)*. Springer-Verlag New York, Inc., jul 2006.

[4] S. McPhail, "Autosub6000: A Deep Diving Long Range AUV," *J. Bionic Eng.*, vol. 6, no. 1, pp. 55–62, mar 2009.

[5] D. Ribas, P. Ridao, A. Turetta, C. Melchiorri, G. Palli, J. J. Fernandez, and P. J. Sanz, "I-AUV Mechatronics Integration for the TRIDENT FP7 Project," *IEEE/ASME Trans. Mechatronics*, vol. 20, no. 5, pp. 2583–2592, oct 2015.

[6] F. Bruno, M. Muzzupappa, A. Lagudi, A. Gallo, F. Spadafora, G. Ritacco, A. Angilica, L. Barbieri, N. Di Lecce, G. Saviozzi, C. Laschi, R. Guida, and G. Di Stefano, "A ROV for supporting the planned maintenance in underwater archaeological sites," in *Ocean. 2015 - Genova*. IEEE, may 2015, pp. 1–7.

[7] Y. Tsusaka, H. Ishidera, and Y. Itoh, "MURS-300 MK II: A remote inspection system for underwater facilities of hydraulic power plants," *IEEE J. Ocean. Eng.*, vol. 11, no. 3, pp. 358–363, jul 1986.

[8] D. Liddle, "TROJAN: Remotely operated vehicle," *IEEE J. Ocean. Eng.*, vol. 11, no. 3, pp. 364–372, jul 1986.

[9] M. Nokin, "Victor 6000-a deep teleoperated system for scientific research," in *Ocean. '97. MTS/IEEE Conf. Proc.*, vol. 1. IEEE, 1997, pp. 167–171.

[10] D. Lynn and G. Bohlander, "Performing ship hull inspections using a remotely operated vehicle," in *Ocean. '99. MTS/IEEE. Rid. Crest into 21st Century. Conf. Exhib. Conf. Proc. (IEEE Cat. No.99CH37008)*, vol. 2. IEEE & Marine Technol. Soc, 1999, pp. 555–562.

[11] S. Sona, "Advancements in ROV and trenching technology," in *Ocean. 2000 MTS/IEEE Conf. Exhib. Conf. Proc. (Cat. No.00CH37158)*, vol. 1. IEEE, 2000, pp. 505–508.

[12] H. Momma, M. Watanabe, K. Mitsuzawa, K. Danno, M. Ida, M. Arita, and I. Ujino, "Search and recovery of the H-II Rocket Flight No. 8 engine," in *Proc. 2000 Int. Symp. Underw. Technol. (Cat. No.00EX418)*. IEEE, 2000, pp. 19–23.

[13] J. Refsnes and A. Sorensen, "Design of control system of torpedo shaped ROV with experimental results," in *Ocean. '04 MTS/IEEE Techno-Ocean '04 (IEEE Cat. No.04CH37600)*, vol. 1. IEEE, 2004, pp. 264–270.

[14] D. L. Stein, J. D. Felley, and M. Vecchione, "ROV observations of benthic fishes in the Northwind and Canada Basins, Arctic Ocean," *Polar Biol.*, vol. 28, no. 3, pp. 232–237, jan 2005.

[15] J. Yuh, "Design and Control of Autonomous Underwater Robots: A Survey," *Autnomous Robot.*, vol. 8, no. 1, pp. 7–24, 2000.

[16] P. E. Hagen, O. Midtgaard, and O. Hasvold, "Making AUVs Truly Autonomous," in *Ocean. 2007*. IEEE, 2007, pp. 1–4.

[17] J. Nicholson and A. Healey, "The present state of autonomous underwater vehicle (AUV) applications and technologies," *Mar. Technol. Soc. J.*, 2008.

[18] P. Stevenson, M. Furlong, and D. Dormer, "AUV shapes - Combining the Practical and Hydrodynamic Considerations," in *Ocean. 2007 - Eur.* IEEE, jun 2007, pp. 1–6.

[19] M. Jacobi and D. Karimanzira, "Underwater pipeline and cable inspection using autonomous underwater vehicles," in *2013 MTS/IEEE Ocean. - Bergen*. IEEE, jun 2013, pp. 1–6.

[20] C. Kaminski, T. Crees, J. Ferguson, A. Forrest, J. Williams, D. Hopkin, and G. Heard, "12 days under ice – an historic AUV deployment in the Canadian High Arctic," in *2010 IEEE/OES Auton. Underw. Veh.* IEEE, sep 2010, pp. 1–11.

[21] E. Bovio, "Autonomous underwater vehicles for port protection," *Proc. Int. Conf. New ...*, 2005.

[22] P. Hagen, N. Storkersen, K. Vestgard, and P. Kartvedt, "The HUGIN 1000 autonomous underwater vehicle for military applications," in *Ocean. 2003. Celebr. Past ... Teaming Towar. Futur. (IEEE Cat. No.03CH37492)*, vol. 2. IEEE, 2003, pp. 1141–1145 Vol.2.

[23] K. Stansfield, D. A. Smeed, G. P. Gasparini, S. McPhail, N. Millard, P. Stevenson, A. Webb, A. Vetrano, and B. Rabe, "Deep-sea, high-resolution, hydrography and current measurements using an autonomous underwater vehicle: The overflow from the Strait of Sicily," *Geophys. Res. Lett.*, vol. 28, no. 13, pp. 2645–2648, jul 2001.

[24] A.-R. Diercks, V. L. Asper, M. Woolsey, J. L. Williams, and F. Cantelas, "NIUST AUV ' s study Shipwrecks in the Northern Gulf of Mexico," in *2010 IEEE/OES Auton. Underw. Veh.*, no. 1. IEEE, sep 2002, pp. 1–5.

[25] F. Fan, "A potential method for underwater charging," in *Ocean. 2015 - Genova*. IEEE, may 2015, pp. 1–5.

[26] G. Griffiths and J. Jamieson, "Energy storage for long endurance AUVs," *ATUV Int. ...*, 2004.

[27] V. Djapic, J. A. Farrell, P. Nfiller, and R. Arrieta, "Design and initial in-water testing of advanced non-linear control algorithms onto an Unmanned Underwater Vehicle (UUV)," in *2007 Ocean. Vols 1-5*. IEEE, 2007, pp. 421–428 2080.

[28] Q. S. Nguyen, S. Heo, H. C. Park, and D. Byun, "Thrust improvement of an fish robot actuated by compressed unimorph piezoelectric composite actuator," in *2009 IEEE Int. Conf. Robot. Biomimetics*. IEEE, dec 2009, pp. 1603–1608.

[29] C. Barngrover, R. Kastner, T. Denewiler, G. Mills, C. Barngrover, R. Kastner, T. Denewiler, and G. Mills, "The stingray AUV: A small and cost-effective solution for ecological monitoring," *Ocean. 2011*, pp. 1–8, 2011.

[30] M. Dunbabin, J. Roberts, K. Usher, G. Winstanley, and P. Corke, "A hybrid AUV design for shallow water reef navigation," in *Proc. - IEEE Int. Conf. Robot. Autom.*, vol. 2005. IEEE, 2005, pp. 2105–2110.

[31] M. Senthil Kumar, S. Chidambara Raja, M. Sarath Kumar, B. Gowthamraj, and R. Ramesh Raja, "A synergetic approach to the conceptual design of Autonomous Underwater Vehicle," *Rob. Auton. Syst.*, vol. 67, pp. 105–114, 2015.

[32] X. Lin and S. Guo, "Development of a spherical underwater robot equipped with multiple vectored water-jet-based thrusters," *J. Intell. Robot. Syst.*, vol. 67, no. 3-4, pp. 307–321, jan 2012.

[33] B. Allen, R. Stokey, T. Austin, N. Forrester, R. Goldsborough, M. Purcell, and C. von Alt, "REMUS: a small, low cost AUV; system description, field trials and performance results," *Ocean. '97. MTS/IEEE Conf. Proc.*, vol. 2, pp. 994–1000, 1997.

[34] N.-h. Tran, M.-M. Woo, H.-s. Choi, and J.-Y. Kim, "Development of a new underwater disk robot," in *2012 Ocean. - Yeosu*. IEEE, may 2012, pp. 1–9.

[35] D. Paster, "Importance of Hydrodynamic Considerations for Underwater Vehicle Design," in *Ocean. '86*. IEEE, 1986, pp. 1413–1422.

[36] W. H. Wang, X. Q. Chen, A. Marburg, J. G. Chase, and C. E. Hann, "A Low-Cost unmanned underwater vehicle prototype for shallow water tasks," in *2008 IEEE/ASME Int. Conf. Mechatronics Embed. Syst. Appl. MESA 2008*. IEEE, oct 2008, pp. 204–209.

[37] C. T. Ross, "A conceptual design of an underwater vehicle," *Ocean Eng.*, vol. 33, no. 16, pp. 2087–2104, 2006.

[38] A. Alvarez, V. Bertram, and L. Gualdesi, "Hull hydrodynamic optimization of autonomous underwater vehicles operating at snorkeling depth," *Ocean Eng.*, vol. 36, no. 1, pp. 105–112, 2009.

[39] K. Alam, T. Ray, and S. G. Anavatti, "Design and construction of an autonomous underwater vehicle," *Neurocomputing*, vol. 142, pp. 16–29, 2014.

[40] M. a. Martz and A. Brown, "Preliminary Design of an Autonomous Underwater Vehicle using a Multiple- Objective Genetic Optimizer Vehicle using a Multiple-Objective Genetic Optimizer," 2008.

[41] P. Jagadeesh, K. Murali, and V. Idichandy, "Experimental investigation of hydrodynamic force coefficients over AUV hull form," *Ocean Eng.*, vol. 36, no. 1, pp. 113–118, 2009.

[42] D. Perrault, N. Bose, S. O'Young, and C. D. Williams, "Sensitivity of AUV added mass coefficients to variations in hull and control plane geometry," *Ocean Eng.*, vol. 30, no. 5, pp. 645–671, 2003.

[43] A. B. Phillips, S. R. Turnock, and M. Furlong, "The use of computational fluid dynamics to aid cost-effective hydrodynamic design of autonomous underwater vehicles," *Proc. Inst. Mech. Eng. Part M J. Eng. Marit. Environ.*, vol. 224, no. 4, pp. 239–254, nov 2010.

[44] T. Sarkar, P. G. Sayer, and S. M. Fraser, "A Study of Autonomous Underwater Vehicle Hull Forms Using Computational Fluid Dynamics," *Int. J. Numer. Methods FLuids*, vol. 25, no. April 1996, pp. 1301–1313, dec 1997.

[45] C. Holden and K. Y. Pettersen, "Robust Globally Exponentially Stabilizing Control Law for Fully Actuated 6-Dof Auvs," *IFAC Proc. Vol.*, vol. 40, no. 17, pp. 343–348, 2007.

[46] P. Herman, "Decoupled PD set-point controller for underwater vehicles," *Ocean Eng.*, vol. 36, no. 6-7, pp. 529–534, 2009.

[47] O.-E. Fjellstad and T. Fossen, "Position and attitude tracking of AUV's: a quaternion feedback approach," *IEEE J. Ocean. Eng.*, vol. 19, no. 4, pp. 512–518, 1994.

[48] L. L. Whitcomb, "Underwater robotics: out of the research laboratory and into the field," *Proc. 2000 ICRA. Millenn. Conf. IEEE Int. Conf. Robot. Autom. Symp. Proc. (Cat. No.00CH37065)*, vol. 1, pp. 1–8, 2000.

[49] R. Wernli, "AUVs-a technology whose time has come," in *Proc. 2002 Interntional Symp. Underw. Technol. (Cat. No.02EX556)*. IEEE, 2002, pp. 309–314.

[50] B. K. Sahu and S. Member, "The State of Art of Autonomous Underwater Vehicles in Current and Future Decades," in *2014 First Int. Conf. Autom. Control. Energy Syst.*, no. i. IEEE, feb 2014, pp. 1–6.

[51] A. B. Phillips, L. Steenson, E. Rogers, S. R. Turnock, C. Harris, M. E. Furlong, A. Philips, L. Steenson, E. Rogers, S. R. Turnock, C. Harris, and M. E. Furlong, "Delphin2: An over actuated autonomous underwater vehicle for manoeuvring research," *Trans. R. Inst. Nav.*

Archit. Part A Int. J. Marit. Eng., vol. 155, no. PART A4, pp. 171—-180, 2013.

[52] L. V. Steenson, A. B. Phillips, M. E. Furlong, E. Rogers, S. R. Turnock, and S. R. Steenson, L. V., Phillips, A. B., Furlong, M., Rogers, E., Turnock, "Maneuvering of an over-actuated autonomous underwater vehicle using both through-body tunnel thrusters and control surfaces," in *Proc. 17th Int. undersea untethered Submers. Technol. Conf.*, no. August, 2011.

[53] D. Ribas, N. N. Palomeras, P. Ridao, M. Carreras, and A. Mallios, "Girona 500 AUV: From survey to intervention," pp. 46–53, feb 2012.

[54] A. Billings, C. Kaiser, C. M. Young, L. S. Hiebert, E. Cole, J. K. Wagner, and C. L. Van Dover, "SyPRID sampler: A large-volume, high-resolution, autonomous, deep-ocean precision plankton sampling system," *Deep Sea Res. Part II Top. Stud. Oceanogr.*, 2016.

[55] K. Alam, T. Ray, and S. G. Anavatti, "A brief taxonomy of autonomous underwater vehicle design literature," *Ocean Eng.*, vol. 88, pp. 627–630, 2014.

[56] K. Vestgard, R. Hansen, P. E. Conference, P. Engineers, B. Jalving, and O. A. Pedersen, "The HUGIN 3000 Survey AUV," in *ISOPE-2001 Elev. Int. Offshore Polar Eng. Conf.*, vol. IV, 2001, pp. 679–684\r702.

[57] R. Marthiniussen, K. Vestgard, and R. Klepaker, "HUGIN-AUV concept and operational experiences to date," in *Ocean. '04 MTS/IEEE Techno-Ocean '04 (IEEE Cat. No.04CH37600)*, vol. 2. IEEE, pp. 846–850.

[58] N. A. Cruz and A. C. Matos, "The MARES AUV, a Modular Autonomous Robot for Environment Sampling," in *Ocean. 2008.* IEEE, 2008, pp. 1–6.

[59] S. Dupré, G. Buffet, J. Mascle, J.-P. P. Foucher, S. Gauger, A. Boetius, C. Marfia, S. Dupré, G. Buffet, J. Mascle, J.-P. P. Foucher, S. Gauger, A. Boetius, C. Marfia, T. A. A. Q. R. Team, T. A. A. Q. R. Team, and T. B. scientific Party, "High-resolution mapping of large gas emitting mud volcanoes on the Egyptian continental margin (Nile Deep Sea Fan) by AUV surveys," *Mar. Geophys. Res.*, vol. 29, no. 4, pp. 275–290, dec 2008.

[60] V. H. Fernandes, A. A. Neto, and D. D. Rodrigues, "Pipeline inspection with AUV," in *2015 IEEE/OES Acoust. Underw. Geosci. Symp. (RIO Acoust.* IEEE, jul 2015, pp. 1–5.

[61] A. Alvarez, A. Caffaz, A. Caiti, G. Casalino, L. Gualdesi, A. Turetta, and R. Viviani, "Fòlaga: A low-cost autonomous underwater vehicle combining glider and AUV capabilities," *Ocean Eng.*, vol. 36, no. 1, pp. 24–38, 2009.

[62] R. Bachmayer, N. E. Leonard, E. Fiorellit, P. Bhattat, D. Paleyf, J. Graver, E. Fiorelli, P. Bhatta, and D. Paley, "Underwater gliders: recent developments and future applications," in *Proc. 2004 Int. Symp. Underw. Technol. (IEEE Cat. No.04EX869).* IEEE, 2003, pp. 195–200.

[63] N. Stilinovic, D. Nad, and N. Miskovic, "AUV for diver assistance and safety - Design and implementation," in *MTS/IEEE Ocean. 2015 - Genova Discov. Sustain. Ocean Energy a New World.* Genova: IEEE, may 2015, pp. 1–4.

[64] D. Souto, A. Faina, F. Lopez-Pena, and R. J. Duro, "Lappa: A new type of robot for underwater non-magnetic and complex hull cleaning," in *Proc. - IEEE Int. Conf. Robot. Autom.* IEEE, may 2013, pp. 3409–3414.

[65] P. Ridao, M. Carreras, D. Ribas, P. J. Sanz, and G. Oliver, "Intervention AUVs: The next challenge," *Annu. Rev. Control*, vol. 40, pp. 227–241, 2015.

[66] F. Y. Bi, Y. J. Wei, J. Z. Zhang, and W. Cao, "Position-tracking control of underactuated autonomous underwater vehicles in the presence of unknown ocean currents," *Control Theory Appl. IET*, vol. 4, no. 11, pp. 2369–2380, nov 2010.

[67] K. Do, Z. Jiang, J. Pan, and H. Nijmeijer, "Global output feedback universal controller for stabilization and tracking of underactuated ODIN-an underwater vehicle," *Proc. 41st IEEE Conf. Decis. Control. 2002.*, vol. 1, no. December, pp. 504–509, 2002.

[68] J. Refsnes, K. Pettersen, and A. Sorensen, "Control of slender body underactuated AUVs with current estimation," *Proc. 45th IEEE Conf. Decis. Control*, pp. 43–50, 2006.

[69] M. Chyba and R. N. Smith, "A First Extension of Geometric Control Theory to Underwater Vehicles," *IFAC Proc. Vol.*, vol. 41, no. 1, pp. 79–84, 2008.

[70] J. Carlton, *Marine propellers and propulsion.* Butterworth-Heinemann,, 2012.

[71] Y. Tipsuwan and P. Hoonsuwan, "Design and implementation of an AUV for petroleum pipeline inspection," in *2015 7th Int. Conf. Inf. Technol. Electr. Eng.* IEEE, oct 2015, pp. 382–387.

[72] A. Marouchos, B. Muir, R. Babcock, and M. Dunbabin, "A shallow water AUV for benthic and water column observations," in *MTS/IEEE Ocean. 2015 - Genova Discov. Sustain. Ocean Energy a New World.* IEEE, may 2015, pp. 1–7.

[73] J. Pyo, H. Cho, H. Joe, T. Ura, and S. C. Yu, "Development of hovering type AUV "cyclops" and its performance evaluation using image mosaicing," *Ocean Eng.*, vol. 109, pp. 517–530, 2015.

[74] A. J. Healey, S. M. Rock, S. Cody, D. Miles, and J. P. Brown, "Toward an improved understanding of thruster dynamics for underwater vehicles," *IEEE J. Ocean. Eng.*, vol. 20, no. 4, pp. 354–361, 1995.

[75] A. Palmer, G. E. Hearn, and P. Stevenson, "Modelling Tunnel Thrusters for Autonomous Underwater Vehicles," *IFAC Proc. Vol.*, vol. 41, no. 1, pp. 91–96, 2008.

[76] L. Whitcomb and D. Yoerger, "Comparative experiments in the dynamics and model-based control of marine thrusters," *'Challenges Our Chang. Glob. Environ. Conf. Proceedings. Ocean. '95 MTS/IEEE*, vol. 2, no. 5, pp. 1019–1028, 1995.

[77] A. Saunders and M. Nahon, "The effect of forward vehicle velocity on through-body AUV tunnel thruster performance," in *Ocean. '02 MTS/IEEE*, vol. 1. IEEE, pp. 250–259.

[78] S. Abu Sharkh, S. Lai, and S. Turnock, "Structurally integrated brushless PM motor for miniature propeller thrusters," *IEE Proc. - Electr. Power Appl.*, vol. 151, no. 5, p. 513, 2004.

[79] Y. sheng Yang, Y. chun Xie, and S. lin Nie, "Nozzle optimization for water jet propulsion with a positive displacement pump," *China Ocean Eng.*, vol. 28, no. 3, pp. 409–419, jun 2014.

[80] U. A. Korde, "Study of a jet-propulsion method for an underwater vehicle," *Ocean Eng.*, vol. 31, no. 10, pp. 1205–1218, 2004.

[81] A. P. Thomas, M. Milano, M. G. G'Sell, K. Fischer, and J. Burdick, "Synthetic Jet Propulsion for Small Underwater Vehicles," in *Robot. Autom. 2005. ICRA 2005. Proc. 2005 IEEE Int. Conf.*, no. April. IEEE, 2005, pp. 181–187.

[82] L. Jian, L. Xiwen, Z. Zuti, L. Xiaohui, and Z. Yuquan, "Numerical investigation into effects on momentum thrust by nozzle's geometric parameters in water jet propulsion system of autonomous underwater vehicles," *Ocean Eng.*, vol. 123, pp. 327–345, 2016.

[83] E. P. Vega, O. Chocron, J. V. Ferreira, M. Benbouzid, and P. S. Meirelles, "Evaluation of AUV fixed and vectorial propulsion systems with dynamic simulation and non-linear control," in *IECON 2015 - 41st Annu. Conf. IEEE Ind. Electron. Soc.* IEEE, nov 2015, pp. 000 944–000 949.

[84] S. Jin, J. Kim, J. Kim, and T. Seo, "Hovering underwater robotic platform with four tilting thrusters," in *IEEE/ASME Int. Conf. Adv. Intell. Mechatronics, AIM*, 2014, pp. 1547–1551.

[85] Y. Le Page and K. Holappa, "Simulation and control of an autonomous underwater vehicle equipped with a vectored thruster," in *Ocean. 2000 MTS/IEEE Conf. Exhib.*, 2000, pp. 2129–2134.

[86] B. Xin, L. Xiaohui, S. Zhaocun, and Z. Yuquan, "A vectored water jet propulsion method for autonomous underwater vehicles," *Ocean Eng.*, vol. 74, pp. 133–140, 2013.

[87] L. Shi, S. Guo, S. Mao, C. Yue, M. Li, and K. Asaka, "Development of an amphibious turtle-inspired spherical mother robot," *J. Bionic Eng.*, vol. 10, no. 4, pp. 446–455, oct 2013.

[88] E. Cavallo, R. C. Michelini, and V. F. Filaretov, "Conceptual Design of an AUV Equipped with a Three Degrees of Freedom Vectored Thruster," *J. Intell. Robot. Syst. Theory Appl.*, vol. 39, no. 4, pp. 365–391, apr 2004.

[89] H. Zhang, L. Hao, Y. Wang, Y. Liu, Z. Wu, S. Wang, S. Shao, D. Wei, and W. Hou, "The general design of a seafloor surveying AUV system," in *2013 Ocean. - San Diego*, no. January 2012. San Diego: MTS, 2013, pp. 4–8.

[90] O. Chocron, U. Prieur, and L. Pino, "A Validated Feasibility Prototype for AUV Reconfigurable Magnetic Coupling Thruster," *IEEE/ASME Trans. Mechatronics*, vol. 19, no. 2, pp. 642–650, apr 2014.

[91] L. E. J. Ackermann, K. D. Von Ellenrieder, D. Karl, and D. Beach, "Hydrodynamic testing of a Vectored-Thruster Propelled UUV," in *Ocean. 2006.* IEEE, sep 2006, pp. 1–4.

[92] O. Hegrenaes, O. Hallingstad, and B. Jalving, "Comparison of Mathematical Models for the HUGIN 4500 AUV Based on Experimental Data," in *2007 Symp. Underw. Technol. Work. Sci. Use Submar. Cables Relat. Technol.*, no. 7491. IEEE, apr 2007, pp. 558–567.

[93] J. Bellingham, C. Goudey, T. Consi, J. Bales, D. Atwood, J. J. Leonard, and C. Chryssostomidis, "A second generation survey {AUV}," *Proc. IEEE Symp. Auton. Underw. Veh. Tech.*, pp. 148–155, 1994.

[94] E. Bovio, D. Cecchi, and F. Baralli, "Autonomous underwater vehicles for scientific and naval operations," *Annu. Rev. Control*, vol. 30, no. 2, pp. 117–130, 2006.

[95] A. R. Palmer and M. Rights, "Analysis of the Propulsion and Manoeuvring Characteristics of Survey-Style AUVs and the Development of a Multi-Purpose AUV," Ph.D. dissertation, School of Engineering Sciences, Faculty of Engineering, Science & Mathematics, University of Southampton, UK, sep 2009.

[96] K. P. D'Epagnier, "AUV propellers: Optimal design and improving existing propellers for greater efficiency," in *Ocean. 2006.* IEEE, sep 2006, pp. 1–7.

[97] A. Phillips, S. Turnock, and M. Furlong, "Comparisons of CFD simulations and in-service data for the self propelled performance of an Autonomous Underwater Vehicle," pp. 5–10, 2008.

[98] R. G. Coe and W. L. Neu, "Asymmetrical wake and propulsor effects on control surface effectiveness on AUVs," *Ocean. 2012 MTS/IEEE Harnessing Power Ocean*, pp. 1–4, oct 2012.

[99] L. K. P. D'Epagnier, H.-L. Chung, M. J. Stanway, and R. W. Kimball, "An Open Source Parametric Propeller Design Tool," in *Ocean. Conf. Rec.* IEEE, 2007, pp. 1–8.

[100] A. Nemati Hayati, S. M. Hashemi, and M. Shams, "A study on the behind-hull performance of marine propellers astern autonomous underwater vehicles at diverse angles of attack," *Ocean Eng.*, vol. 59, pp. 152–163, 2013.

[101] J. L. D. Dantas and E. A. de Barros, "Numerical analysis of control surface effects on AUV manoeuvrability," *Appl. Ocean Res.*, vol. 42, pp. 168–181, 2013.

[102] M. E. Furlong, D. Paxton, P. Stevenson, M. Pebody, S. D. McPhail, and J. Perrett, "Autosub Long Range: A long range deep diving AUV for ocean monitoring," in *2012 IEEE/OES Auton. Underw. Veh. AUV 2012*. IEEE, sep 2012, pp. 1–7.

[103] Forng-Chen Chiu, Jenhwa Guo, Ji-Gang Chen, and Yen-Hwa Lin, "Dynamic characteristic of a biomimetic underwater vehicle," in *Proc. 2002 Interntional Symp. Underw. Technol. (Cat. No.02EX556)*, vol. 2002-Janua, no. 1. IEEE, 2002, pp. 172–177.

[104] J. E. Colgate and K. M. Lynch, "Mechanics and control of swimming: A review," *IEEE J. Ocean. Eng.*, vol. 29, no. 3, pp. 660–673, jul 2004.

[105] M. F. Shaari, Z. Samad, C. J. Jun, H. A. B., and A. M. Omar, "Conceptual design and preliminary analysis on bio-inspired squid micro AUV," in *2013 IEEE Int. Conf. Mechatronics Autom.* IEEE, aug 2013, pp. 1594–1598.

[106] W. Khalil, G. Gallot, and F. Boyer, "Dynamic Modeling and Simulation of a 3-D Serial Eel-Like Robot," *IEEE Trans. Syst. Man Cybern. Part C (Applications Rev.*, vol. 37, no. 6, pp. 1259–1268, nov 2007.

[107] H.-J. Kim, S.-H. Song, and S.-H. Ahn, "A turtle-like swimming robot using a smart soft composite (SSC) structure," *Smart Mater. Struct.*, vol. 22, no. 1, p. 014007, jan 2013.

[108] J. Yu, Y. Hu, R. Fan, L. Wang, and J. Huo, "Mechanical design and motion control of a biomimetic robotic dolphin," *Adv. Robot.*, vol. 21, no. 3-4, pp. 499–513, jan 2007.

[109] J. Yu, M. Tan, S. Wang, and E. Chen, "Development of a biomimetic robotic fish and its control algorithm," *IEEE Trans. Syst. Man, Cybern. Part B Cybern.*, vol. 34, no. 4, pp. 1798–1810, aug 2004.

[110] A. Willy and K. Low, "Development and initial experiment of modular undulating fin for untethered biorobotic AUVs," *2005 IEEE Int. Conf. Robot. Biomimetics - ROBIO*, pp. 45–50, 2005.

[111] A. N. A. Mazlan, "A Fully Actuated Tail Propulsion System for a Biomimetic Autonomous Underwater Vehicle," Ph.D. dissertation, University of Glasgow, 2015.

[112] M. S. Triantafyllou, A. H. Techet, and F. S. Hover, "Review of experimental work in biomimetic foils," *IEEE J. Ocean. Eng.*, vol. 29, no. 3, pp. 585–594, jul 2004.

[113] P. Krishnamurthy, Khorrami, J. De Leeuw, M. E. Porter, K. Livingston, and J. H. Long, "An electric ray inspired biomimetic autonomous underwater vehicle," *Am. Control Conf. ACC*, pp. 5224–5229, jun 2010.

[114] A. Chemori, K. Kuusmik, T. Salumäe, and M. Kruusmaa, "Depth control of the biomimetic U-CAT turtle-like AUV with experiments in real operating conditions," in *2016 IEEE Int. Conf. Robot. Autom.*, 2016, pp. 4750–4755.

[115] W. S. of biomimetic underwater robots using smart actuators Chu, K. T. Lee, S. H. Song, M. W. Han, J. Y. Lee, H. S. Kim, M. S. Kim, Y. J. Park, K. J. Cho, and S. H. Ahn, "Review of biomimetic underwater robots using smart actuators," *Int. J. Precis. Eng. Manuf.*, vol. 13, no. 7, pp. 1281–1292, jul 2012.

[116] Le Zhang, We, Yonghui Hu, Dandan Zhang, and Long Wang, "Development and depth control of biomimetic robotic fish," in *2007 IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, vol. 2, no. 3. IEEE, oct 2007, pp. 3560–3565.

[117] J. Guo and Y.-J. Joeng, "Guidance and control of a biomimetic autonomous underwater vehicle using body-fin propulsion," *Proc. Inst. Mech. Eng. Part M J. Eng. Marit. Environ.*, vol. 218, no. 2, pp. 93–111, jan 2004.

[118] C. Zhou, Z. Cao, S. Wang, and M. Tan, "The Posture Control and 3-D Locomotion Implementation of Biomimetic Robot Fish," in

2006 IEEE/RSJ Int. Conf. Intell. Robot. Syst. IEEE, oct 2006, pp. 5406–5411.

[119] S. B. Williams, P. M. Newman, G. Dissanayake, J. Rosenblatt, and H. F. Durrant-Whyte, "A decoupled, distributed AUV control architecture," in *Proc. 31st Int. Symp. Robot.*, 2006, pp. 246–251.

[120] A. Nag, S. S. Patel, K. Kishore, and S. A. Akbar, "A robust H-infinity based depth control of an autonomous underwater vehicle," in *2013 Int. Conf. Adv. Electron. Syst.* IEEE, sep 2013, pp. 68–73.

[121] T. Elmokadem, M. Zribi, and K. Youcef-Toumi, "Trajectory tracking sliding mode control of underactuated AUVs," *Nonlinear Dyn.*, vol. 84, no. 2, pp. 1079–1091, apr 2016.

[122] Xinqian Bian, Jiajia Zhou, Zheping Yan, and Heming Jia, "Adaptive neural network control system of path following for AUVs," in *2012 Proc. IEEE Southeastcon.* IEEE, mar 2012, pp. 1–5.

[123] S. A. Watson and P. N. Green, "Depth Control for Micro-Autonomous Underwater Vehicles (uAUVs): Simulation and Experimentation," *Int. J. Adv. Robot. Syst. 2014;11.*, pp. 1–10, 2014.

[124] Siciliano Bruno and O. Khatib, *Springer Handbook of Robotics.* Springer Science & Business Media, 2008.

[125] Side Zhao and J. Yuh, "Experimental study on advanced underwater robot control," *IEEE Trans. Robot.*, vol. 21, no. 4, pp. 695–703, aug 2005.

[126] B. Jalving, "The NDRE-AUV flight control system," *IEEE J. Ocean. Eng.*, vol. 19, no. 4, pp. 497–501, 1994.

[127] P. Sarhadi, A. R. Noei, and A. Khosravi, "Model reference adaptive PID control with anti-windup compensator for an autonomous underwater vehicle," *Rob. Auton. Syst.*, vol. 83, pp. 87–93, 2016.

[128] R. K. Lea, R. Allen, and S. L. Merry, "A comparative study of control techniques for an underwater flight vehicle," *Int. J. Syst. Sci.*, vol. 30, no. 9, pp. 947–964, jan 1999.

[129] J. Petrich, W. L. Neu, and D. J. Stilwell, "Identification of a simplified AUV pitch axis model for control design: Theory and experiments," in *Ocean. Conf. Rec.* IEEE, 2007, pp. 1–7.

[130] E. Dong, S. Guo, X. Lin, X. Li, and Y. Wang, "A neural network-based self-tuning PID controller of an autonomous underwater vehicle," in *Mechatronics Autom. (ICMA), 2012 Int. Conf.* IEEE, aug 2012, pp. 898–903.

[131] Qiang Chen, Tao Chen, and Yong Zhang, "Research of GA-based PID for AUV motion control," in *2009 Int. Conf. Mechatronics Autom.* IEEE, aug 2009, pp. 4446–4451.

[132] D. Yoerger and J. Newman, "Demonstration of closed-loop trajectory control of an underwater vehicle," in *Ocean. '85 - Ocean Eng. Environ.*, vol. 7. IEEE, 1985, pp. 1028–1033.

[133] G. Antonelli, S. Chiaverini, N. Sarkar, and M. West, "Adaptive control of an autonomous underwater vehicle: experimental results on ODIN," *IEEE Trans. Control Syst. Technol.*, vol. 9, no. 5, pp. 756–765, 2001.

[134] K. Do, J. Pan, and Z. Jiang, *Robust and adaptive path following for underactuated autonomous underwater vehicles*, 2004, vol. 31, no. 16.

[135] B. K. Sahu and B. Subudhi, "Adaptive tracking control of an autonomous underwater vehicle," *Int. J. Autom. Comput.*, vol. 11, no. 3, pp. 299–307, jun 2014.

[136] F. Rezazadegan, K. Shojaei, F. Sheikholeslam, and A. Chatraei, "A novel approach to 6-DOF adaptive trajectory tracking control of an AUV in the presence of parameter uncertainties," *Ocean Eng.*, vol. 107, pp. 246–258, 2015.

[137] L. T. N. D. Valladarez and N. E. Du Toit, "Robust Adaptive Control of Underwater Vehicles for Precision Operations," in *Ocean. 2015 - MTS/IEEE Washingt.*, Washington, 2015, pp. 1–7.

[138] P. Babcock and J. Zinchuk, "Fault-tolerant design optimization: application to an autonomous underwater vehicle navigation system," *Symp. Auton. Underw. Veh. Technol.*, no. 4, pp. 34–43, 1990.

[139] N. Kato, J. Kojima, Y. Kato, S. Matumoto, and K. Asakawa, "Optimization of configuration of autonomous underwater vehicle for inspection of underwater cables," in *Proceedings. 1998 IEEE Int. Conf. Robot. Autom. (Cat. No.98CH36146)*, vol. 2. IEEE, 1998, pp. 1045–1050.

[140] K. Alam, T. Ray, and S. G. Anavatti, "A new robust design optimization approach for unmanned underwater vehicle design," *Proc. Inst. Mech. Eng. Part M J. Eng. Marit. Environ.*, vol. 226, no. 3, pp. 235–249, 2012.

[141] K. Vasudev, R. Sharma, and S. Bhattacharyya, "A CAGD+CFD integrated optimization model for design of AUVs," in *Ocean. 2014 - TAIPEI*. IEEE, apr 2014, pp. 1–8.

[142] J. Sobieszczanski-Sobieski, "Multidisciplinary Design Optimization: An Emerging New Engineering Discipline," in *Adv. Struct. Optim.* Springer Netherlands, 1995, vol. 25, no. May 1993, pp. 483–496.

[143] C. McAllister, T. Simpson, P. Kurtz, and M. Yukish, "Multidisciplinary Design Optimization Testbed Based on Autonomous Underwater Vehicle Design," in *9th AIAA/ISSMO Symp. Multidiscip. Anal. Optim.* Reston, Virigina: American Institute of Aeronautics and Astronautics, sep 2002.

[144] M. Martz and W. Neu, "Multi-objective optimization of an autonomous underwater vehicle," *Ocean. 2008*, pp. 1–9, 2008.

[145] G. Xia, C. Liu, and X. Chen, "Multi-objective optimization for AUV conceptual design based on NSGA-II," in *Ocean. 2016 - Shanghai*. IEEE, apr 2016, pp. 1–6.

[146] T. Fossen, *Guidance and Control of Ocean Vehicles. 1994*.

[147] T. Koh, M. Lau, E. Low, G. Seet, S. Swei, and P. Cheng, "A study of the control of an underactuated underwater robotic vehicle," in *IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, vol. 2. IEEE, 2002, pp. 2049–2054.

[148] T. H. Koh, M. W. S. Lau, G. Seet, and E. Low, "A Control Module Scheme for an Underactuated Underwater Robotic Vehicle," *J. Intell. Robot. Syst.*, vol. 46, no. 1, pp. 43–58, jul 2006.

[149] "ATI Force / Torique Sensor: NANO17 IP65/IP6," 2014.

[150] E. P. Vega, O. Chocron, and M. Benbouzid, "A flat design and a validated model for an auv reconfigurable magnetic coupling thruster," *IEEE/ASME Transactions on Mechatronics*, vol. 21, no. 6, pp. 2892–2901, Dec 2016.

[151] G. Akoun and J.-P. Yonnet, "3D analytical calculation of the forces exerted between two cuboidal magnets," *IEEE Trans. Magn.*, vol. 20, no. 5, pp. 1962–1964, sep 1984.

[152] O. Chocron and H. Mangel, "Reconfigurable magnetic-coupling thrusters for agile AUVs," in *2008 IEEE/RSJ Int. Conf. Intell. Robot. Syst.* Nice, France: IEEE, sep 2008, pp. 3172–3177.

[153] J. Coulomb, Y. Du Terrail, and G. Meunier, "Flux3D, a finite element package for magnetic computation," *IEEE Trans. Magn.*, vol. 21, no. 6, pp. 2499–2502, nov 1985.

[154] B. Ferreira, M. Pinto, A. Matos, and N. Cruz, "Hydrodynamic modeling and motion limits of AUV MARES," *2009 35th Annu. Conf. IEEE Ind. Electron.*, pp. 2241–2246, nov 2009.

[155] T. Prestero and . Prestero, Timothy (Timothy Jason), "Verification of a six-degree of freedom simulation model for the REMUS autonomous underwater vehicle," Ph.D. dissertation, Massachusetts Institute of Technology, 2001.

[156] T. Prestero, "Development of a six-degree of freedom simulation model for the REMUS autonomous underwater vehicle," in *MTS/IEEE Ocean. 2001. An Ocean Odyssey. Conf. Proc. (IEEE Cat. No.01CH37295)*, vol. 1. Marine Technol. Soc, 2001, pp. 450–455.

[157] E. GUYON, J.-P. HULIN, and L. PETIT, *Hydrodynamique Physique*. InterEditions, 2001.

[158] M. Issac, S. Adams, M. He, and N. Bose, "Manoeuvring trials with the MUN explorer AUV: data analysis and observations," *Ocean. 2007*, pp. 1–8, sep 2007.

[159] M. T. M. Issac, S. Adams, M. He, N. Bose, C. D. C. Williams, R. Bachmayer, T. Crees, and Moqin He, "Manoeuvring Experiments Using the MUN Explorer AUV," in *2007 Symp. Underw. Technol. Work. Sci. Use Submar. Cables Relat. Technol.* IEEE, apr 2007, pp. 256–262.

[160] P. Millán, L. Orihuela, I. Jurado, and F. R. Rubio, "Formation Control of Autonomous Underwater Vehicles Subject to Communication Delays," *IEEE Trans. Control Syst. Technol.*, vol. 22, no. 2, pp. 770–777, 2014.

[161] H. Yang, C. Wang, and F. Zhang, "Robust geometric formation control of multiple autonomous underwater vehicles with time delays," *Am. Control Conf. (ACC), 2013*, pp. 1380–1385, jun 2013.

[162] F. Alonge, F. D'Ippolito, and F. Raimondi, "Trajectory tracking of underactuated underwater vehicles," in *Proc. 40th IEEE Conf. Decis. Control (Cat. No.01CH37228)*, vol. 5. IEEE, 2001, pp. 4421–4426.

[163] T. I. Fossen and T. A. Johansen, "A Survey of Control Allocation Methods for Ships and Underwater Vehicles," in *2006 14th Mediterr. Conf. Control Autom.* IEEE, jun 2006, pp. 1–6.

[164] E. P. Vega, O. Chocron, and M. Benbouzid, "AUV Propulsion Systems Modeling Analysis," *Int. Rev. Model. Simulations*, vol. 7, no. 5, pp. 827–837, 2014.

[165] M. Nakhaeinejad and M. D. Bryant, "Observability analysis for model-based fault detection and sensor selection in induction motors," *Meas. Sci. Technol.*, vol. 22, no. 7, p. 075202, jul 2011.

[166] Y. Cao, "Complex step Jacobian," *MATLAB Cent. File Exch. Available http//mathworks.com/matlabcentral/fileexchange/18176*, 2008.

[167] M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, and A. Y. Ng, "ROS: An Open-Source Robot Operating System," in *Proc. Open-Source Softw. Work. Int. Conf. Robot. Autom.*, 2009.

[168] M. Prats, J. Perez, J. J. Fernandez, and P. J. Sanz, "An open source tool for simulation and supervision of underwater intervention missions," in *IEEE Int. Conf. Intell. Robot. Syst.* IEEE, oct 2012, pp. 2577–2582.

[169] T. Bäek, *Evolutionary algorithms in theory and practice : evolution strategies, evolutionary programming, genetic algorithms*. Oxford University Press, 1996.

[170] A. R. Yildiz, "Comparison of evolutionary-based optimization algorithms for structural design optimization," *Eng. Appl. Artif. Intell.*, vol. 26, no. 1, pp. 327–333, 2013.

[171] O. Chocron and P. Bidaud, "Evolutionary algorithms in kinematic design of robotic systems," *Proc. 1997 IEEE/RSJ Int. Conf. Intell. Robot Syst. Innov. Robot. Real-World Appl. IROS '97*, vol. 2, no. February, pp. 1111–1117, 1997.

[172] A. Bataller, J. Cabrera, M. Clavijo, and J. Castillo, "Evolutionary synthesis of mechanisms applied to the design of an exoskeleton for finger rehabilitation," *Mech. Mach. Theory*, vol. 105, pp. 31–43, 2016.

[173] O. Chocron and P. Bidaud, "Conception Evolutionnaire de Systemes Robotiques," Ph.D. dissertation, Université de Paris 06, 2000.

[174] C. Darwin, "On the origin of species by means of natural selection, or the preservation of favoured races in the struggle for life," *Murray, London*, 1859.

[175] D. E. D. E. Goldberg, *Genetic algorithms in search, optimization, and machine learning.* Addison-Wesley Longman Publishing Co., Inc., 1989.

[176] H. J. Lichtfuss, "Evolution eines rohrkrümmers," Ph.D. dissertation, Technische Universität der Berlin, 1965.

[177] I. Rechenberg, "Cybernetic solution path of an experimental problem," in *R. Aircr. Establ. Transl. No. 1122*, Farnborough, Hants., UK, 1965.

[178] D. B. Fogel, *System Identification trough Simulated Evolution: A Machine Learning Approach.* Ginn Press, 1991.

[179] M. Srinivas and L. M. Patnaik, "Genetic Algorithms: A Survey," *Computer (Long. Beach. Calif).*, vol. 27, no. 6, pp. 17–26, 1994.

[180] J. H. Holland, *Adaptation in Natural and Artificial Systems: An introductory analysis with applications to biology, control, and artificial intelligence.* University of Michigan Press, 1975.

[181] K. A. De Jong, "An analysis of the behaviour of a class of genetic algorithms," Ph.D. dissertation, University of Michigan, 1975.

[182] J. J. Grefenstette, "Optimization of Control Parameters for Genetic Algorithms," *IEEE Trans. Syst. Man Cybern.*, vol. 16, no. 1, pp. 122–128, jan 1986.

[183] M. Molga and C. Smutnicki, "Test functions for optimization needs," 2005.

[184] S. Koos, J. B. Mouret, and S. Doncieux, "The transferability approach: Crossing the reality gap in evolutionary robotics," *IEEE Trans. Evol. Comput.*, vol. 17, no. 1, pp. 122–145, feb 2013.

[185] N. Jakobi, "Minimal simulations for evolutionary robotics," Ph.D. dissertation, University of Sussex, 1998.

[186] A. Thompson, P. Layzell, and R. S. Zebulum, "Explorations in design space: Unconventional electronics design through artificial evolution," *IEEE Trans. Evol. Comput.*, vol. 3, no. 3, pp. 167–195, 1999.

[187] D. Floreano and J. Urzelai, "Evolution of plastic control networks," *Auton. Robots*, vol. 11, no. 3, pp. 311–317, 2001.

[188] G. Hornby, S. Takamura, J. Yokono, O. Hanagata, T. Yamamoto, and M. Fujita, "Evolving robust gaits with AIBO," *Proc. 2000 ICRA. Millenn. Conf. IEEE Int. Conf. Robot. Autom. Symp. Proc. (Cat. No.00CH37065)*, vol. 3, no. April, pp. 3040–3045, 2000.

[189] V. Zykov, J. Bongard, and H. Lipson, "Evolving dynamic gaits on a physical robot," *Proc. Genet. ...*, 2004.

[190] J. B. Pollack, H. Lipson, S. Ficici, P. Funes, G. Hornby, and R. a. Watson, "Evolutionary Techniques in Physical Robotics," in *Int. Conf. Evolvable Syst.*, vol. 1801, no. Moravec 1999. Springer Berlin Heidelberg, 2000, pp. 175–186.

[191] S. Nolfi and D. Floreano, "How to evolve autonomous robots: Different approaches in evolutionary robotics," *Artif. Artif. life IV Proc. 4th Int. Work. Artif. Lifel life IV*, vol. 1997, pp. 190–197, 1994.

[192] J. Bongard and H. Lipson, "Once more unto the breach: Co-evolving a robot and its simulator," *... Conf. Simul. ...*, pp. 57–62, 2004.

[193] J. Bongard, V. Zykov, and H. Lipson, "Resilient Machines Through Continuous Self-Modeling," *Science (80-.).*, vol. 314, no. 5802, pp. 1118–1121, 2006.

[194] J. C. Zagal and J. Ruiz-Del-Solar, "Combining simulation and reality in evolutionary robotics," *J. Intell. Robot. Syst. Theory Appl.*, vol. 50, no. 1, pp. 19–39, 2007.