POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Executive Summary of the Thesis

Design and Experiments of a Bioinspired Aquatic Snake Robot

Laurea Magistrale in Mechanical Engineering - Ingegneria Meccanica

Author: Luca Lanzetti, Daniele Mariana

Advisor: Giovanni Bianchi

Co-advisor: Simone Cinquemani

Academic year: 2022-2023

1. Introduction

The underwater realm, presents a unique set of challenges for traditional robotic platforms, such as the capability of navigating through confined spaces, negotiating complex terrain, and executing precise maneuvers. Taking inspiration from serpentine locomotion of aquatic species and applying it to robotic systems is a promising solution to these challenges.
The simplest strategy to reproduce and efficiently control the sinusoidal movement of a snake is to segment it into sub-sequential, equally lengthed joint modules.
The goal of this work is to optimize the planar swimming performances of the existent aquatic snake robot previously built in PoliMi [6] and, if necessary, use it as an inspiration to create a new improved version able to be extensively tested on the field and capable of autonomously avoid obstacles.

2. Physical Principles of Aquatic Snake Locomotion

The swimming pattern of the snake is known as Anguilliform [2], a swimming gait performed by bending the body into backward moving propulsive waves along the body.

The main swimming pattern of the sea snake observed in planar swimming is known as lateral undulation swimming. It is basically a traveling sine wave along its body as shown in Figure 1.

Figure 1: Lateral undulation performed by a sea snake [4]

Another marine animal that performs an Anguilliform swimming gait is the eel with the difference that the head of eel has relatively lower oscillation than the undulatory swimming. Figure 2 shows an eel performing its swimming gait.

Figure 2: Eel-like swimming performed by an Eel [7]

3. First Version of the Robot

The first thesis' objective was to optimize the performances of the snake robot previously built in PoliMi within the frame of a master's thesis [6]. Its structure is composed of a head, followed by eight modules, enclosed in a waterproof polyethylene cover.

Figure 3: First version of the robot tested on the field.

From the experimental tests performed, many limitations emerged. The polyethilene cover's rigidity causes a relative motion between the moving joints inside and the almost still surrounding, preventing an efficient motion transmission. Moreover, water was able to pass through non-perfect closures.
The servomotor is directly screwed onto a thin 3D printed frame positioned inside the module, while plastic servo arms are screwed onto a thin plastic projection that transmit the torques together with a M3 screw. After few tests, many frames and servo arms were found damaged and deeply consumed, causing the target loss.
Many ballasts are added to increase its low sinking level caused by air contained inside the cover, dramatically increasing of the inertia each servomotor has to carry on.
The inadequate camera performances, together with the Bluetooth data losses and the non-waterproof ultrasonic sensor forced a head redesign.
Ultimately, to switch-on the robot nine consecutive triggers hided inside the opaque cover must be switched, causing severe time losses.

4. Design of the New Version of the Robot

4.1. The Modules

The new improved version of the robot features an entirely 3D-printed structure, with an external surface cover made of a chemical mixture of ABS and acetone in order to achieve impermeability.
To maximize the force applied to the fluid, the distance between two consecutive modules is reduced to 1 mm.
An aluminum servo arm was introduced. The motion transmission is achieved by two different shape coupling, one between the servomotor's stator and the second between the aluminum servo arm and the principal frame of the previous module. To ensure rigidity, the housing of the following aluminum servo arm and the current servomotor's stator are coupled to the same frame. To improve the robot's control, an external magnetic encoder is added.
The module is divided into four parts, 3D-printed and connected together with the usage of M3 screws, brass threaded inserts and, to avoid any possible water leakage, watertightness dedicated accommodation with dielectric gel poured inside.

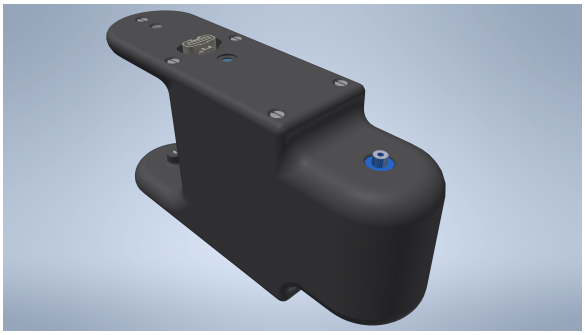

Figure 4: The module of the aquatic snake robot

4.2. Buoyancy

Once decided to completely 3D-print the robot, the level of buoyancy was calculated as a function of the print density. By properly choosing this parameter, the equilibrium between hydrostatic force and the weight force of the robot can be imposed, making the robot float slightly below the water's surface. With a print density of 20% this task was achieved.

4.3. The Head

In re-designing the head, the video camera was removed due to its inadequate output and a different, waterproof, ultrasonic sensor model was chosen.
A micro-SD reader component is added to minimize the risk of data losses. One single, global,

power button was introduced in the head, together with a single connector for battery charging and an IMU. The shape was created by using the Autodesk Inventor FreeForm feature.
The head is split, 3D-printed and connected using the same technique explained for the modules.

Figure 5: The head of the aquatic snake robot

4.4. The circuit

New circuit tries to overcome limitations that emerged from the previous version. A single switch has been realized with the use of low voltage relay, while a high-voltage relay was used to isolate electronic component from batteries when they are charging. Ultrasonic sensor was replaced with a different version that allow to increase the minimum distance detected. In order to prevent data loss during communication an internal SD-reader module was integrated in the head.

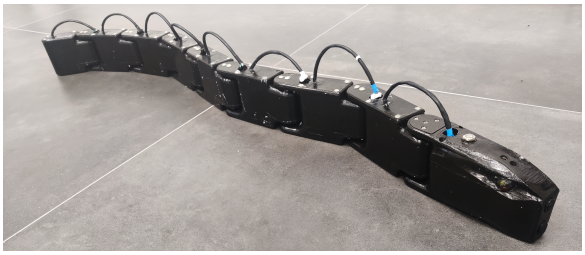

Figure 6: The robot assembled

5. Simulation

5.1. Introduction

The first objective of this thesis was to optimize the first version of the robot, understanding its limitations and highlighting which parameters would maximize its swimming performances. Those needs led to the introduction of a simulation able to reproduce the structure characteristics and inertial properties of the robot as well as rapidly testing the huge amount of motion law's parameters combination.
The simulation is created inside Unity taking advantage of its built-in 3D physics engine, through the usage of the *C#* programming language adopting the `MonoBehaviour.FixedUpdate()` function. The structure of the snake robot is created with the use of the Articulation Body class, able to build hierarchically organized physics articulations and apply the inverse dynamics. The Moving Least Squared-Material Point Method algorithm used to simulate the water behaviour is introduced inside the simulation with the usage of the Zibra Liquids plug-in [1, 3, 5].
It must be stressed out that the choice of this algorithm is forced by the strict time limitations: test a sample of parameters combination with a more precise CFD approach would be prohibitively time consuming. The less accurate output of this approach is compensated by the possibility of simulating a great number of different motion laws to visual inspect the robot behaviour, evaluate a comprehensive global speed trend among the different combinations and the order of magnitude of the torques involved.
A virtual tub of $3 \times 1.5 \times 0.25m^3$ is created through the usage of the Zibra Liquids plug-in. The grid resolution is set to 512 and about $5{\times}10^6$ particles of 5 mm are simulated in real time with the fixed time step value $\Delta t = 0.02s$. The dimensions of the tub, the grid and the particles are limited by the computational effort of simulating the fluid in real time and, according to the Zibra Liquids Support Team, there is no possibility of performing it in non-real time.
A liquid emitter is added to the container as well as the Articulation Body. Then, the mesh of the snake is sent to the Zibra Liquids server which generates the collider shape that interacts with the fluid particles.
A first 60-minute simulation is performed once with the Articulation Body locked in its position and saved, so, each simulation can start from this "Baked Liquid" configuration, where the fluid is at rest.
The Articulation Body is created from the CAD version of the snake robot, reoriented considering the different reference system of Unity. Its mesh is generated inside FreeCaD with the us-

age of the Mephisto algorithm and then scaled down using Blender. The measured inertial parameters of each component, together with the physical limitations of each of the links and the maximum available torque for each drive are introduced to the simulation.

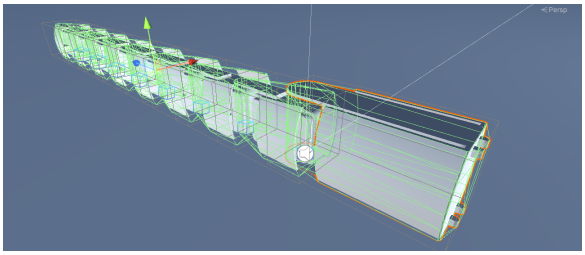

Figure 7: First robot articulation body inside Unity.

In absence of a detailed servo motor's characteristic curve, the torque-speed dependency is assumed as linear.
Each articulation drive is able to apply a torque to reach a user-specified target. Two different $C\#$ codes were created, one to implement the eel-like and the lateral-undulation motion laws. The user is able to set amplitude A, frequency f, and phase shift φ for each motion law before the simulation starts, while the simulation's duration is set to 10s. The function `ArticulationBody.SetDriveTargets()`, calculates for every $\Delta t = 0.02$ s, for each drive, the angular position to aim to.
The Eel-like motion target angles for each drive are coded as:

$$\theta_i = (1 - e^{-\lambda k \Delta t}) A \frac{i-1}{N+1} \sin(\omega k \Delta t - i\varphi), \quad (1)$$

with $i = 1, ..., 8$.
The Lateral-Undulation motion target angles for each drive are coded as:

$$\theta_i = (1 - e^{-\lambda k \Delta t}) A \sin(\omega k \Delta t - i\varphi), \quad (2)$$

with $i = 1, ..., 8$.

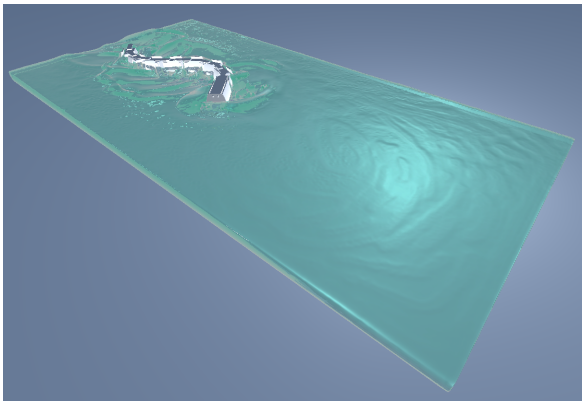

Figure 8: First robot performing lateral undulation motion.

The results will refer to the lateral undulation motion law, since highlights better the involved phenomena.

5.2. Torques Estimation

The inverse dynamics calculation is implemented inside Unity with the usage of the command `ArticulationBody.GetDriveTorques()`.
According to the data extrapolated from the tests and shown in Figure 9, the first' robot servomotors were not able to provide the needed torque.

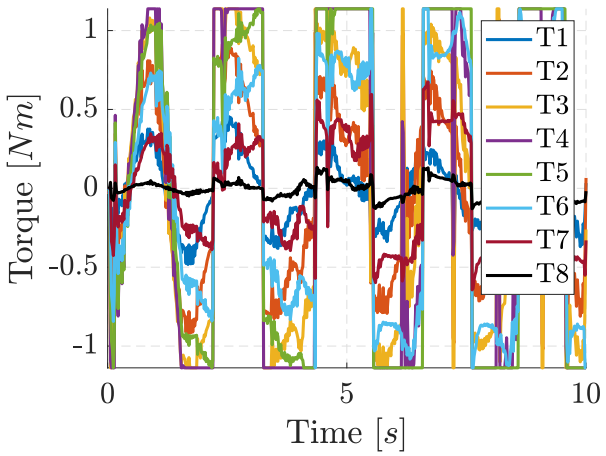

Figure 9: Inverse dynamics with fluid with $A = 30, f = 0.5Hz, \varphi = 0.6rad$

5.3. Velocity Trend Estimation

The new robot design with the correspondent inertial parameters and torques limitations was introduced inside the simulation.

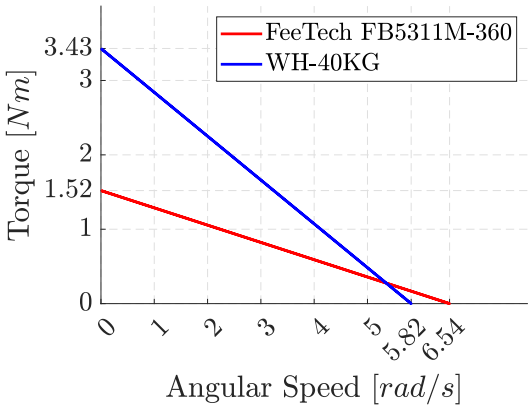

Figure 10: FeeTech FB5311M-360 and WH-40KG characteristic curves

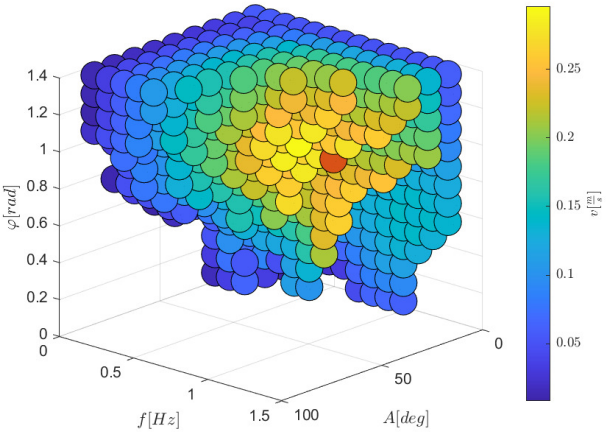

Figure 12: Simulation's result

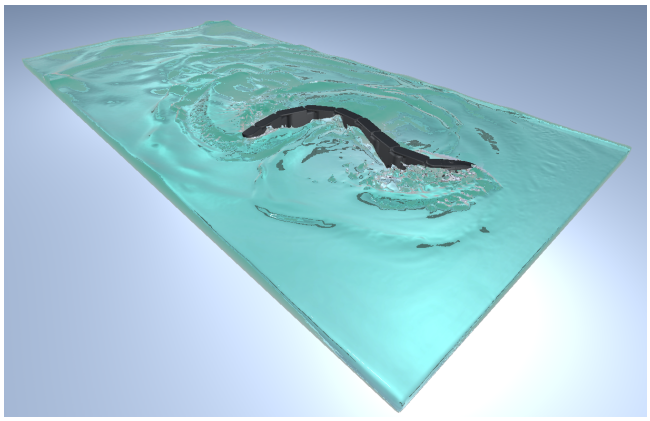

Figure 11: Second robot performing lateral undulation motion

The code of the simulation is slightly edited in order to self-trigger Unity's play mode and test all the possible combinations of A, f and φ. The range of parameters was chosen according to data shown in Table 1, for a total of 1764 simulations.

	Minimum	Maximum	Unit
A	10	90	deg
f	0,1	1,4	Hz
φ	0,1	1,4	rad

Table 1: List of parameters for simulation's test

The final value of the speed was stored as well as all the possible flags that would make the test null, such as the robot going backwards, self-colliding, saturating torque or non reasonable speed values in degenerated simulations.

As emerged from the graph, not all the parameters combinations are feasible: some of them would cause the aquatic snake robot self-collide while others triads would need a torque level that the servo motor is not able to provide in that conditions. After having filtered out all the unfeasible combinations, a speed trend emerges, highlighting an optimal yellow zone characterized by a red dot representing the maximum achievable speed of 0.2955 m/s.

6. Control

This chapter provides a comprehensive overview of the control strategy, low-level and high-level control systems, serpentine curve generation, and methods for steering and obstacle avoidance in the robot. The control strategy is subdivided into two main components: feedback on motor control and high-level control on motion law generation.

6.1. Low-level control system

The low-level control system has two main tasks:

- Reaching the target angles,
- Evaluating the target angle required based on the motion law.

Target angles are reached using two integrated PID controllers within each module. Servomotors exhibit high precision in reaching target angles, primarily due to their built-in controllers. However, in order to eliminate reliance on unknown components, an encoder is used to provide real-angle position information. The presence of encoders is also crucial for developing future control strategies. Different strategies are

considered for generating the serpentine curve. Lateral undulation is known for its high speed, while eel-like motion offers good forward speed with smaller amplitude oscillations, making it more efficient. Lateral undulation is achieved by creating continuous body waves propagated from head to tail. A modification involving an exponential function is applied to limit torque during the initial movement phase.

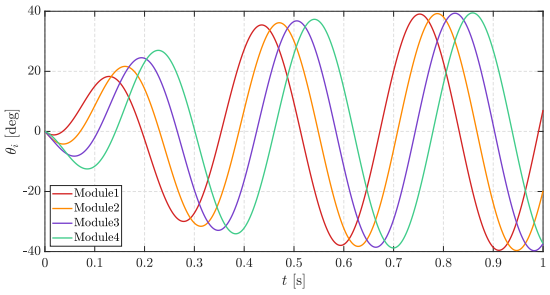

Figure 13: Reference signal for lateral undulation

Eel-like motion is another strategy that includes a term based on module number to vary the amplitude along the robot's body. This variation allows the tail to perform larger movements while keeping the head's rotation small. Two different strategies were used for the steering operation:

- Constant offset method: A phase offset is added to the signal to induce left or right turns.
- Constant hold method: The sinusoidal wave is saturated before reaching its maximum value, affecting steering.

6.2. High-Level Control System

To make the robot follow the reference signal for motion, two controllers are implemented: one at the high level and one at the low level. The high-level controller involves feedforward action, receiving signals from an external user device via an Arduino Mega and Bluetooth module. The Arduino Mega processes the target angles and sends them to Arduino NANOs using the I^2C protocol. Due to the absence of GPS or external positioning systems, trajectory planning is limited. Obstacle avoidance is implemented by modifying the offset of the reference signal with the information that is received from the ultrasonic sensor and encoders.

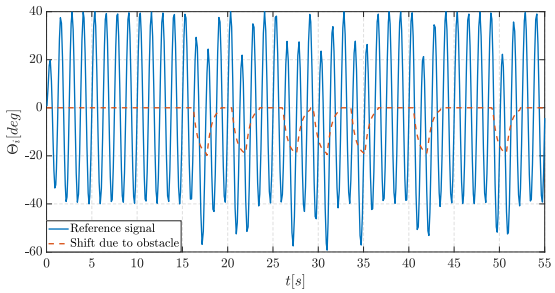

Figure 14: Reference signal for lateral undulation while avoiding obstacles

7. Experimental Tests

In this chapter, results obtained during the first experimental tests are presented. The first objective was to verify the correct execution of the four different tasks: lateral undulation, eel-like, steering and obstacle avoidance.

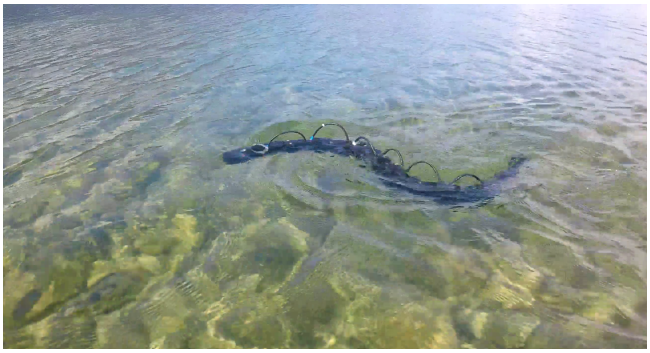

Figure 15: The robot performing lateral undulation during experimental tests

The straight motion was affected by an internal unbalancing of the module, leading the overall structure to shift to the right. This problem was solved by applying a global shift to the overall reference signal. After that, all the two different tasks for straight movement were tested, and for lateral undulation, a velocity of $21\frac{cm}{s}$ was found. Eel-like motion needs to be tested with higher parameters in order to generate a speed comparable to lateral undulation, so all other features were tested with lateral undulation motion only. Steering in constant hold resulted to be more efficient from the pure steering point of view, while constant offset allows the robot to keep higher velocities while performing the steering task.

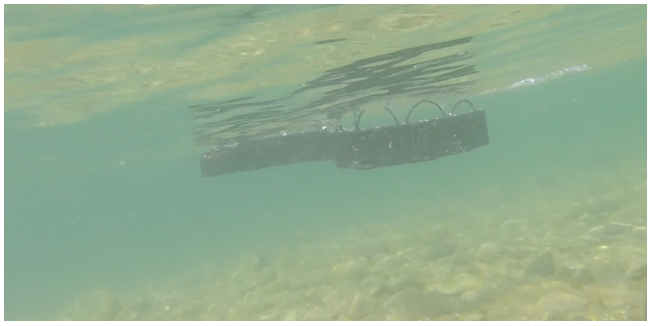

Figure 16: Submerget view of the swimming robot

Obstacle avoidance was adjusted by accounting for waves traveling speed in the water, so modifying the distances that the sensor detects. The robot was then able to correctly overcome obstacles. As a final result, parameters evaluated from the simulation that give the best performance in terms of speed were tested. For parameters shown in Table 2 the robot was able to reach a maximum speed of $34,47\frac{cm}{s}$.

	Value	Unit
A	30	deg
f	$0,9$	Hz
φ	$0,9$	rad

Table 2: List of parameters for finding maximum speed with lateral undulation

8. Conclusions and Future Developments

With the new version of the robot, most of the previous issues were solved, making it capable of correctly executing all the tasks. Moreover, higher velocities were reached, obtaining values up to $34,47\frac{cm}{s}$. The simulation correctly estimates speed and torque, within the limit derived from the approximation. For the future version, thanks to the presence of IMU's feedback and by adding to the robot a compass or GPS, trajectory planning can be developed. Also, a more robust control system using CPG algorithm can be implemented. Moreover, simulation results should be better investigated, as well as building the real characteristic curve of the motor, so that less approximated results can be found.

References

[1] Zibra liquids, 2023.

[2] James K Hopkins, Brent W Spranklin, and Satyandra K Gupta. A survey of snake-inspired robot designs. *Bioinspiration & biomimetics*, 4(2):021001, 2009.

[3] Yuanming Hu, Yu Fang, Ziheng Ge, Ziyin Qu, Yixin Zhu, Andre Pradhana, and Chenfanfu Jiang. A moving least squares material point method with displacement discontinuity and two-way rigid body coupling. *ACM Trans. Graph.*, 37(4), jul 2018.

[4] Daniel Iseli. How do snakes swim? amazing!, 2021.

[5] Chenfanfu Jiang, Craig Schroeder, Joseph Teran, Alexey Stomakhin, and Andrew Selle. The material point method for simulating continuum materials. In *ACM SIGGRAPH 2016 Courses*, SIGGRAPH '16, New York, NY, USA, 2016. Association for Computing Machinery.

[6] Kavinda Pradeep Herath Herath Mudiyanselage. Design, realization and control of an aquatic snake robot. Master's thesis, Politecnico di Milano, 2022.

[7] Oceana. Watch: Mesmerizing ribbon eels swim gracefully | oceana, 2020.