DM

Neural Network-Based Model for Bioinspired Underwater Propulsion Systems

MASTER THESIS

Fábio Emanuel Fiqueli Abreu
MASTER IN INFORMATICS ENGINEERING

UNIVERSIDADE da MADEIRA
A Nossa Universidade
www.uma.pt

September | 2025

Neural Network-Based Model for Bioinspired Underwater Propulsion Systems

MASTER THESIS

Fábio Emanuel Fiqueli Abreu
MASTER IN INFORMATICS ENGINEERING

SUPERVISOR
Luiz Carlos Guerreiro Lopes

CO-SUPERVISOR
Emanuel António Rodrigues Camacho

Neural Network-Based Model for Bioinspired Underwater Propulsion Systems

Fábio Emanuel Fiqueli Abreu
B.Sc. in Informatics Engineering

A thesis presented to the University of Madeira
in fulfillment of the requirements for the degree
of Master of Science in Informatics Engineering

Supervisor:
Prof. Dr. Luiz Carlos Guerreiro Lopes

Co-Supervisor:
Dr. Emanuel António Rodrigues Camacho

Evaluation Committee:
Prof. Dr. Karolina Baras, FCEE/UMa (Committee Chair)
Prof. Dr. Fábio Rúben Silva Mendonça, FCEE/UMa
Prof. Dr. Luiz Carlos Guerreiro Lopes, FCEE/UMa

Funchal, Portugal, September 2025

"The important thing is not to stop questioning. Curiosity has its own reason for existing. One cannot help but be in awe when he contemplates the mysteries of eternity, of life, of the marvelous structure of reality. It is enough if one tries merely to comprehend a little of this mystery every day."

— Albert Einstein [1]

Acknowledgements

I would like to express my special thanks to Prof. Dr. Luiz Guerreiro Lopes and Dr. Emanuel Camacho, not only for suggesting the project idea, but also for their guidance and valuable collaboration throughout its development.

I would also like to thank IntellMax – Optimization, Artificial Intelligence and Data Science, Lda., Portugal, for their financial support, which made it possible to acquire some of the equipment necessary for the project.

Lastly, I would like to extend my gratitude to my family for their constant support.

Contents

Abstract

Researchers have been studying animal propulsion mechanisms to develop agile, efficient, and versatile biomimetic robotic systems for marine science, industry, defense, and recreation applications. This project concerns the development of an experimental methodology to analyze the propulsive performance of a system inspired by fish caudal fin kinematics. The experimental rig features a 3D-printed fin that oscillates with the help of a servo motor, whose frequency and amplitude vary over time, creating multiple test conditions. In the rig, a load cell is used to measure the thrust force generated by the oscillation, and an electric current sensor is used to measure the current consumed by the servo when making the movements. The data extracted from these experiments are used to train a Kolmogorov–Arnold Network (KAN), which learns to predict the thrust and power outputs of the system based on input parameters such as fin amplitude, frequency, and angular velocity. The results obtained offer experimental evidence supporting the viability of using neural networks to model thrust and power in bio-inspired systems, while providing valuable findings that may contribute to the design and improvement of bio-inspired underwater propulsion systems.

Keywords: Neural networks, KANs, Biomimetic oscillatory propulsion, Pitching hydrofoils, Thrust efficiency optimization

Resumo

Mecanismos de propulsão animal têm sido estudados por investigadores com o objetivo de desenvolver sistemas robóticos biomiméticos ágeis, eficientes e versáteis destinados a aplicações nas áreas das ciências marinhas, indústria, defesa e recreação. Este projeto visa desenvolver uma metodologia experimental para analisar o desempenho propulsivo de um sistema inspirado na cinemática da barbatana caudal de peixes. O banco de ensaio experimental possui uma barbatana impressa em 3D que oscila com a ajuda de um servomotor, que varia em frequência e amplitude ao longo do tempo, criando múltiplas condições de ensaio. No banco de ensaio, uma célula de carga é utilizada para medir a força propulsora gerada pela oscilação, e um sensor de corrente elétrica é utilizado para medir a corrente consumida pelo servo durante os movimentos. Os dados extraídos destas experiências são utilizados para treinar uma rede de Kolmogorov–Arnold (KAN), que aprende a prever a força propulsora e a potência do sistema com base em parâmetros de entrada como a amplitude, a frequência e a velocidade angular da barbatana. Os resultados obtidos fornecem evidências experimentais que suportam a viabilidade da utilização de redes neuronais para modelar a propulsão e a potência em sistemas bioinspirados, apresentando resultados úteis que poderão contribuir para o projeto e a melhoria de sistemas de propulsão subaquática bioinspirados.

Palavras-chave: Redes neuronais, KAN, Propulsão oscilatória biomimética, Hidrofólios oscilantes, Otimização da eficiência de impulsão

List of Figures

List of Tables

Nomenclature

f	Frequency	Hz
t	Time	s
α	Angle of attack	rad
A	Plunging amplitude	m
U_∞	Flow velocity	$m \cdot s^{-1}$
St	Strouhal number	–
ρ	Fluid density	$Kg \cdot m^{-3}$
c	Aerodynamic chord	m
Re	Reynolds number	–
k	Reduced frequency	–
AR	Aspect ratio	–
h	Non-dimensional plunge amplitude	–
T	Thrust force	N
S	Reference area	m^2
C_t	Thrust coefficient	–
C_p	Power coefficient	$m \cdot s^{-1}$
P	Power required	W
η	Propulsive efficiency	–
E	Energy spent	J
d	Distance traveled	m
CTO	Cost of Transport	$J \cdot (Kg \cdot m)^{-1}$
D	Drag	–

List of Acronyms

AI	Artificial Intelligence
AUVs	Autonomous Underwater Vehicles
BCF	Body-caudal fin
CFD	Computacional Fluid Dinamics
DPVs	Diver Propulsion Vehicles
KANs	Kolmogorov–Arnold Networks
MPF	Median-Paired fin
USVs	Uncrewed Surface Vessels

Chapter 1

Introduction

This thesis presents a project carried out at IntellMax, a technology startup specializing in Optimization, Artificial Intelligence and Data Science, as part of the requirements for obtaining the Master's degree in Informatics Engineering at the University of Madeira.

In this chapter, the motivation and objectives of the project are presented, followed by an overview of the thesis structure.

1.1 Motivation

The underlying motivation of this study is related to the fact that the present period is one of energy transition, in which one of the European goals is to reduce greenhouse gases and decarbonize maritime transport. Given the increasing demand for sustainable alternatives, an opportunity arises to highlight the area of bio-inspired systems as one of the viable solutions.

Thanks to advances in computational resources and simulation tools, this has been a field of great interest for the study of more efficient propulsion systems. Recent studies and prototypes even demonstrate that flapping hydrodynamics may be more efficient than conventional systems and can reduce fuel emissions by up to 50% [2].

1.2 Objectives

This project aims to contribute to this transition by developing a system that investigates the propulsion generated by a flapping motion. The system is designed to be replicable and suitable for application in bio-inspired Autonomous Underwater Vehicles (AUVs), Diver Propulsion Vehicles (DPVs), and Uncrewed Surface Vehicles (USVs).

The project had three main objectives:

- To create a system that mimics the motion of a flapping foil;
- To extract data from the foil's flapping behavior;
- To use this data to train a neural network capable of analyzing the system's performance and potentially improving propulsion efficiency and energy management.

With this experimental methodology, the aim is to create a system that enables testing bio-inspired hydrofoils in a realistic, fully submerged aquatic environment before their integration into a final prototype.

As mentioned, this system mimics the motion of a flapping foil while simultaneously collects data through a set of sensors and actuators. These data are subsequently used to train a neural network capable of predicting the thrust and energy consumption for parameter combinations not previously tested, which represents a significant advantage.

1.3 Thesis Structure

This thesis is organized into five chapters.

The first chapter introduces the research topic, along with the motivation and objectives of the study, as well as the overall structure of the thesis.

In the second chapter, a literature review is presented. This chapter describes the evolution of the scientific field and highlights some of the most relevant prototypes developed in recent years, based on various sources and authors.

The third chapter details the methodology, including the materials used, the implementation of a system to simulate caudal fin movement, the

acquisition of sensor data, and the introduction of the Kolmogorov–Arnold neural network, a relatively recent model employed in this thesis to learn from the different combinations of amplitudes and frequencies tested, which simulate the behavior of a flapping motion.

In the fourth chapter, the data collected by the system are pre-processed so that they can be used to train neural networks. In addition to the pre-processing procedures, this chapter presents the configuration used to train the KAN model and the results obtained from this training. Finally, two other, more common network models are also tested: Gradient Boosting and a Long Short-Term Memory (LSTM) network. Both models are trained, and their results are presented and compared with those of the KAN model.

In the fifth and final chapter, the conclusions are presented, summarizing the main aspects addressed, the results achieved, and the system's limitations, followed by suggestions for future work.

Chapter 2

Literature Review

The literature review is organized into three sections. The first seeks to answer the question: 'How many aquatic propulsion systems exist in nature?' The aim is to gain a deeper understanding of fish morphology and the different natural propulsion mechanisms.

The second section presents a survey of the main discoveries that marked the field of flapping hydrodynamics, from the most relevant scientific articles to some of the fundamental equations and parameters currently used to evaluate the performance of these systems. This section also includes an analysis of some prototypes developed to date from an evolutionary perspective. This analysis serves as a bridge to the third section.

The third section briefly addresses the evolution of artificial intelligence, exploring its advantages and disadvantages. Based on this analysis, practical examples are provided to illustrate how the combination of nature-inspired propulsion and artificial intelligence can support the development of more efficient aquatic locomotion systems. Such systems are capable of intelligently managing energy consumption, thereby extending autonomy and enhancing navigation in open seas.

2.1 Types of Biomimetic Propulsion Systems

This section provides a concise overview of the different types of biomimetic propulsion systems found in nature.

By looking at the different types of organisms, the vast majority of

biologists and scientists have classified the propulsion systems based on how animals swim, based on how they manage to move and they identify four to five types of systems [3, 4]:

- **Oscillatory:** Propulsion generated by the periodic oscillation of relatively rigid appendages (e.g., pectoral fins or the caudal fin in thunniform swimmers) to generate thrust for locomotion.
- **Undulatory:** Propulsion generated by a traveling wave that passes along a flexible body or fin (e.g., anguilliform swimming), displacing water backward.
- **Pulsatile:** It consists of discharging a quantity of water in the form of a jet, producing thrust in the opposite direction to the flow of the jet.
- **Drag-based:** When an animal uses its limbs to push water backward, generating thrust by reaction, as observed in frogs.
- **Lift-based:** Propulsion is generated by lift force, similar to how an airplane's wings work. This type of swimming can be seen in turtles and penguins.

Within each of these general categories, it is also common to classify propulsion systems based on animal morphology.

In the case of fish, they are usually classified not only based on whether they move in an oscillatory or undulating manner, or through a combination of both, but also by the type of fins they have. It is possible for a fish to have more than one type of fin, which is, in fact, the most common situation in most species.

Figure 2.1 shows the morphology of a fish, highlighting the different types of fins, while Table 2.1 presents the various swimming modes.

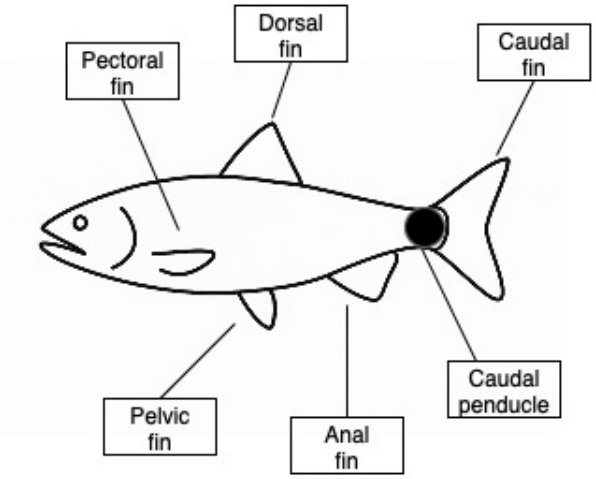

Figure 2.1: Fish morphology

Table 2.1: Different swimming modes

Category	**Fin Type**	**Oscillatory**		**Undulatory**
Body-Caudal		Ostraciform Thunniform	Carangiform Subcarangiform	Anguilliform
Median-Paired	Pectoral Anal Dorsal	Labriform Tetraodontiform	Diodontiform Gymnotiform Balistiform	Rajiform Amiiform

These classifications are not only essential for distinguishing and categorizing the different types of propulsion that exist but also for understanding the vast scope of bioinspired systems.

In this thesis, the body-caudal fin (BCF) propulsion system is studied. Depending on the species and swimming mode, BCF propulsion can be oscillatory or undulatory. In tuna, for example, it is predominantly oscillatory, as thrust is generated mainly by the tail.

This swimming mode can be further classified as thunniform or ostraciiform, depending on the species and fin shape.

2.2 Flapping Hydrodynamics

Flapping is a term used in aerodynamics to describe a combined plunging and pitching motion.

Initially, wing studies focused predominantly on isolated plunging motion. However, over time, the focus of this area of study has shifted to a combined solution between plunging and pitching, since pitching combined with plunging is more effective in generating lift and propulsion, producing more significant aerodynamic forces.

This combined solution is in line with what is observed in nature, being used by both flying animals and efficient swimmers, who use combined movements to maximize the generation of propulsive force.

Some of the most notable articles, which inspired a variety of subsequent studies, were published in 1909 and 1912 by Knoller [5] and Betz [6]. These scientists demonstrated that the vertical oscillation of an airfoil (plunging) changes its effective angle of attack, simultaneously generating lift and thrust forces.

In 1922, Katzmayr [7] validated this theory by demonstrating that air oscillation around a wing generates thrust.

In 1924, Birnbaum [8] discovered that, in addition to thrust, this motion behavior could also generate unwanted vibrations, known as flutter.

A decade later, in 1935, Karman and Burgers [9] supported the Knoller-Betz effect by providing a theoretical explanation that connected the position and orientation of wake vortices to the forces acting on the wing, such as drag and thrust.

In that same year Theodorsen [10] wrote an article that would become very well known for describing the aerodynamic force on a wing and introduced a complex function to predict flutter events.

The area that studies bioinspired aquatic propulsion systems emerged in 1936 when James Gray investigated the power required for a dolphin to swim at high speeds. This work became known as Gray's Paradox and was later

revisited in an article by Rahul et al. [11], due to the discrepancy between the estimated force generated by the dolphin's muscle and the estimated drag force.

At the time, the data was very limited, and there were only a few studies related to hydrodynamics, animal muscle physiology, and swimming performance. Because of that, some of the results were later demystified.

Gray's study suggested that dolphins needed to generate seven times more muscle power than other mammals to overcome drag, based on a rigid-body hydrodynamic model.

With subsequent scientific advances, these results were rejected. Later, researchers discovered that muscles could be more efficient than initially estimated, and that important factors such as body flexibility and undulating motion had been disregarded in a rigid body study, and that these factors would have a significant impact on drag reduction.

Shortly thereafter, in 1937, Garrick [12] used Theodorsen's theory of vortices to calculate the horizontal force generated by an airfoil.

In 1939 Silverstein and Joyner [13] demonstrated the relationship between the frequency of motion and the air flow speed and identified the optimal conditions for maximizing propulsion and minimizing drag.

In 1950, Bratt [14] conducted a wind tunnel experiment to study flow patterns behind a moving airfoil and observed that higher motion frequencies increased the formation of distinct vortex structures in the wake.

In 1996, Streitlien et al. [15] published a study about increasing the propulsive efficiency of an airfoil through vortex control. The paper demonstrated that maximum efficiency, which was around 87 %, occurred when leading edge vortices interacted optimally with trailing edge vortices, forming a reverse Kármán vortex street, as shown in Figure 2.2, extracted from Prats' thesis [16].

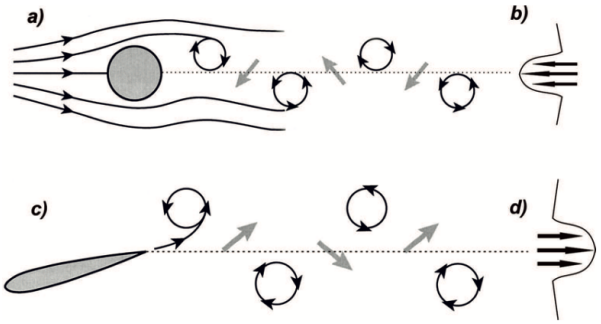

Figure 2.2: Standard vs. reverse Kármán vortex street. a) Kármán vortex street; b) velocity profile; c) reverse Kármán vortex street; d) velocity profile

In 1998, Jones [17] explored the Knoller–Betz effect in flapping airfoils, focusing on how vortices influence the balance between drag and thrust. They found that as plunging velocity increased, wake vortices transitioned from drag-inducing configurations to those producing net thrust, due to organized vortex patterns generating propulsive force.

At Strouhal numbers $St \leq 0.3$, viscous effects dominate, reducing airflow circulation around the airfoil. At higher Strouhal numbers ($St \geq 1$), the wake patterns become less symmetric due to increased flow separation, which negatively impacts thrust generation.

In 1999, Tolkoff [18] published a thesis on the study of the forces measured in what was the world's first tuna robot, investigating the power required for propulsion and the role of body flexibility in hydrodynamic efficiency.

Tuncer and Platzer [19], in 2000, investigated the behavior of the viscous and unsteady flow fields around a NACA 0012 airfoil, through the variation of plunging and pitching motions. The results of this study show that maximum propulsive efficiency occurred in scenarios when airflow remained predominantly attached to the airfoil, maintaining lift and reducing drag.

In 2003, Taylor et al. [20] conducted a study on various aquatic and aerial species and found that the most efficient ones operated at Strouhal numbers between 0.2 and 0.4.

This range corresponds to the ideal formation of a reverse Kármán vortex street in the wake, which is associated with efficient thrust generation.

Guglielmini and Blondeaux [21], in 2004, after performing a numerical simulation of foils, concluded that oscillation only in heave would not be

sufficient to generate thrust and that pitching is necessary to obtain optimal values, especially in hovering mode.

In 2005, Tuncer and Kaya [22] investigated maximizing thrust and propulsive efficiency of an oscillating airfoil's combined plunging and pitching motions using a numerical steepest-ascent method. They varied plunge and pitch amplitudes and phase differences, finding optimal efficiency with plunge amplitudes of 0.4–0.8, pitching amplitudes of 15°–35°, and phase differences of 80°–90°. Maximum thrust was achieved with plunge amplitudes of 1.6–2.1 and pitching amplitudes of 23°–36°. The study linked high thrust to leading-edge vortex formation, though this compromised efficiency.

In 2012, Esposito [23] conducted a study in which he examined how fin stiffness and servo motor control could influence locomotor performance.

In 2019, Emanuel Camacho [24] conducted a numerical study in his master's thesis, analyzing the aerodynamic performance of a NACA0012 airfoil in vertical motion (plunging) at low Reynolds numbers.

Among the results obtained, the study demonstrated that even without varying the angle of attack, it is possible to generate considerable thrust. However, this thrust depends on several factors, including frequency, motion amplitude, and Reynolds number (Re).

Additionally, it was found that increasing the amplitude and frequency led to a higher thrust coefficient, a trend consistent with theoretical expectations.

The thesis revealed that propulsive efficiency varies with Reynolds number. For $Re = 8500$ and $Re = 17000$, the highest propulsive efficiency is achieved with high reduced frequencies and low amplitudes, whereas for $Re = 34000$, optimal efficiency occurs with low reduced frequencies and high amplitudes.

Increasing the Reynolds number was shown to improve both the thrust coefficient and propulsive efficiency, particularly at a Strouhal number of $St = 0.1$, where high efficiency is achieved at low reduced frequencies (≈ 1). For $St > 0.1$, the maximum efficiency shifts to a reduced frequency of approximately ($k \approx 3$).

Although the Strouhal number alone is insufficient to fully characterize the flow due to multiple possible flow configurations, it exhibits a strong correlation with aerodynamic performance, including thrust production,

required power, and propulsive efficiency, especially at higher Reynolds numbers.

The influence of the angle of attack was also examined. Increasing the angle of attack reduces the thrust coefficient (C_T) and propulsive efficiency (η), while increasing the lift coefficient, as expected. Notably, the power coefficient remained largely unaffected by angle of attack variations, an effect not widely explored in the literature.

The study employed Reynolds-Averaged Navier-Stokes (RANS) equations with the Realizable ($k-\varepsilon$) and SST ($k-\omega$) turbulence models. The ($k-\omega$) model proved more suitable for simulating plunging airfoils, as the ($k-\varepsilon$) model struggled to predict thrust and efficiency at Strouhal numbers below 0.15. Numerical validation was achieved through boundary location, mesh, and time step independence studies, with results aligning well with existing literature.

Turbulent effects at low frequencies were identified as contributing to reduced efficiency, consistent with the observed challenges in predicting performance at low Strouhal numbers.

2.2.1 Governing Parameters

In this study, a pitching motion is applied to the airfoil, which is essentially a sinusoidal motion, as described by the equation below:

$$\alpha(t) = \alpha_0 + \alpha_{amp} \cos(2\pi f t). \tag{2.1}$$

The Strouhal number (St) is a dimensionless parameter commonly used to characterize vortex shedding. It relates the oscillation frequency and amplitude of motion to the free-stream velocity, and is defined as:

$$St = \frac{2fA}{U_\infty}. \tag{2.2}$$

The Reynolds number (Re) characterizes the flow regime:

$$Re = \frac{\rho U_\infty c}{\mu}. \tag{2.3}$$

The reduced frequency (k) quantifies the level of unsteadiness in the flow:

$$k = \frac{2\pi f c}{U_\infty}. \tag{2.4}$$

The aspect ratio (AR) is a geometric parameter that characterizes how elongated a wing or fin is. In aerodynamics and hydrodynamics, the standard definition is:

$$AR = \frac{b^2}{S}, \tag{2.5}$$

where b is the span and S is the planform area.

In some simplified cases, such as two-dimensional foils or fish fins, AR is sometimes approximated by the ratio of span to chord ($AR = b/c$).

The motion is characterized by the non-dimensional amplitude parameter h, defined as:

$$h = \frac{H}{c}, \tag{2.6}$$

where H is the maximum oscillation amplitude and c is the chord length.

2.2.2 Performance Parameters

The thrust coefficient (C_t) quantifies the generated thrust:

$$C_t = \frac{T}{\frac{1}{2}\rho U^2 S}. \tag{2.7}$$

The power coefficient (C_p) represents the power consumption:

$$C_p = \frac{P}{\frac{1}{2}\rho U^3 S}. \tag{2.8}$$

Propulsive efficiency (η) measures the effectiveness of power conversion into thrust:

$$\eta = \frac{\overline{T}U}{\overline{P}}. \tag{2.9}$$

The governing parameters directly affect the performance parameters, a higher frequency means that there is also a higher C_t, and consequently there may also be an increase in (C_p) which both affects the value of η.

In this work, a study of a fin attached to a stationary structure is

presented.

It is also important to note that, in addition to the performance parameters, locomotion systems are often evaluated using the cost of transport (COT), defined as:

$$COT = \frac{E}{m \cdot d}, \tag{2.10}$$

where E is the energy consumed, m is the body mass, and d is the distance traveled.

2.2.3 Existing Prototypes

As described in Section 2.2, the first bioinspired robot only appeared in 1994, but well before that there was already another technology with foils being investigated and which are still used today.

The technology used is based on the principle of converting wave energy into propulsion through the use of foils. Foils are usually attached to shafts or moving structures that change angle based on the lift and plunging movement.

The first publication of this type of technology dates back to 1895, with Herman Linden [25]. Today, this technology is still in use and can be found in USVs developed by companies such as AutoNaut [26] and Liquid Robotics [27].

- **RoboTuna**

  The morphology and swimming behavior of fish result from an evolutionary process spanning thousands of years of natural selection [28]. This efficiency has motivated substantial interest in studying fish locomotion and developing robotic prototypes.

  The first bioinspired robotic fish, known as RoboTuna, was created in 1994 at MIT by a team of researchers led by Prof. Michael S. Triantafyllou, with David S. Barrett as a key contributor [29]. As shown in Figure 2.3, RoboTuna was designed to serve as a platform for studying fish-like propulsion mechanisms and for evaluating their potential advantages over conventional systems.

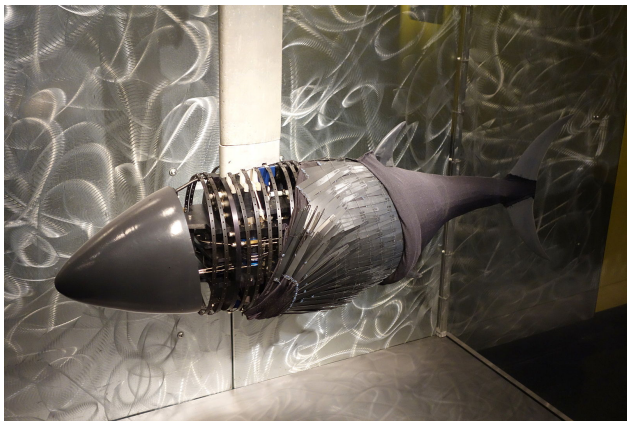

Figure 2.3: RoboTuna

At the time, discussions were already underway regarding the construction of untethered prototypes, such as the Robot Pike later developed in 2000 [30]. Research efforts also explored the use of genetic algorithms to optimize control parameters and improve efficiency. As Barrett noted, "So the overall efficiency gets exponentially better and better, and we end up with an arrangement of control parameters that make for the most efficient run" [31].

- **Proteus, the 'Penguin Boat'**

  In 1997, the prototype Proteus, also known as the "Penguin Boat", was developed at MIT in collaboration with the U.S. Navy. This 12-foot vessel was propelled by two fins attached to the stern, mimicking the swimming of penguins. As shown in Figure 2.4, Proteus was the first project to explore the use of multiple flapping fins as a propulsion strategy.

  According to early laboratory tests [32], the vessel demonstrated promising performance, reaching an efficiency of approximately 87 %, compared to about 70 % for conventional ships.

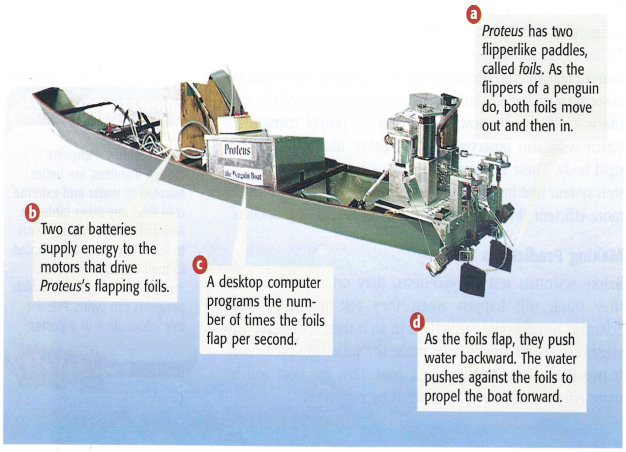

Figure 2.4: Proteus

- **O-Foil**

  In 2008, the Dutch company MARIN developed an experimental project called O-Foil, which was a vessel resembling a full-size tanker equipped with a propulsion system inspired by dolphin locomotion, as illustrated in Figure 2.5. The project indicated that the implemented system could reduce fuel consumption and emissions by up to 50 % [2].

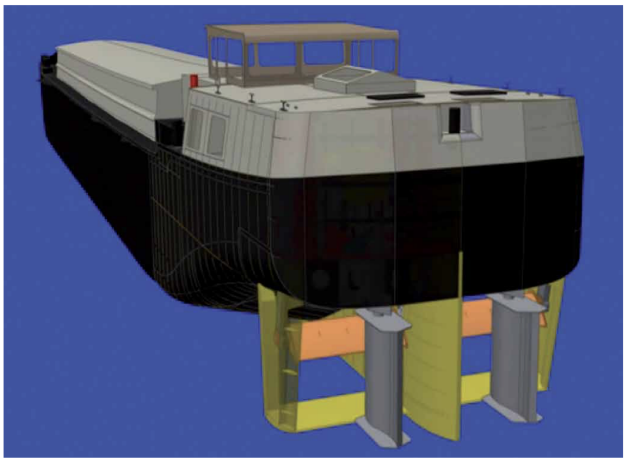

Figure 2.5: O-Foil experimental vessel

- **Suntory Mermaid II**

  Also in 2008, the catamaran Suntory Mermaid II was developed, featuring a biomimetic wave-powered propulsion system. The vessel

was equipped with two fin-like structures at the bow that converted the vertical motion of waves into forward thrust, emulating the locomotion of marine organisms.

This boat became famous for completing a voyage from Hawaii to Japan in 2008 and received an award for the world's longest solo trip on a wave-powered vessel.

- **SoFi, the Soft Robotic Fish**

  In 2018, MIT developed the SoFi prototype [33], illustrated in Figure 2.6, which was described as the most advanced robotic fish of its kind ever built. This robot could be controlled using an underwater remote to communicate with a human operator [34].

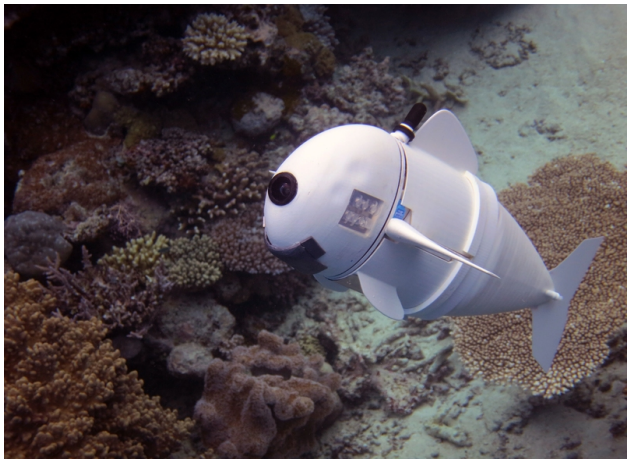

Figure 2.6: SoFi robotic fish

The main objective of its development was to study marine life without disturbing it, as it was equipped with cameras and environmental sensors. It enabled a new form of observation, with communication capability at distances ranging from 0 to 18 meters and an operating autonomy of about 40 minutes.

Looking ahead, the researchers indicated their intention to develop a prototype capable of remaining underwater for longer periods and operating without direct human intervention. They also highlighted that such a robot would be important in assessing whether robotic presence can influence the behavior of marine life.

- **Bluebots, the Robotic Swarm**

  In 2021, Harvard University presented a project involving multiple robots called Bluebots [35, 36], organized in a swarm system, as shown in Figure 2.7. These robots demonstrated the ability to disperse, group into schools, and perform tasks together in circular patterns.

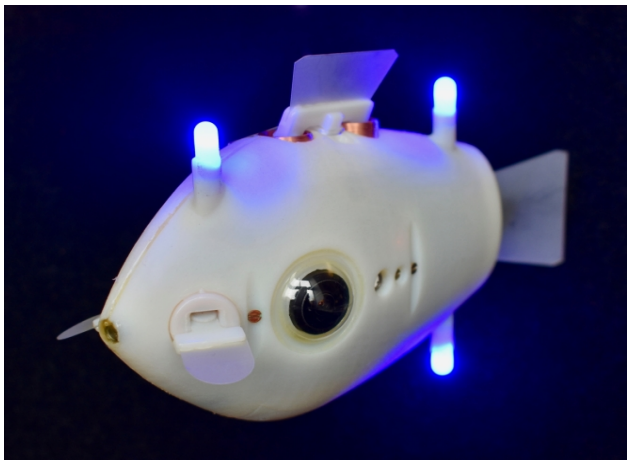

Figure 2.7: Bluebot robot

The goal of studying marine life through swarm technology may bring great benefits in the future. As with land and aerial robots, it is expected that such systems will be able to work collectively to achieve common objectives.

For example, when a group of fish swims in line, the leading fish generates a flow pattern that reduces water resistance for those behind, allowing them to expend less energy and swim more efficiently than if swimming alone [37].

2.2.4 Business Solutions and Applications

Although bioinspired systems have been discussed for some time, only a few applications are currently available on the market.

When examining larger AUV development companies such as L3Harris (with its Iver AUVs), RTSYS, SAAB, and others, none have presented a commercial solution with bioinspired propulsion systems.

However, there are numerous prototypes developed by universities, naval academies, and through partnerships between companies and institutions. Since this technology also has military relevance, it is not easy to obtain detailed information or data on real applications close to a final prototype.

During the research, a company based in Beijing, called ROBOSEA[38], was identified as developing this type of robot for educational and professional purposes. One of its best-known platforms, the Robo-Shark, is shown in Figure 2.8.

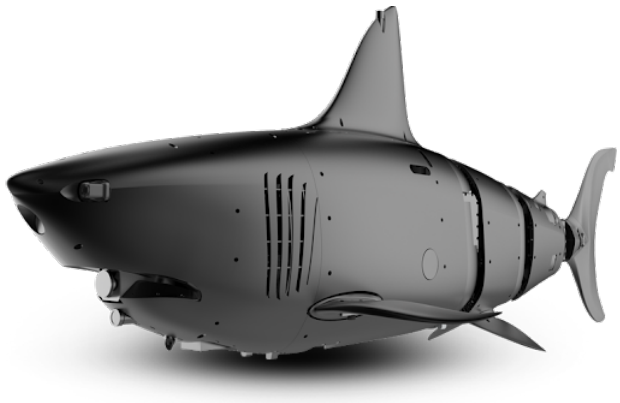

Figure 2.8: Robo-Shark

Although there are not many commercial prototypes yet, numerous studies and experimental platforms are under development in areas such as filtering microplastics [39], exploration and documentation of marine life, analysis and repair of marine structures, surveillance, and more.

2.3 Artificial Intelligence

The development of artificial intelligence began later due to its dependence on advances in computing power.

The first paper to formalize the idea of neural networks was published in 1943 by McCulloch et al. [40], describing artificial neurons as logic circuits.

In 1950, Turing [41] published the Turing Test, a criterion for evaluating machine intelligence. Turing discussed the possibility of machines thinking and proposed the "Imitation Game," a test whose objective was to make it impossible to differentiate whether the interrogator was interacting with a

human being or a machine. Currently, it is understood that this test is not sufficient to determine whether a machine is more intelligent than a human.

The term artificial intelligence only emerged in 1956 at the Dartmouth Conference organized by John McCarthy et al. [42], where the objective was defined as creating machines that could simulate human intelligence.

In 1961, James Slagle [43] created the first AI program capable of solving integrals.

Five years later, Joseph Weizenbaum [44] created one of the first chatbots.

In 1997, a supercomputer called Deep Blue made history by defeating the world chess champion for the first time.

Due to advances in computing, increasingly intelligent and lightweight models have emerged over time.

Over the last decade, generative models have become a major breakthrough in the field. They sparked widespread public interest and highlighted the need for digital transformation in public and private organizations.

These models possess general knowledge across various subjects, as they were trained on large-scale data from the internet. With access to such massive datasets, they are now capable of generating text, creating images, and much more. From these early models, more complex systems have evolved, including multi-agent models, where multiple specialized agents collaborate to solve tasks.

The long-term vision remains the development of a general model, one with cognitive abilities that can understand, adapt, and make decisions, even when faced with situations it was never explicitly trained for.

2.3.1 Advantages and Disadvantages of Using AI

Artificial intelligence currently has both advantages and disadvantages, as is the case with any technology. Generative models were gradually introduced to the public, marking a major breakthrough in the industry. However, along with these advances come concerns related to the security of personal data, as well as the positive and negative impacts that artificial intelligence may have on society and businesses.

Some of the problems of artificial intelligence

AI presents several challenges [45] that have persisted for many years and remain difficult to solve. The industry continues to focus on addressing these issues in order to obtain a clearer explanation of how networks process data, eliminate heuristic processes, and arrive more quickly at an optimal solution. Some of the main problems are:

1. **"Black box":** One of the problems with AI is that it operates as a "black box" [46]. While the inputs and outputs are observable, the internal decisions of the network and the relationships that determine the presented outputs are not fully understood. Substantial investment and research have been directed toward unraveling this problem.

2. **Determining the proper network structure:** Another major challenge concerns the rapid growth in the number of models and the lack of established rules for defining network structures. It is typically a process of trial and error, where training parameters such as learning rate, number of nodes, and stop criteria are defined empirically, without fully understanding the impact of these choices, all while attempting to avoid underfitting or overfitting problems [47].

3. **Data:** In the past, data processing was a limitation: the theory existed, but there was no way to apply it. Today, the situation is reversed, as computational resources are available, but there is a substantial shortage of data in the industry. This has led to extensive discussion about datasets and the generation of artificial data through simulations and digital twins.

4. **Computational resources:** Computing resources remain a challenge, particularly from an environmental perspective, as training large-scale models requires substantial energy. Research is ongoing into reducing training time through more efficient algorithms and hardware accelerators such as GPUs and TPUs.

5. **Training duration:** At present, there is still no reliable way to predict training time in advance.

Some of the advantages of artificial intelligence

Among the various benefits of AI, some notable advantages include:

1. **Implicit detection of complex nonlinear relationships:** One of the key advantages of neural networks is their ability to automatically identify complex, nonlinear interactions among input variables. This eliminates the need for users to explicitly define such interactions, allowing them to focus on optimizing model parameters such as weights and biases rather than manually engineering relationships.

2. **Distributed parallel processing:** Another advantage is the ability to divide tasks across multiple machines. Distributing tasks through clustering improves resource management and allows systems to scale efficiently.

3. **Machine learning and generalization:** Neural networks can learn from historical data and generalize patterns to make decisions in similar situations.

4. **Fault tolerance:** Neural networks also exhibit fault tolerance, as even if part of the structure fails (e.g., corrupted neurons), the model continues to generate outputs thanks to inherent redundancy.

2.3.2 Evaluation Metrics

Depending on the type of problem at hand, different metrics are used.

There are classification problems and regression problems. Both terms are related to prediction:

1. In the case of classification, the aim is to predict the class to which each data sample most likely belongs.

2. In regression, the goal is to predict a continuous value based on what the model has learned during training.

The problem addressed in this work is a regression problem, where the network is trained on a dataset to estimate thrust and power values. This

approach requires less computational effort compared to CFD solvers, which are far more time-consuming.

Common metrics for regression problems include:

- Mean Squared Error (MSE):

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \tilde{y}_i)^2. \tag{2.11}$$

  The MSE measures the average of the squared differences between the actual values y_i and the predicted values $\tilde{y}_i$.

  Here, n is the number of samples in the dataset, y_i represents the true observed value, and $\tilde{y}_i$ is the predicted value produced by the model.

- Mean Absolute Error (MAE):

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \tilde{y}_i|. \tag{2.12}$$

  The MAE calculates the average magnitude of the errors between the actual values y_i and the predicted values $\tilde{y}_i$, without considering their direction.

  As in MSE, n denotes the number of samples, y_i the true observed value, and $\tilde{y}_i$ the model prediction.

- Coefficient of Determination (R^2):

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \tilde{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}. \tag{2.13}$$

  The R^2 metric indicates the proportion of the variance in the observed data that is explained by the model.

  In this expression, y_i are the observed values, $\tilde{y}_i$ are the predicted values, and $\bar{y}$ is the mean of the observed values.

  The value of R^2 ranges up to 1, corresponding to a perfect prediction, while negative values indicate that the model performs worse than using the mean $\bar{y}$ as a predictor.

2.3.3 AI Implementations

Artificial intelligence is currently applied across a wide range of fields. This subsection, however, does not aim to provide a comprehensive overview of its applications, but rather focuses specifically on how AI is used to optimize the performance and efficiency of airfoils.

Most studies in the literature that combine airfoils and AI focus on optimizing shape and stiffness, enhancing stability, or conducting validation experiments using AI-driven models. These models assess fin performance based on data extracted from fluid dynamics simulations and numerical studies, where the typical objectives are to optimize parameters to maximize the lift-to-drag ratio ($\frac{\text{Lift}}{\text{Drag}}$), increase the lift coefficient, and minimize the drag coefficient.

For example, in an article by Sawant et al. [48], a method combining reinforcement learning and Trust Region Policy Optimization (TRPO) was used to improve the aerodynamic efficiency of different airfoils, including NACA 2412, NACA 0012 and NACA 0015, through the optimization of parameters such as camber, camber position, and maximum thickness. After testing, the airfoil that achieved the best results was the NACA 0012, showing an improvement of 22.3 % in C_L, along with a substantial reduction in C_D.

In the study by Akram et al. [49], an S809 airfoil was optimized using a system that combined MATLAB and XFOIL to analyze the aerodynamic parameters. A genetic algorithm was applied in conjunction with two parametrization methods: Class Shape Transformation (CST) and Parametric Section (PARSEC).

In addition, Computational Fluid Dynamics (CFD) simulations were performed in ANSYS CFX using the Shear Stress Transport (SST) turbulence model to validate the results.

The study demonstrated that the CST method combined with the genetic algorithm outperformed the PARSEC approach, achieving substantial improvements in C_L (+24.6 %) and lift-to-drag ratio $\frac{L}{D}$ (+12.4 %) in the CFD simulations.

Chapter 3

Methodology

In this chapter, the implemented methodology is presented. The chapter is divided into three main sections. The first covers the experimental rig, including the design created in CATIA and the equipment used. The second describes the implementation of the MATLAB solution, which was applied to the servo control system and sensor data extraction. The third section details the neural network architecture and its parameters.

3.1 Experimental Rig

The system was designed to measure the thrust force generated by the fin's flapping motion using a load cell, while a current sensor records the electrical current required to perform different movements. In this setup, the total electrical power consumed by the servo motor is captured, rather than the hydrodynamic power.

Hydrodynamic power refers only to the energy transferred to the water to produce thrust, and propulsive efficiency is typically calculated from this force. In this study, however, the total power consumed by the servo motor is considered, which includes all energy losses in the system such as gear friction, electrical losses, and the energy required to overcome inertia.

Hydrodynamic power does not depend on the motor or the electrical system. As long as the system and the test environment remain unchanged, hydrodynamic power remains constant.

In contrast, total electrical power varies significantly with the servo motor

used. A more efficient motor will consume less energy to achieve the same motion, while a less efficient one will consume substantially more, leading to very different results.

It is important to understand that any modifications made to the system described in this study may alter the results, especially if a different engine or hydrofoil is used.

The hydrofoil itself directly influences the results, since its size, shape, and flexibility determine its behavior under different conditions.

Being subject to various forces such as weight, lift, buoyancy, drag, and thrust, the hydrofoil plays a key role in the system. Through its oscillation, driven by the servo motor, the water is agitated and vortices are generated, enabling thrust measurement.

The choice of the fin or foil usually depends on the study and application scenario, but it is often based on aspect ratio. A long, narrow, lunate-shaped fin [50] has a high aspect ratio, which allows for high cruising speeds and high energy efficiency, although with reduced maneuverability. Conversely, a fin with a lower aspect ratio, shorter and wider, does not achieve such high cruising speeds but provides greater maneuverability.

In this study, a symmetrical hydrofoil with an E836 airfoil is used. The airfoil is defined based on its geometry. Figure 3.1 shows the geometry of this hydrofoil, extracted from the Airfoil Tools website [51].

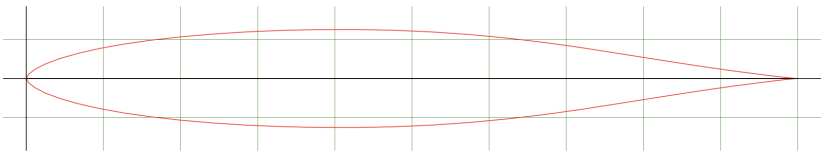

Figure 3.1: Eppler E836 hydrofoil shape

The choice of a symmetrical profile, which is more common, relates to the main objective of this experimental methodology, which is precisely to demonstrate its feasibility.

In the context of this methodology, only a single profile was used, with dimensions of 10 cm $\times$ 10 cm, printed in PETG.

In this initial phase in which this methodology is presented, the focus was not on testing different ratios, formats, or materials. However, it is important

to note that all these issues are pertinent for future studies because they offer the advantage of allowing for a much more comprehensive and generalized dataset.

Since this printed hydrofoil had no internal filling, it is crucial to submerge it as deeply as possible to minimize air trapping; otherwise, buoyancy effects may occur, negatively affecting the results.

Figure 3.2 illustrates the hydrofoil coupled to the servo and some of the main forces acting on it. The figure shows part of the system, where a fin is fixed to a servo motor housed inside the support structure. This mechanism is attached to a load cell at the top, which measures the force, and then mounted on a custom-built crane, as shown in Figure 3.3.

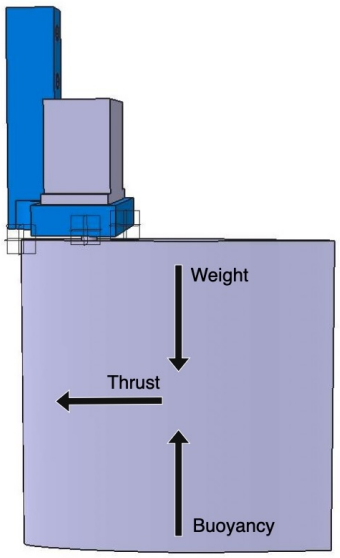

Figure 3.2: Forces applied to the fin

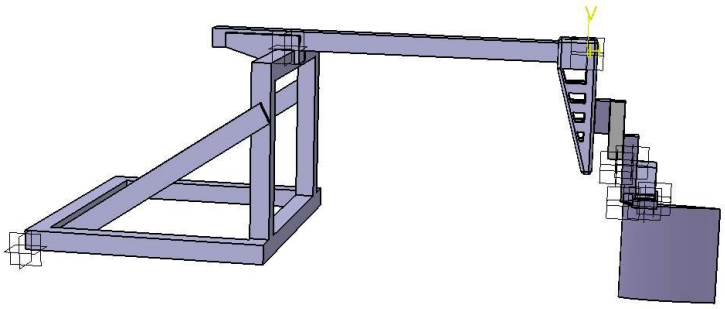

Figure 3.3: Overview of the experimental rig

Figure 3.4: Photograph of the experimental rig

The system developed for this study consists of aluminum profiles and parts designed in CATIA V5 that were 3D printed in PETG and PLA. It is composed of the following components:

- E836 hydrofoil 3d printed in PETG;
- Joy-IT COM-Motor02 servo motor;
- Joy-IT Load cell 1 kg + amplifier with a 24-bit A/D converter HX711;
- INA219 current sensor module;
- Custom-built servo motor hub;
- ESP-WROOM-32 microcontroller;
- Aluminum crane with connectors 3d printed in PLA.

3.1.1 Sensors

This subsection describes and analyzes the characteristics of the sensors selected for the development of the system, namely the load cell, the current sensor, and the servo motor.

3.1.1.1 Load Cell

The load cell was chosen based on the low-magnitude forces to be measured. A 1 kg capacity cell was selected, as it was the smallest available at the nearest electronics store and offered the best precision for the task. It has an accuracy of ±0.02 % full scale (FS), which corresponds to an average error of approximately 0.2 g. Figure 3.5 shows the load cell, used as a force sensor, and its amplifier, the HX711 module.

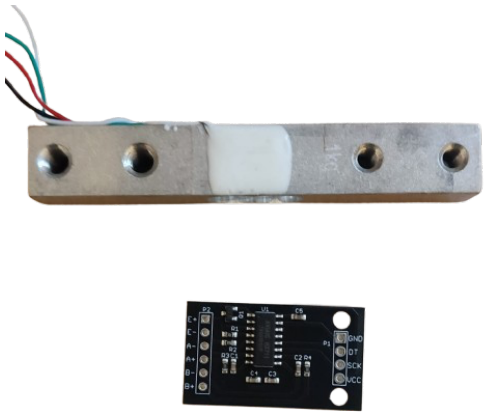

Figure 3.5: Load cell + amplifier

In this work, the $HX711\_ADC$ library [52] was used to obtain data from the HX711 sensor, and its implementation is described in the next subsection.

3.1.1.2 Current Sensor

To measure the electrical consumption of the servo motor, the INA219 current sensor was employed. This device measures currents up to 3.2 A and voltages up to 26 V, providing good resolution for low- to medium-current applications, with typical errors no greater than ±1 %. The sensor used in

this study is shown in Figure 3.6. For data acquisition, the Arduino INA219 library [53] was adopted.

Figure 3.6: INA219 current sensor

3.1.2 Microcontrollers

Among the available microcontrollers—Arduino Uno R3, Arduino Due, ESP-WROOM-32, and ESP32-S2-SOLO-2U-N8R2—the ESP-WROOM-32 was selected. Each of these options has distinct characteristics in terms of number of pins, clock frequency, memory capacity, and processing power.

In this work, the main limitation arose from the fact that the load cell measured a maximum of 80 data points per second, which meant that the microcontroller did not need the highest performance but rather a balance choice between cost and scalability. Figure 3.7 shows the selected microcontroller.

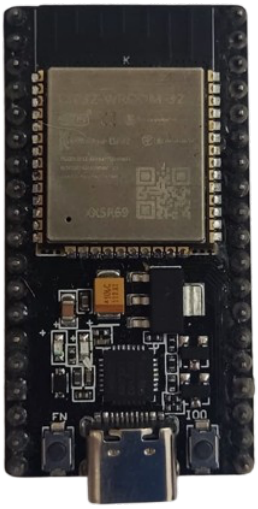

Figure 3.7: Esp32 microcontroller

3.1.3 Servo Motor

The servo motor used was a Joy-IT COM-Motor02, which operates between 5 V to 7.4 V, delivers a maximum torque of 18.3 kg·cm to 21.5 kg·cm, and achieves a no-load speed of 0.15 s to 0.16 s for a 60° rotation. Figure 3.8 shows the motor viewed from different angles.

The servo motor has a nominal angular speed of 400 °/s, which corresponds to 6.98 rad/s, calculated as follows:

$$\omega = \frac{\Delta\theta}{\Delta t} = \frac{60}{0.15} = 400 \text{ °/s} \tag{3.1}$$

$$\omega_{\text{rad/s}} = 400 \times \frac{\pi}{180} = 6.98 \text{ rad/s} \tag{3.2}$$

Figure 3.8: Servo motor viewed from different angles

3.2 Implementation

This section describes the implementation of the system, which includes the assembly of the electrical circuit, the development of the model in MATLAB/Simulink, and the development of the neural network. Figure 3.9 shows the system's architecture, where the ESP32 microcontroller connects to the sensors and actuator. The configuration is carried out through Simulink, which enables the extraction of data later provided to a neural network for training, performed in Google Colab.

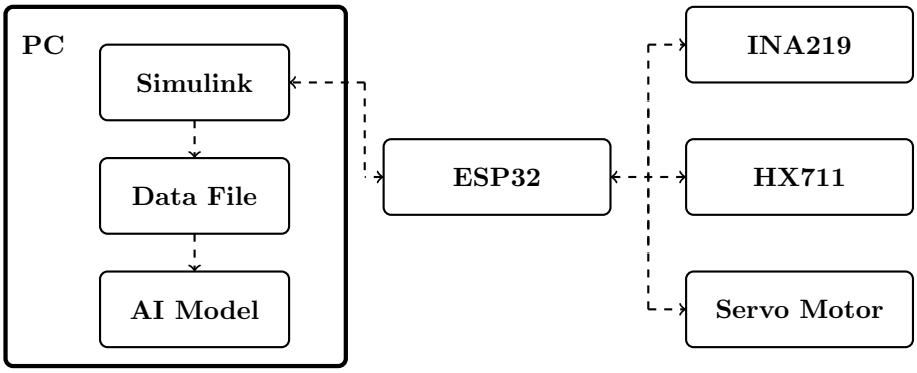

Figure 3.9: System's architecture

3.2.1 Circuit Implementation

The circuit was designed as a centralized solution, controlled by a single board responsible for driving the servo motor and recording sensor data. Figure 3.10 illustrates the schematic of the developed system. In this configuration, the ESP32 is connected to the current sensor, the load cell, and the servomotor. It powers both sensors, while the servo motor is powered separately by a 6 V battery.

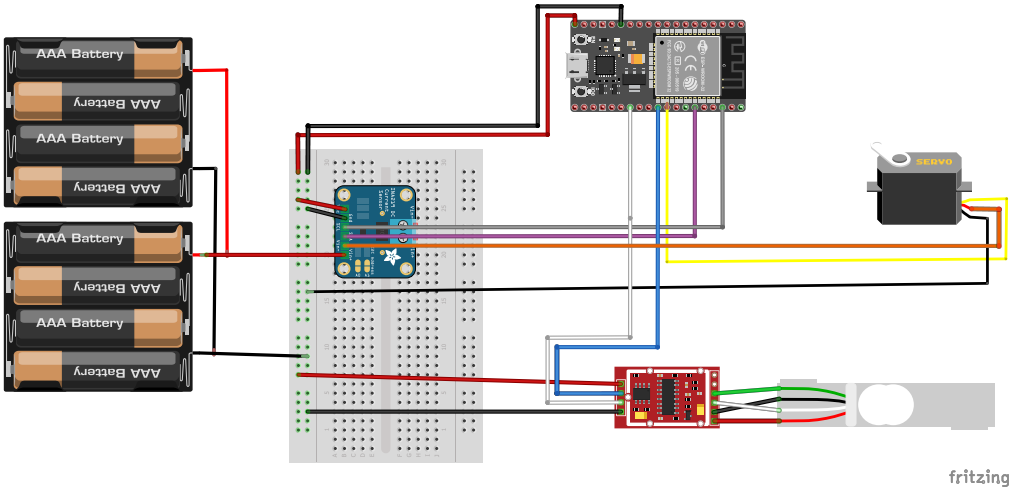

Figure 3.10: Circuit diagram created with Fritzing

3.2.2 Simulink Implementation

Once the circuit was assembled, Simulink (MATLAB 2024b) was used to program the microcontroller and execute all tasks. Simulink is a block diagram environment that enables the design of systems with multidomain models and allows simulations before deploying to hardware.

Figure 3.11 shows the model created in Simulink, which is composed of several blocks, each with a specific function: red blocks correspond to the servo motor control system, the blue block is responsible for reading sensor data, and the cyan block sends the data to the Workspace environment for later analysis.

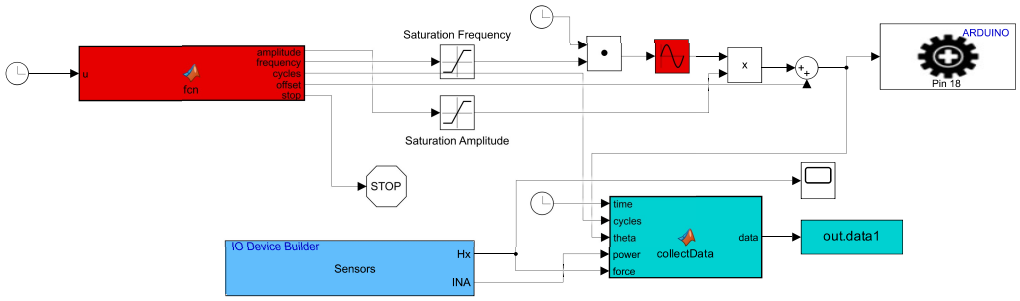

Figure 3.11: Simulink overview

3.2.2.1 Servo Motor Control

The servo motor control is one of the most critical components, as it manages the motion applied to the servo and directly influences the foil's movement. Since this system affects all the data collected, it must be implemented consistently and reliably.

Each data cycle corresponds to a specific combination of amplitude and frequency. For every cycle, all force measurements are recorded over a 40-second period, during which the servo motor remains at rest for the first and last 10 seconds.

The amplitude and frequency defined in each cycle pass through saturation blocks that validate whether the values are within the acceptable range, avoiding errors. After this, the frequency is combined with the clock signal in a dot product block, as illustrated in Figure 3.12.

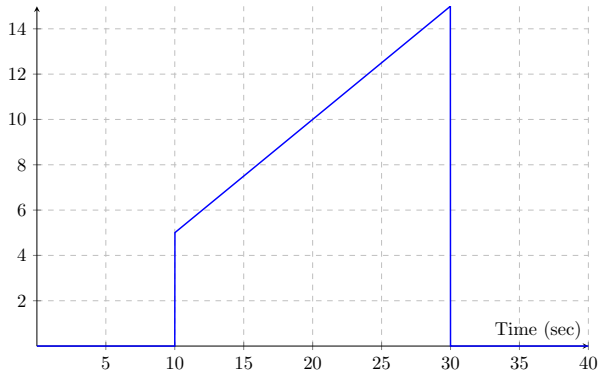

Figure 3.12: Signal generation for $A_\theta = 30°$ and $f = 0.5\,\text{Hz}$

The resultin signal is then processed by a sine wave block, where the sinusoidal equation is applied, producing a waveform like the one shown in Figure 3.13.

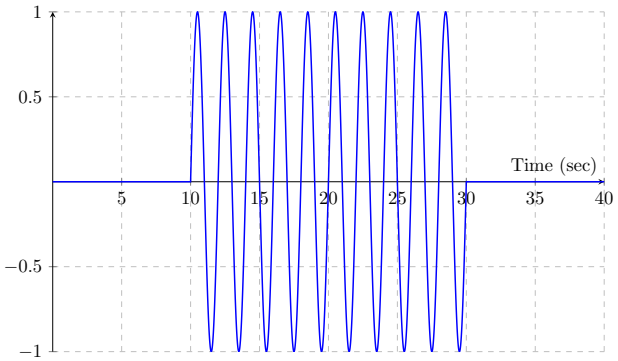

Figure 3.13: Signal generation for $A_\theta = 30°$ and $f = 0.5\,\text{Hz}$

The sinusoidal signal and its frequency are subsequently processed by a product block and passed through a sum block, where an offset can be added if required. Finally, the value is multiplied by the amplitude and an offset is added, which, for the purposes of this study, was not necessary. The result of this transformation can be seen in Figure 3.14.

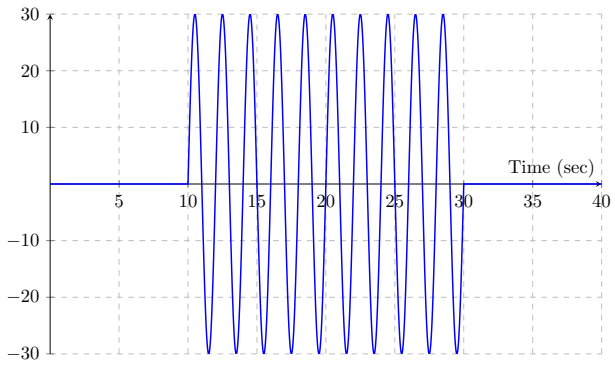

Figure 3.14: Signal generation for $A_\theta = 30°$ and $f = 0.5\,\text{Hz}$

The process is completed when the resulting signal is sent to the servo motor block to apply the motion. After 40 seconds, a new combination of parameters is tested, and the procedure is repeated until the defined limit is reached.

3.2.2.2 Sensor Data Acquisition

For sensor data acquisition in Simulink, since no direct libraries were available, the IO Device Builder tool [54] was used to integrate the existing Arduino IDE libraries.

A crucial step was defining the libraries and their dependencies correctly. In this work, the libraries introduced in previous subsections were used.

The *HX711_ADC* library [52] required modifications, as it included a default filter that smoothed the data. This filter was disabled so that filtering could be applied later in the pipeline. When creating the IO Device Builder block, it was also necessary to select the appropriate Wire library depending on the board in use.

In the final step of implementing the IO Device Builder task, it is important to generate a driver in CPP. This driver will be used to write the code that configures the sensors and collects data from them.

At this stage, the system was ready for data acquisition. The block was fully configured and could be connected either directly to a data storage block or routed through a function that standardizes all values to the double format before storage.

After collecting all the data in time-series format, a MATLAB script was

developed to convert the data into a *.txt* file. This script not only performed the conversion but also calculated the angular velocity, defined as the rate of change of the angle with respect to time, $\dot{\theta}$.

The angular velocity is mathematically defined as the time derivative of the angular position, expressed as:

$$\dot{\theta} = \frac{d\theta(t)}{dt} \tag{3.3}$$

In some contexts, the angular velocity may also be expressed using partial derivatives:

$$\frac{\partial \theta}{\partial t} = \dot{\theta}(t) \tag{3.4}$$

The parameters θ and $\dot{\theta}$ are essential input variables in dynamical systems, as they determine the angular position and the velocity at which the system must reach that position.

The final data structure is represented in the following vector:

$$\begin{bmatrix} t & \theta & P & F & \dot{\theta} & A & f \end{bmatrix}.$$

3.3 Neural Network Modeling

The neural network selected for this study was defined according to the objectives, which were to predict force and power from a set of input parameters. Among the various available model options, such as feedforward neural networks, recurrent neural networks, and others, the Kolmogorov–Arnold Network (KAN) was chosen.

In addition to the KAN model, other models were used for comparison, namely Gradient Boosting and Long Short-Term Memory (LSTM). However, greater emphasis was placed on the KAN model because it is a more recent and highly promising approach.

Based on the objectives and on the results obtained from the correlation matrix, the input parameters of the network were identified, and a conceptual diagram was created, as shown in Figure 3.15.

Figure 3.15: Conceptual diagram of the AI model

According to the Kolmogorov–Arnold theorem, any continuous multivariate function can be represented as a superposition of continuous univariate functions. This suggests that complex high-dimensional functions can be decomposed into simpler one-dimensional components, making it possible to model intricate relationships through a structured combination of single-variable functions. The Kolmogorov–Arnold theorem [55, 56] is mathematically expressed as:

$$f(x) = \sum_{q=1}^{2n+1} \phi_q \left(\sum_{p=1}^{n} \psi_{q,p}(x_p) \right), \tag{3.5}$$

where ϕ_q and $\psi_{q,p}$ are continuous univariate functions (i.e., functions of a single variable). The index q ranges from 1 to $2n+1$ in the outer summation, corresponding to the functions ϕ_q, while p ranges from 1 to n, indexing the components of the input vector $x = (x_1, x_2, \ldots, x_n)$.

The architecture of the KAN model is illustrated in Figure 3.16. Each input component x_p is passed through a set of functions $\psi_{q,p}$. For each fixed q, these transformations are summed across all p, producing an intermediate result. The results of these inner sums are then passed through the functions ϕ_q, forming the second layer of univariate transformations. Finally, these outputs are summed over q to produce the final output, $f(x)$.

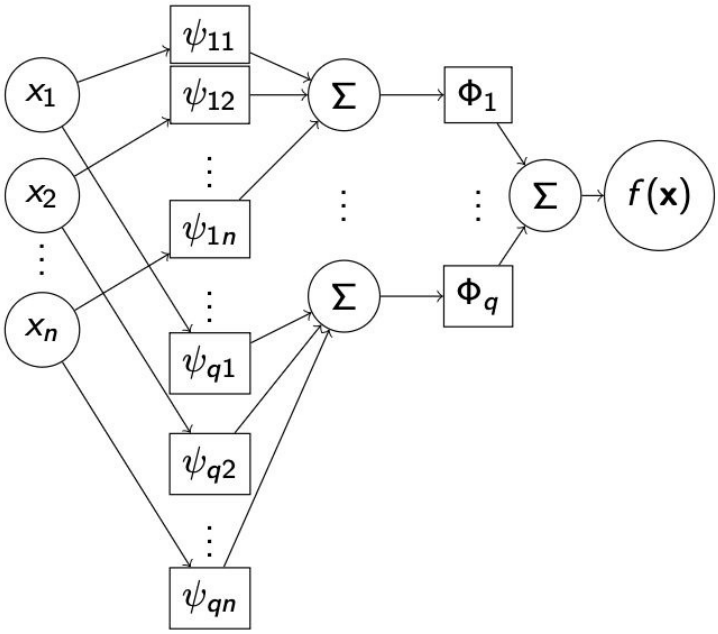

Figure 3.16: KAN model

In practical implementations, each univariate function (ϕ_q and $\psi_{q,p}$) is parameterized using B-splines, which are piecewise polynomial functions defined over a set of intervals. These splines are governed by parameters including knots, polynomial degree, and control coefficients. For a univariate function $\psi_{q,p}(x_p)$, the B-spline representation is:

$$\psi_{q,p}(x_p) = \sum_{i=1}^{m} c_{q,p,i} B_{i,k}(x_p; t) \tag{3.6}$$

where $B_{i,k}(x_p; t)$ is the $i-th$ B-spline basis function of degree k, defined over a knot vector $t = (t_1, t_2, ..., t_{m+k+1})$, and $c_{q,p,i}$ are the trainable coefficients. The knot vector forms a grid that partitions the input domain, often initialized as uniform but adaptable during training to focus on regions with complex behavior.

During training, these spline parameters (knots and coefficients) are adjusted to fit the data, enabling the model to capture complex nonlinear relationships between inputs and outputs by carefully composing and combining these learned one-dimensional components.

In the context of this study, two optimization algorithms were tested:

Adam and limited-memory BFGS (L-BFGS).

The Adam algorithm is a first-order gradient-based optimization method for stochastic objective functions that uses adaptive estimates of the first- and second-order moments of the gradients. It combines the advantages of AdaGrad, which handles sparse gradients effectively, and RMSProp, which performs well in non-stationary problems, resulting in faster and more stable convergence in many machine learning scenarios.

The limited-memory Broyden–Fletcher–Goldfarb–Shanno (L-BFGS) algorithm [57] is a second-order Quasi-Newton method widely used for large-scale nonlinear optimization. It approximates the inverse Hessian matrix using a limited history of gradients. Unlike the conventional BFGS method, which requires full Hessian storage, L-BFGS stores only a reduced set of gradient and position vectors, making it efficient for high-dimensional problems.

Although the present case does not involve a high-dimensional problem, using L-BFGS to optimize KAN models often yields satisfactory performance, as the method utilizes second-order information and converges more quickly. Nevertheless, this combination must be monitored carefully, since the rapid adjustment capacity can lead to overfitting, making validation and monitoring essential.

The loss function employed was the Mean Squared Error (MSE), which measures the mean squared difference between predicted values and target values. This algorithm enables the model to be optimized toward lower error and, consequently, higher predictive accuracy.

Chapter 4

Results and Discussion

In the previous chapters, the principles of the flapping mechanism were presented based on investigations carried out by other authors. Then, the construction of a methodology that could capture the force and power generated by a bioinspired fin during a flapping movement was proposed.

To build this system, a list of of material resources was defined, and then it was demonstrated how to create in MATLAB Simulink a system that could apply a sinusoidal movement to the servo, and how to obtain data from the sensors. It was then explained that the extracted data would be used to train a neural network that could understand how the system works.

In this chapter, the results of the methodology will be presented. For better explanation, this chapter will be divided into data analysis and network training results.

4.1 Data Analysis

This section focuses on analyzing the data obtained from our experimental rig. It will be demonstrated how the data was adjust so that it could then be applied to a neuronal network.

First of all, it is important to highlight that the MATLAB solver was set to operate at a frequency of 80 Hz which is the maximum frequency at which the load cell used can handle. Running the entire system at 80 Hz allowed data to be captured with maximum detail.

Since a load cell is being used, and since it is intended that the cell

be properly tared at the beginning of each measurement, along with the calibration process that helps measure the generated force, an additional method was developed to assist the system in taring.

For each time-series collected from the system, the system had to remain still for 10 seconds at first, then perform the oscillatory movement for 20 seconds and then remain still for 10 seconds again.

Figure 4.1 shows the force and power data collected with this delay time for the data set with amplitude 30° and frequency of 0.5 Hz. Regardless of the amplitude and frequency, the power and force graph should be more or less similar to the one shown. It must be possible to look at the graphs and verify that there is a correlation between them, because the increase in energy consumption must have consequences at the force level and this must be visible regardless of the frequency.

This, however, does not mean that higher power consumption necessarily results in greater force. It does mean that the two recorded forces must have similar wavelengths, taking into account, of course, the possibility of time lags. Even so, the sinusoidal motion imposed on the servomotor should be visible, to some extent, in both the force and power graphs.

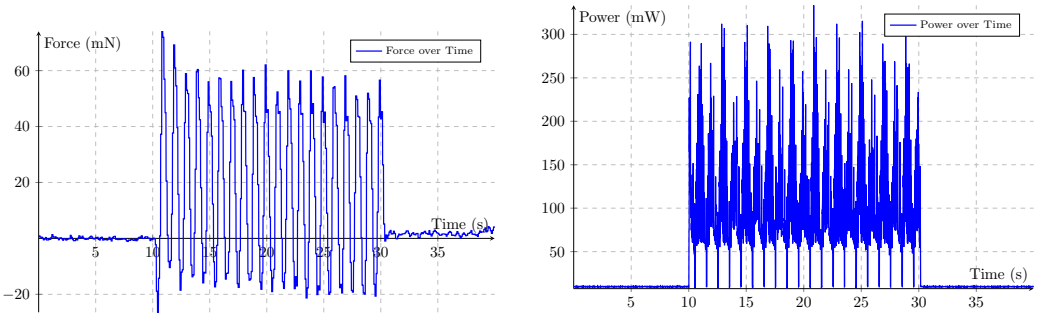

Figure 4.1: Thrust force and electrical power versus time, with original data at $A_\theta = 30°$ and $f = 0.5\,\text{Hz}$

The force graph shows additional peaks because the servo motor is controlled by a PWM signal. This type of signal repeatedly switches the power on and off, generating consecutive variations in the measured force. Because of this characteristic, the force graph always requires careful analysis.

Another important consideration is ensuring that vortices generated by the system do not flow back into it. Poor vortex shedding can cause backflow,

introducing turbulence that distorts the measurements, reduces reliability, and can lead to incorrect interpretations of the system's performance. Ambient temperature can also affect sensor performance and, consequently, the accuracy of the collected data.

Two measures were adopted in this work to minimize such disturbances. First, time series were collected over reduced time intervals to avoid problems such as sensor drift. Second, the data were tared using the average of the first ten seconds during which the system was at rest, eliminating possible disturbances and validating that the measured force was as close as possible to the actual value. This procedure was applied separately to each time series.

The final ten seconds of each sequence also play a crucial role. If the force value does not gradually decrease to zero at the end of each oscillation, it is usually a sign that something went wrong and that the measured forces are incorrect. Some residual water motion is normal, especially in test environments with poor drainage or inadequate dimensions, but in these specific cases the force should decrease over time. If the signal fails to drop quickly after the flapping movement, this may indicate that the water is too rough or that sensor drift issues are present.

After applying the taring procedure to the system, it was no longer necessary to retain the data corresponding to the periods when the servo motor was stopped.

Figure 4.2 shows the graphs for an amplitude of 30° and a frequency of 0.5 Hz, over a shorter time interval, after the taring process had been applied.

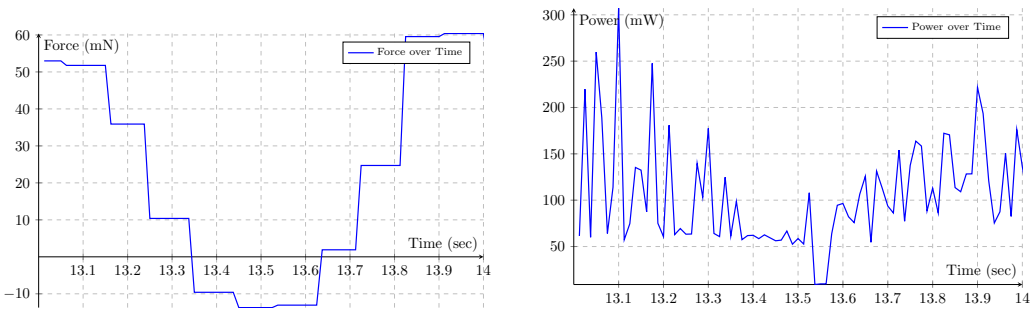

Figure 4.2: Thrust force and electrical power versus time, with unfiltered data at $A_\theta = 30°$ and $f = 0.5\,\text{Hz}$

In the previous figures, the force plot on the left displayed two initial peaks distinct from the subsequent ones. This effect resulted from the start of the sinusoidal motion, whereas the intention was to record the motion already in progress, the so-called periodic condition. Consequently, the first three seconds were removed from each time series to eliminate this effect and avoid impairing prediction accuracy.

To smooth the data without compromising its integrity, a moving average with a 5-point window was applied to the force data, and a moving average with a 2-point window was applied to the power data. The purpose of applying this light smoothing filters was to preserve the dynamics of the power signal as much as possible while reducing high-frequency noise.

Load cells operating at their maximum sampling frequency may exhibit noise due to saturation and other effects; applying this filter mitigates such issues. A similar filtering process was applied to the power measurements, but with a shorter window to maintain the integrity of this more complex data while still correcting potential measurement errors. The results of these changes are presented in Figure 4.3.

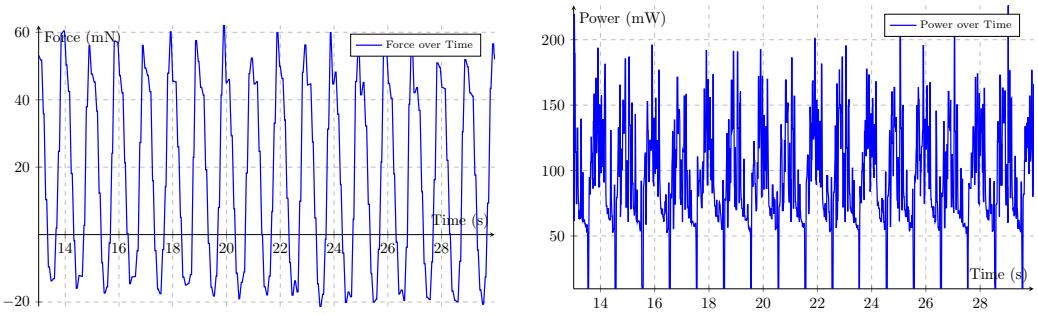

Figure 4.3: Thrust force and electrical power versus time, with filtered data at $A_\theta = 30°$ and $f = 0.5\,\text{Hz}$

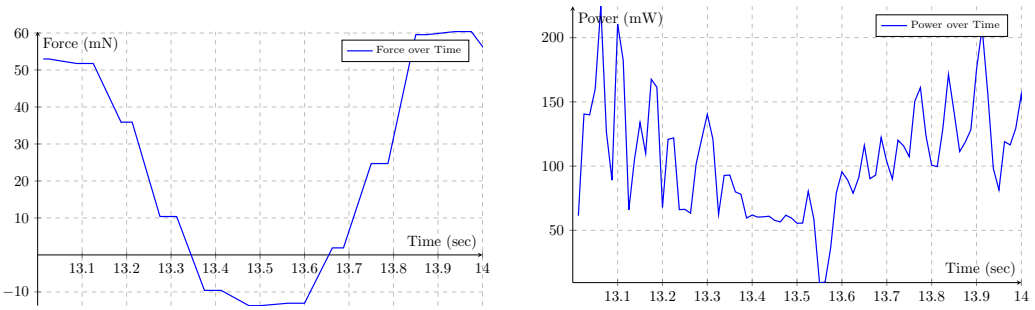

Figure 4.4: Thrust force and electrical power versus time, with filtered data at $A_\theta = 30°$ and $f = 0.5\,\text{Hz}$ (close-up)

Once these changes were made, the efficiency of the collected data was analyzed to determine which parameters yielded the best results, in other words, which combinations transformed the consumed energy into thrust more effectively.

Figure 4.5 displays a graph representing all measured the amplitude–frequency combinations, with the highest average efficiency highlighted in yellow and the lowest values shown in darker colors. It should be noted that the efficiency depicted in this graph refers to the overall system efficiency, not hydrodynamic efficiency. Based on the tested combinations, the best average efficiency was observed at an amplitude of 50° and a frequency of 0.75 Hz.

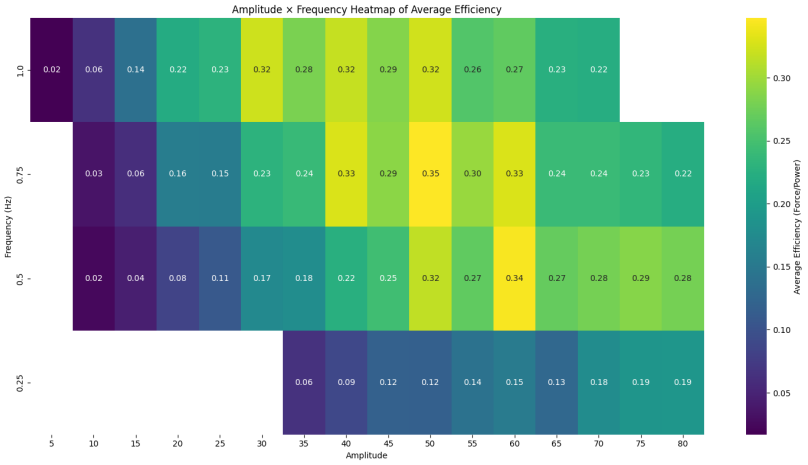

Figure 4.5: Matrix of all amplitude and frequency combinations, highlighted by efficiency

From this graph, it can be concluded that the best combination was able to convert 35 % of the available energy into useful mechanical power.

A total of 53 time series were collected. After processing, each time series contained 1,360 records, which, multiplied by the 53 time series, results in a total of 72,080 data points.

The graph provides a fundamental reference for evaluating the performance of the airfoil used in this study and allows comparison with other airfoil designs. The blank cells appear in the matrix for two reasons. First, the load cell was unable to register very low force values. Second, at high amplitudes combined with high frequencies, the fin became unstable, resulting in inconsistent data and forces significantly lower than those measured at lower frequencies for the same amplitude. In such cases, it was not meaningful to represent these phenomena.

Although further measurements could not be performed without substantially exceeding the available amplitude range, several studies indicate that the use of a flexible fin or tail would allow testing new ranges of amplitude and frequency not yet been explored. Such configurations could potentially yield efficiencies higher than those obtained with the rigid fin. The most likely outcome would be to identify an optimal combination at amplitudes below 50° and frequencies above 0.75 Hz.

For a more detailed view of the data, a scatterplot was created, as shown in Figure 4.6. The plot provides a comprehensive representation of the dataset. Based on the scatterplot, it is possible to analyze the collected data more effectively and observe, for example, that increases in frequency and amplitude are correlated with higher energy consumption.

More complex patterns can also be observed. The power data exhibit an asymmetric distribution with a right-hand tail, indicating that most recorded values were low, with higher values occurring only rarely.

This observation is highly relevant and may indicate that, when training an artificial intelligence model, the model may have difficulty accurately predicting certain combinations—especially from 1000 mW onwards—due to the lack of data in that region, which prevents it from comparing patterns and adjusting sufficiently.

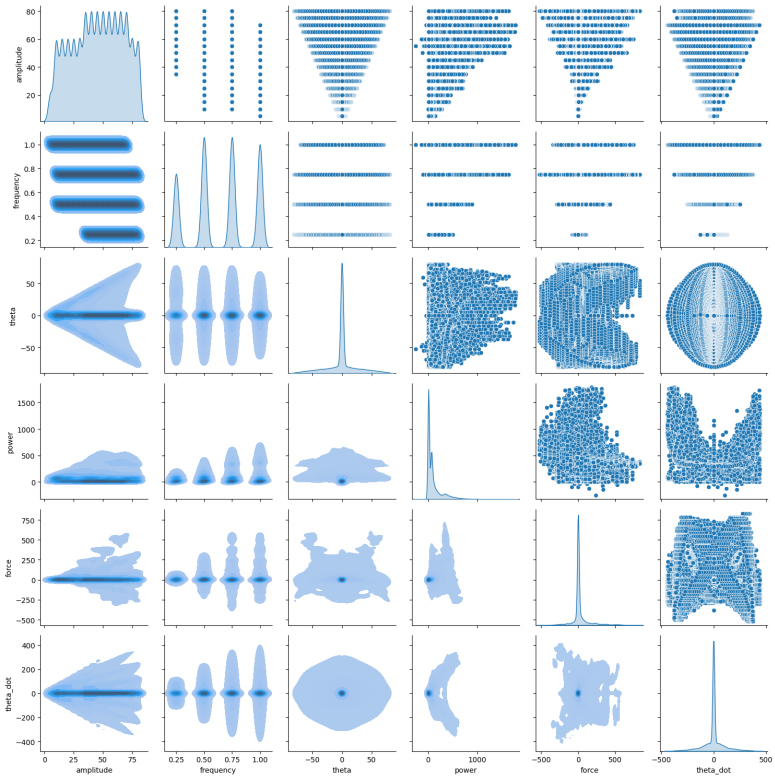

Figure 4.6: Scatter matrix

The force distribution appears symmetric because it includes both positive and negative values due to the dynamic oscillatory forces. However, when only positive values are considered, an asymmetric distribution emerges, with many values close to zero and a decreasing curve as higher frequencies are tested.

When comparing power data with force data, it is observed that both exhibit similar patterns when only positive values are considered. The main difference lies in the force range, which is narrower but simultaneously denser. This higher data density can facilitate the learning process of an artificial intelligence model, since the model has more reference points for comparison when estimating force, in contrast to power, whose dataset is less dense.

An additional noteworthy observation supports the earlier argument regarding flexible fins or tails. In the force–power scatterplot, the upper left region is sparsely populated. This area corresponds to the most efficient regime, where maximum force is achieved with minimal energy consumption. Categorizing the data by frequency shows that this region gradually becomes

occupied as higher frequencies are tested, suggesting that further exploration with flexible structures could unlock even greater efficiency.

In concluding this analysis of Figure 4.6, it is important to note that, in the force–power scatterplot, higher energy consumption does not necessarily imply the generation of greater propulsive force. This is an important misconception to highlight, as it shows that the system under study does not behave linearly, contrary to what one might assume.

The hydrodynamic analysis is inherently complex due to mechanical losses, friction in the servo motor, PWM signal delays, and variations associated with the fluid's dynamic loading—all factors that significantly influence the data obtained.

After these analyses, a correlation matrix graph was created, as shown in Figure 4.7.

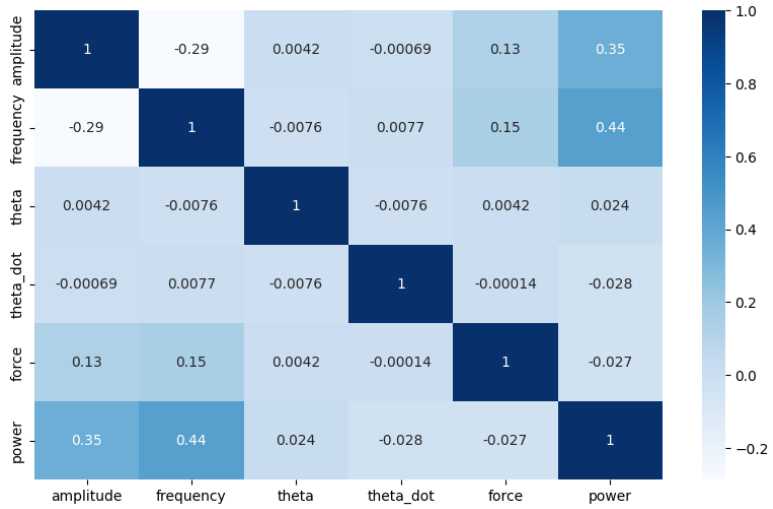

Figure 4.7: Correlation matrix

The matrix shows that amplitude and frequency are critical predictors of force and power, whereas θ and $\dot{\theta}$ appear less directly correlated.

Nevertheless, it is important to note that correlation matrices capture only simple linear relationships and do not account for temporal dynamics. Consequently, θ and $\dot{\theta}$ remain essential features for model learning, as their importance becomes evident once system dynamics are considered.

4.2 KAN Model Results

After analyzing and preparing the data, everything was ready to train the KAN model.

It is important to note that 80 % of the data was used for training and 20 % for validation. The split was performed so that each time series corresponding to a given amplitude and imposed frequency remained entirely in either the training set or the test set. This strategy ensures that no data leakage occurs by preventing any improper transfer of information between sets. The splitting process was carried out before applying standard scaling to the data, which also helps avoid data leakage.

Table 4.1 presents the parameters used to train the network.

Table 4.1: KAN network parameter configuration

width	[4, 10, 10, 2]
grid	5
k	3
optimizer	Adam
lr	0.001
lambs	1e-5
steps	100
batch	32
loss function	MSE
data division	Train 80 %; Test 20 %

The dataset was split into 80 % for training and 20 % for testing, a common and widely recommended procedure.

In this case, the data were not shuffled, as preserving their original temporal order allows the model to identify sequential patterns that would otherwise be lost. This enables the network to better learn the relationship between consecutive inputs and the underlying dynamics of the system.

The model was evaluated using the mean squared error (MSE), a widely used metric that calculates the average of the squared differences between the predicted values and the actual values in regression analysis.

To minimize error, the Adam optimization algorithm was used. This method is a stochastic gradient-based optimizer that iteratively updates the model parameters using estimates of first- and second-order moments of the gradients.

The training process was configured with 100 optimization steps, meaning that the algorithm performed 100 updates to iteratively reduce the MSE and converge toward an optimal solution.

In this model, a batch size of 32 was also defined, corresponding to the number of samples processed at once before each parameter update. A learning rate of 0.001 was used, along with an L2 regularization coefficient (lambda) of 0.00001.

In KAN, it is also possible to define parameters such as grid and k-squared. The grid specifies the discretization of the input domain, i.e., the set of points where the basis functions are evaluated. The k-square parameter determines the order of the B-spline basis functions.

Using the PyKAN library, a model was created with these parameters, employing a four-layer structure with two hidden layers, each containing 10 neurons.

It is also important to note that the data were standardized using a standard scaler before training the model, which helps improve performance. A standard scaler was chosen instead of normalization because it preserves the influence of natural data variance while centering the distribution, which is advantageous for gradient-based optimization methods such as Adam.

Tables 4.2 and 4.3 present the training results of the KAN model. The first table shows the model's ability to predict energy consumption, while the second table presents its ability to predict force.

Table 4.2: KAN model power results

Train R^2	0.9134
Test R^2	0.7420
Train MAE	32.7229
Test MAE	81.5911

Table 4.3: KAN model force results

Train R^2	0.9738
Test R^2	0.8628
Train MAE	14.7583
Test MAE	54.6684

For the training data, the figure shows that the model effectively captured the system's behavior, achieving prediction scores of 0.7420 for power and

0.8628 for force. This indicates that power is more difficult to predict than force and suggests that additional data would likely improve the model's ability to capture power variations.

For the test data, the model was evaluated on previously unseen inputs, yielding prediction scores slightly lower than those for the training data. This small drop was expected and, together with the absolute error results, indicates that the model does not suffer from overfitting or underfitting. Overall, power proved to be more difficult to predict than force, but the model achieved good performance on both metrics.

A regression plot was then created to compare the actual values with the predicted values, for both the training and test data, as shown in Figure 4.8.

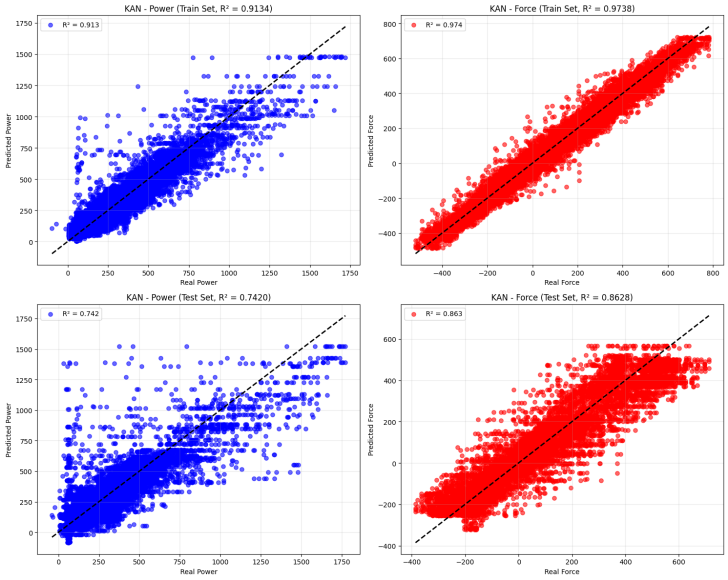

Figure 4.8: Kolmogorov–Arnold network – regression plots

When comparing the regression plots obtained from the training data with those from the test data, it becomes evident that the model performed well during training, with R^2 values above 90 % for both power and force, despite clear signs of outliers in the power predictions. However, the same behavior was not observed in the test data, where the model showed greater difficulty generalizing to unseen samples. For force, an $R^2 = 0.8628$ was obtained, which is not a poor result. Nevertheless, the more concerning case is power, with an $R^2 = 0.7420$, indicating that power prediction has greater difficulty generalizing and suggesting possible signs of overfitting.

To further analyze the model's prediction capability, additional complementary plots were created. In these plots, the model was given a time series used during training, and the same procedure was applied to a time series used exclusively for testing. These graphs allow the visualization of the original force and power values alongside the model-predicted values in their natural temporal order.

Figure 4.9 shows the plot containing both the original and predicted values for a time series used in training. In this example, the time series corresponds to a test performed with an amplitude of 50° combined with a frequency of 0.5 Hz.

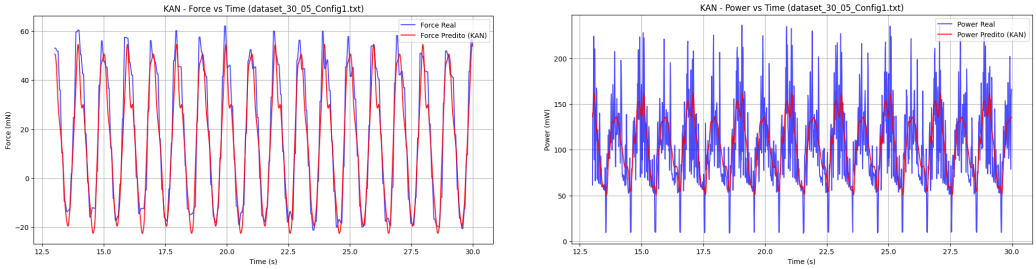

Figure 4.9: Original force and power (in blue) and neural network predictions (in red) for $A_\theta = 30°$ and $f = 0.5\,\text{Hz}$

These plots confirm what was already observed in the regression graphs: the model shows a better ability to predict force compared to power. In the power plot, the difficulty in predicting the power signal becomes evident, especially at lower frequencies, where greater instability is also noticeable in the data itself.

For the test set, the time series with an amplitude of 70° combined with a frequency of 1.0 Hz was used. Figure 4.10 shows the resulting predictions.

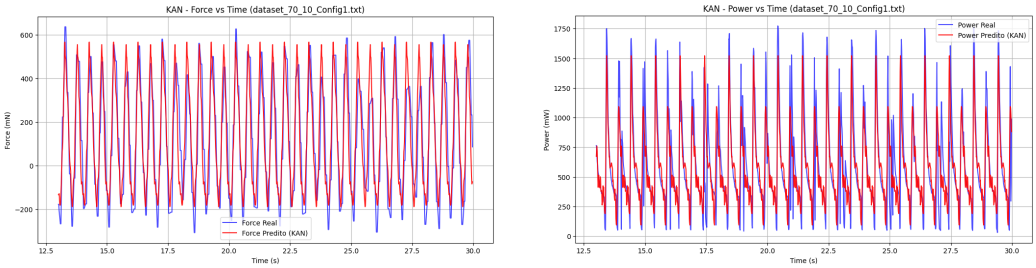

Figure 4.10: Original force and power (in blue) and neural network predictions (in red) for $A_\theta = 70°$ and $f = 1.0\,\text{Hz}$

In this plot, it can be seen that, when fed with an unseen time series, the model struggled to capture how force and power varied, particularly due to the limited amount of training data at higher power levels. Although the model was unable to accurately reproduce the periodic behavior and detailed variations of the force signal, it was generally capable of predicting the force generated by the imposed amplitude and frequency, which is a positive indication.

From these results, two conclusions can be drawn. First, the model requires more data to better capture overall power variations. The absence of training examples at high amplitudes and frequencies around 1 Hz left the model unable to predict these conditions reliably. This limitation was already suggested in the scatterplot analysis, where the power distribution showed a skewed right-hand tail. Second, the model does not yet fully capture small, rapid variations over short time intervals.

To address this issue, it would be important to conduct additional tests to evaluate whether using a shorter acquisition time per time series and a higher sampling frequency could improve the model's learning capability—potentially introducing more noise but providing richer data. Alternatively, depending on the intended application, reducing the sampling frequency could be considered in order to obtain more stable signals and potentially improve accuracy.

4.3 Comparison with Other Models

This section compares the KAN model with two other common models, specifically Gradient Boosting and Long Short-Term Memory (LSTM). These are two very different approaches. Gradient Boosting is a tree-based ensemble algorithm that combines multiple weak learners into a single, stronger predictive model, enabling it to iteratively correct errors and improve accuracy. LSTM, on the other hand, is a recurrent neural network architecture capable of retaining information over long sequences and leveraging this stored context for future predictions. These models exhibit characteristics that differ both from the KAN model and from each other, which motivated their inclusion in the experimental evaluation.

The objective of this analysis is to compare the learning capability of the different models and identify which one performs best. For each model, a grid search was applied to explore different combinations of hyperparameters, avoiding the need for successive manual adjustments. For all models, without exception, hyperparameter optimization was necessary because the training dataset is still preliminary and smaller than the final dataset. With a reduced dataset, the risk of overfitting is inherently higher. This risk must be mitigated to ensure that the model can generalize and learn effectively. Therefore, to avoid overfitting, regularization techniques were applied, and the model architectures were deliberately kept simple by limiting the number of layers and neurons per layer, allowing the models to generalize and achieve good predictive performance.

Table 4.4 and Table 4.5 present the results obtained from training the three different models. The first reports the performance in predicting force, and the second reports the performance in predicting power.

Table 4.4: R^2 performance of machine learning methods trained with selected best parameters – force results

Method	Train R^2	Test R^2
KAN	0.9738	0.8628
LSTM	0.9680	0.9203
Gradient Boosting	0.9727	0.8460

Table 4.5: R^2 performance of machine learning methods trained with selected best parameters – power results

Method	Train R^2	Test R^2
KAN	0.9134	0.7420
LSTM	0.9062	0.8225
Gradient Boosting	0.9220	0.7857

Upon analyzing the results, it can be observed that the KAN model achieves accuracy within the same range as the other models during training, both for force and power. However, this is no longer as clear when examining the test data. In the case of power, the model with the best predictive

performance was the LSTM, with an $R^2 = 0.9203$, and for force, the LSTM again proved to be the best model, achieving $R^2 = 0.8225$. It became evident that the KAN model had the most difficulty in predicting power. Thanks to its recurrent structure—which allowed the use of a 20-point window while maintaining the number of layers and increasing the number of neurons—the LSTM was able to better capture temporal dependencies in the data, as can be seen in Figure 4.11.

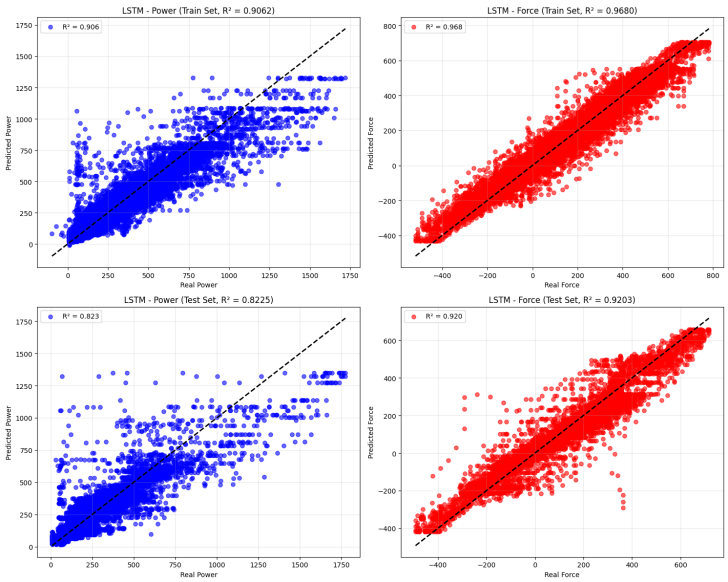

Figure 4.11: Long short-term memory – regression plots

Chapter 5

Conclusions and Future Work

This thesis aimed to develop an experimental rig to study the propulsion in a body-and-caudal-fin locomotion system.

To study the propulsion generated by oscillatory motion, a system was designed to replicate this behavior. The setup measures the forces produced by the oscillatory motion imposed by a servo motor, using a load cell, and records the motor's energy consumption through a current sensor.

This methodology provides a low-cost alternative to Computational Fluid Dynamics (CFD) methods, as it requires less time and computational effort. Nevertheless, the system is not intended to replace CFD but to complement it, supporting the validation of results and enabling faster predictions of the force and energy required to perform a given movement.

In this study, a solution was not only designed to measure thrust force and power consumption during flapping motion but also used to process the acquired data and train a Kolmogorov–Arnold Network (KAN) model.

The dataset obtained from the experimental setup clearly demonstrates system correlation: whenever a power peak occurs, a corresponding peak appears in the force graph.

The analysis identified which combinations of amplitude and frequency produced the highest average system efficiency and included a correlation matrix to explore the relationships among the variables. At first glance, amplitude and frequency appear to be the most important parameters, although θ and $\dot{\theta}$ also play key roles in the system's dynamics.

Subsequently, the KAN model was trained. It achieved an $R^2 = 0.8628$

for force prediction and an $R^2 = 0.7420$ for power prediction. The result for power was rather disappointing, which led to the decision to evaluate two additional models with different characteristics. Among them, the Long Short-Term Memory (LSTM) network yielded significantly better results, reaching an $R^2 = 0.9203$ for force and an $R^2 = 0.8225$ for power.

This study establishes a foundation for several promising research directions, encompassing not only work focused on exploring efficiency through various fin designs, geometries, flexibility, thicknesses, and materials but also serving as a gateway for further comparisons between existing systems.

Furthermore, the methodology could be applied to more advanced systems, potentially incorporating additional sensors and utilizing transfer learning to enhance the learning process and compare results.

In conclusion, although further work remains, this study provides a new method for analyzing, validating, and developing aquatic bio-inspired systems. After CFD simulations, the proposed physical system may be used to compare and validate computational results, helping bridge the gap between simulation and real-world application. As described throughout this work, there are many possible directions for future research that could add value to this thesis and also to the field.

In the case of this system, it would be interesting to conduct the same study with a flexible tail or fin, since one of the limitations presented was that the rigid fin does not allow the use of very high frequencies.

By using a different fin, it might be possible to test other frequencies and potentially identify new efficiency regions, obtaining results with more significant forces that could broaden the range of possible applications. The increase in force itself would also allow the load cell to operate in a more comfortable range, more aligned with its precision capacity.

Another key priority is to conduct comparative studies of traditional propeller propulsion and flapping-based systems. Although some publications report that bio-inspired flapping systems can be significantly more efficient than propellers, such evidence remains limited. More empirical research is needed to confirm and strengthen these claims.

To promote credibility and faster growth in this field, it would also be

crucial to develop more UAVs and USVs, not only increasing the number of projects but also advancing them closer to final stages and conducting more tests in real operating environments. The current scarcity of these systems may create a perception that they are expensive or inherently inferior to conventional propulsion methods. Changing this perception will require more independent studies.

With increasing interest in energy-efficient and green technologies, these bio-inspired systems have strong potential to become more relevant and widely adopted in the near future.

References

[1] PBS NOVA. Einstein quotes. https://www.pbs.org/wgbh/nova/einstein/wisd-nf.html, 2005. Accessed 12 Aug 2025.

[2] Autodesk. Smooth sailing into a greener tomorrow. http://images.autodesk.com/adsk/files/0-foil_customer_story.pdf, 2013. Accessed 12 Aug 2025.

[3] C. C. Lindsey. Form, function, and locomotory habits in fish. In W. S. Hoar and D. J. Randall, editors, *Fish Physiology, Vol. VII: Locomotion*, pages 1–100. Academic Press, New York, 1978.

[4] N. P. B. Mannam. Biological propulsion systems for ships and underwater vehicles. In *Propulsion Systems*. IntechOpen, 2019.

[5] R. Knoller. Die Gesetze des Luftwiderstandes. *Flug- und Motortechnik (Wien)*, 3(21):1–7, 1909.

[6] A. Betz. Ein Beitrag zur Erklärung des Segelfluges. *Zeitschrift fur Flugtechnik und Motorluftschiffahrt*, 3:269–272, 1912.

[7] R. Katzmayer. Effect of periodic changes of angle of attack on behavior of airfoils. *Zeitschrift fur Flugtechnik und Motorluftschiffahrt*, 13:1–10, 1922.

[8] A. I. Nekrasov. Theory of wings in nonstationary flow. *Bulletin of the Academy of Sciences of the USSR*, 1947.

[9] T. von Kármán and J. M. Burgers. *Aerodynamic Theory*. Springer, 1935.

[10] T. Theodorsen. General theory of aerodynamic instability and the mechanism of flutter. Technical Report 496, National Advisory Committee for Aeronautics, 1935.

[11] R. Bale, M. Hao, A. P. S. Bhalla, N. Patel, and N. A. Patankar. Gray's paradox: A fluid mechanical perspective. *Scientific Reports*, 4:5904, 2014.

[12] I. E. Garrick. Propulsion of a flapping and oscillating airfoil. *Journal of the Aeronautical Sciences*, 4:1–10, 1937.

[13] A. Silverstein and U. T. Joyner. Experimental verification of the theory of oscillating airfoils. Technical Report 673, National Advisory Committee for Aeronautics, 1939.

[14] J. B. Bratt. Flow patterns in the wake of an oscillating aerofoil. *Aeronautical Research Council Reports & Memoranda*, 1950.

[15] K. Streitlien, G. S. Triantafyllou, and M. S. Triantafyllou. Efficient foil propulsion through vortex control. *AIAA Journal*, 34(11), 1996.

[16] R. Fernández Prats. *Hydrodynamics of Pitching Foils: Flexibility and Ground Effects*. PhD Dissertation, Universitat Rovira i Virgili, 2015.

[17] K.D. Jones, C.M. Dohring, and M.F. Platzer. Experimental and computational investigation of the Knoller–Betz effect. *AIAA Journal*, 36(7), 1998.

[18] S. W. Tolkoff. Robotics and Power Measurements of the RoboTuna. MSc Thesis, MIT, 1999.

[19] I. H. Tuncer and M. F. Platzer. Computational study of flapping airfoil aerodynamics. *Journal of Aircraft*, 37(3), 2000.

[20] G. K. Taylor, R. L. Nudds, and A. L. R. Thomas. Flying and swimming animals cruise at a Strouhal number tuned for high power efficiency. *Nature*, 425:707–711, 2003.

[21] L. Guglielmini and P. Blondeaux. Propulsive efficiency of oscillating foils. *European Journal of Mechanics – B/Fluids*, 23(2):255–278, 2004.

[22] I. H. Tuncer and M. Kaya. Optimization of flapping airfoils for maximum thrust and propulsive efficiency. *AIAA Journal*, 43(11), 2005.

[23] C. J. Esposito, J. L. Tangorra, B. E. Flammang, and G. V. Lauder. A robotic fish caudal fin: effects of stiffness and motor program on locomotor performance. *Journal of Experimental Biology*, 215(1):56–67, 2012.

[24] E. A. R. Camacho. Numerical Analysis of a Plunging NACA0012 Airfoil. MSc Thesis, University of Beira Interior, 2019.

[25] SeaTech 2020. History of biomimetic flapping foil wave augmented ship propulsion. `https://seatech2020.eu/history/`. Accessed 12 July 2025.

[26] Autonaut USV. Fuel free crew free data rich. `https://www.autonautusv.com/vessels-0`. Accessed 12 July 2025.

[27] Liquid Robotics. Converting wave motion into propulsion. `https://www.liquid-robotics.com/wave-glider/how-it-works/`. Accessed 12 July 2025.

[28] M. S. Triantafyllou and G. S. Triantafyllou. An efficient swimming machine. *Scientific American*, 272(3):64–70, 1995.

[29] D. S. Barret. The Design of a Flexible Hull Undersea Vehicle Propelled by an Oscillating Foil. MSc Thesis, MIT, 1994.

[30] M. Kumph. Maneuvering of a Robotic Pike. MSc Thesis, MIT, 2000.

[31] E. A. Thomson. Robotuna is first of new 'genetic' line. MIT News. Accessed: 2025-02-2.

[32] MIT. Proteus, the penguin boat. `https://techtv.mit.edu/videos/650-proteus-the-penguinboat`, 2018. Accessed 9 December 2025.

[33] A. Conner-Simons. Soft robotic fish swims alongside real ones in coral reefs. `https://news.mit.edu/2018/soft-robotic-fish-`

swims-alongside-real-ones-coral-reefs-0321, 2018. Accessed 9 December 2024.

[34] K. K. Robert, D. Joseph, M. Robert, and Daniela R. Exploration of underwater life with an acoustically controlled soft robotic fish. *Science Robotics*, 3(16):eaar3449, 2018.

[35] F. Berlinger, M. Gauci, and R. Nagpal. Implicit coordination for 3D underwater collective behaviors in a fish-inspired robot swarm. *Science Robotics*, 6(50):eabd8668, 2021.

[36] Leah Burrows. Robotic swarm swims like a school of fish. https://seas.harvard.edu/news/2021/01/robotic-swarm-swims-school-fish, 2021. Accessed 9 December 2024.

[37] S. Verma, G. Novati, and P. Koumoutsakos. Efficient collective swimming by harnessing vortices through deep reinforcement learning. *Proceedings of the National Academy of Sciences*, 2018.

[38] RoboSea. Robo-shark. http://www.robosea.org/sanguanjie.html. Accessed 9 December 2024.

[39] S. Ramachandran. Robo-fish capable of filtering micro plastics in waterways makes waves. https://digpu.com/science/robo-fish-filters-micro-plastics, 2022. Accessed 9 December 2024.

[40] W. S. McCulloch and W. Pitts. A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematica Biology*, 52(1/2):99–115, 1943.

[41] A. M. Turing. Computing machinery and intelligence. *Mind, New Series*, 59(236):433–460, 2008.

[42] J. McCarthy, M. L. Minsky, N. Rochester, and C. E. Shannon. A proposal for the Dartmouth summer research project on artificial intelligence, August 31, 1955. *AI Magazine*, 27(4):12–14, 2006.

[43] J. R. Slagle. *A Heuristic Program that Solves Symbolic Integration Problems in Freshman Calculus, Symbolic Automatic Integrator (SAINT).* PhD Thesis, Massachusetts Institute of Technology, 1961.

[44] J. Weizenbaum. ELIZA—A computer program for the study of natural language communication between man and machine. *Communications of the ACM*, 9(1):36–45, 1966.

[45] J. V. Tu. Advantages and disadvantages of using artificial neural networks versus logistic regression for predicting medical outcomes. *Journal of Clinical Epidemiology*, 49(11):1225–1231, 1996.

[46] M. Kosinski. What is black box artificial intelligence AI? IBM Think. Accessed: 2025-02-2.

[47] T. Mucci. Overfitting vs. underfitting: Finding the balance. IBM Think. Accessed: 2025-02-2.

[48] H. H. Sawant, S. P. Komble, R. Gujar, N. Mandhare, M. J. Sable, P. K. Ambadekar, and S. H. Gawande. Reinforcement learning-driven airfoil optimization: Enhancing aerodynamic performance. *Mathematical Modelling of Engineering Problems*, 12(3):851–859, 2025.

[49] M. T. Akram and M.-H. Kim. Aerodynamic shape optimization of NREL S809 airfoil for wind turbine blades using Reynolds-averaged Navier Stokes model—part ii. *Applied Sciences*, 11(5):2211, 2021.

[50] S. C. van den Berg. Design of a High Speed Soft Robotic Fish. MSc Thesis, TU Delft, 2019.

[51] AirfoilTools. Eppler e836 hydrofoil airfoil. Accessed 25 October 2024.

[52] O. Kallhovd. HX711_ADC. `https://github.com/olkal/HX711_ADC`, 2017. Accessed 24 Aug 2025.

[53] J. De Cristofaro. ArduinoINA219. `https://github.com/johngineer/ArduinoINA219`, 2012. Accessed 24 July 2025.

[54] MathWorks. How to build custom sensor blocks for Arduino in Simulink. `https://www.mathworks.com/videos/how-to-build-`

`custom-sensor-blocks-for-arduino-in-simulink-1704432957396.html`, 2024. Accessed 24 Aug 2025.

[55] A. N. Kolmogorov. On the representation of continuous functions of several variables as superpositions of continuous functions of a smaller number of variables. *Doklady Akademii Nauk*, 108:179–182, 1956.

[56] A. N. Kolmogorov. On the representation of continuous functions of many variables by superposition of continuous functions of one variable and addition. *Doklady Akademii Nauk*, 114:953–956, 1957.

[57] D. C. Liu and J. Nocedal. On the limited memory BFGS method for large scale optimization. *Mathematical Programming*, 45(1–3):503–528, 1989.