CUKUROVA UNIVERSITY

ENGINEERING FACULTY

Machine Learning-Based Reconstruction of Lost Acoustic Messages in Unmanned Underwater Vehicles

İbrahim GÜRBÜZ - 2021555401

Göktuğ ŞENKAL - 2020555059

Utku Alp TÜREN - 2022555452

Advisor – Prof. Dr. S. Ayşe ÖZEL

Github: https://github.com/UtkuATuren/usvCommsSim

13.06.2025

Abstract

Unmanned underwater vehicles (UUVs) rely on acoustic communication to exchange control commands and telemetry with a surface controller. However, underwater acoustic channels are inherently unreliable due to multipath propagation, frequency-dependent attenuation, ambient noise, and Doppler effects, often resulting in significant packet loss. This unreliability poses a critical challenge in semi-autonomous UUV missions where command continuity is essential but infrastructure-based solutions such as tethers or relay nodes are impractical. Instead of attempting to eliminate packet loss entirely, this thesis proposes a machine learning-based framework to reconstruct the content of lost command packets in real time, enabling the vehicle to operate reliably even under harsh channel conditions. To support this goal, a custom Python-based acoustic simulator was developed to model realistic underwater environments and generate detailed, labeled datasets capturing both communication success and failure cases. These datasets include over 20 contextual features per packet, encompassing physical channel conditions, UUV dynamics, and signal properties. Using this data, multiple deep learning models, namely LSTM, CNN, and Transformer architectures, were trained and evaluated for their ability to infer the most likely command and parameter values of lost packets. The framework demonstrated that, with sufficient contextual awareness, it is possible to predict missing messages with high accuracy, preserving the integrity of UUV missions without physical link enhancements. This thesis contributes a novel software-defined strategy for loss compensation in underwater communication, a modular simulation toolkit for reproducible experiments, and empirical validation of predictive models in reconstructing command content. The approach enhances mission safety, reduces latency due to retransmission cycles, and lays the groundwork for intelligent, loss-tolerant control systems in bandwidth constrained underwater domains.

Keywords: Unmanned Underwater Vehicles, Acoustic Communication, Packet Loss Reconstruction, Machine Learning, Underwater Simulation.

List of Contents

List of Figures

List of Tables

List of Abbreviations

ACKs - Acknowledgments

ARQ - Automatic Repeat Request

AUVs - Autonomous Underwater Vehicles

CNN - Convolutional Neural Networks

ELF - Extra Low Frequency

FEC - Forward Error Correction

ISI - Intersymbol Interference

LSTM - Long Short-Term Memory

ML - Machine Learning

MPC - Model-Predictive Control

RF - Radio Frequency

ROVs - Remotely Operated Vehicles

SNR - Signal to Noise Ratio

UANs - Underwater Acoustic Networks

UUVs - Unmanned Underwater Vehicles

1 Introduction

1.1 Problem Statement

Unmanned underwater vehicles (UUVs) rely on acoustic wireless links to exchange control commands and telemetry with a remote controller, as radio frequency and optical signals are impractical for any significant distance underwater. However, underwater acoustic communication channels are widely regarded as one of the most challenging and unpredictable communication media in use. Factors such as frequency-dependent signal attenuation, multipath propagation due to reflections, high ambient noise, and the low speed of sound (≈1500 m/s) lead to limited bandwidth and substantial latency [1]. These harsh channel conditions often result in high error rates and intermittent connectivity, making reliable real time control of UUVs difficult in practice. In effect, continuous and robust communication remains a bottleneck for underwater vehicle operations, as even moderate ranges or dynamic water conditions can cause significant packet loss and link disruption [2].

In the specific scenario considered, the UUV is not fully autonomous and must receive timely commands from a human operator to carry out its mission. Due to cost and complexity constraints, the system does not employ tethered cables, signal repeaters, or relay buoys to bolster the communication link, the vehicle depends solely on a direct acoustic connection to the controller. This minimalist infrastructure underscores the critical importance of the acoustic link's reliability [3]. When command or telemetry packets are lost over this single-hop link, the vehicle may fail to execute crucial maneuvers or report its status, potentially jeopardizing mission safety. Unfortunately, such message losses are not rare; under certain conditions of distance, interference, or noise, a substantial fraction of transmitted packets may never reach their destination. The inability to predict when these losses will occur exacerbates the risk, as operators currently have little forewarning of a communication drop-out [4].

Addressing this problem requires a shift in focus from simply reacting to lost messages toward estimating their original content. Instead of attempting to predict whether a packet will be lost or not, this research concentrates on reconstructing the likely command and parameter values of messages that have already been lost in the underwater acoustic channel. The core problem can thus be defined as: how can one predict, with useful accuracy,

the most probable command and associated parameter of a message that was not successfully received under current underwater channel conditions? To tackle this, a data-driven approach is proposed. There is a notable gap in the literature regarding proactive estimation of lost underwater command contents, and very limited empirical data are available on packet content reconstruction in UUV acoustic links [5]. This absence of prior models and datasets motivates the development of a novel predictive solution using machine learning techniques to learn the complex relationship between environmental/operational factors and message delivery outcomes.

1.2 Research Objectives

The primary objective of this research is to develop and validate a machine learning based framework that can reconstruct the lost command contents in an underwater acoustic communication system linking a surface controller with a UUV. In contrast to traditional approaches that focus on improving link reliability via hardware enhancements or fail-safe protocols, this study emphasizes a software-driven predictive strategy. By leveraging patterns gleaned from data, the aim is to estimate the most likely command and parameter that were lost, allowing the system to continue its operation with minimal disruption, even when packet loss occurs. The successful outcome of this objective will be a model or set of models capable of inferring the most probable command and parameter values of a lost transmission based on relevant contextual features, effectively compensating for the missing data without altering the physical communication infrastructure.

Achieving this aim entails several specific sub-objectives. First, a realistic simulation environment must be created to generate the data necessary for training and testing the predictive models. Due to the specialized nature of underwater acoustic propagation, existing network simulators (e.g., ns-3 with Aqua-Sim extensions) were assessed but found inadequate for the fine-grained control and customization required in this project [6]. Therefore, the research includes the development of a custom Python-based simulator tailored to UUV communication. This simulator is designed to model key acoustic channel effects (such as range-dependent attenuation, noise variability, and multipath delays) and to emulate the motion and operational behavior of the UUV and controller. By simulating numerous message exchanges under varied environmental conditions and distances, the

simulator produces a rich dataset for analysis. In fact, two separate datasets are generated to capture both communication directions: one for messages sent from the controller to the UUV, and another for messages from the UUV back to the controller. Each dataset is constructed with careful feature engineering to include parameters that might influence communication performance and quality.

Second, using the data gathered from the simulation, the research objective is to train supervised machine learning models that can predict the original content of a lost message given the input features. This involves experimenting with various classification algorithms to determine which can best capture the complex, non-linear interactions inherent in the underwater channel. The models will be trained on one portion of the simulated dataset and then validated on unseen data to assess their generalization performance. Key performance metrics will be used to evaluate how accurately the model can reconstruct both the command type and its corresponding parameter. An additional objective is to compare the performance of different machine learning architectures for the packet content reconstruction task. Specifically, the study evaluates and contrasts the predictive capabilities of recurrent neural networks (RNN, specifically Long Short-Term Memory - LSTM), convolutional neural networks (CNN), and transformer-based models. This comparative analysis aims to determine which model structure best captures the underlying sequence patterns and nonlinear dependencies present in the underwater communication data, and to assess their relative effectiveness in reconstructing lost command and parameter values under varying environmental and operational conditions.

Third, the research aims to interpret the trained model's findings to some extent identifying which features are most indicative of the lost message content thereby providing insights into the environmental and operational conditions that most strongly affect message reconstruction performance. By meeting these objectives, the study will produce not only a predictive tool for message content reconstruction but also a deeper understanding of UUV communication dynamics derived from the machine learning analysis.

1.3 Significance of the Study

Accurate compensation for lost acoustic messages is crucial to maintaining situational awareness and closed-loop control of a UUV when packets inevitably disappear in a harsh underwater channel. Current practice simply discards missing data and waits for the next valid report, introducing latency and degrading control quality, especially for fine-grained maneuvers or safety critical monitoring [2]. The framework proposed here addresses this gap by supplying post-loss compensation estimates: as soon as a transmission is flagged "lost" by a timeout or negative acknowledgement, the trained model infers the most probable content of that packet from recent context and channel conditions. By filling in these gaps, the controller can continue issuing informed commands and logging telemetry without pausing for a costly retransmission cycle. This capability reduces the operational impact of communication interruptions while retaining the cost benefits of a minimalist, single-hop acoustic link.

Beyond immediate mission resilience, the study contributes a principled, data-driven methodology for loss compensation that can be adapted to other low-bandwidth or intermittently connected marine systems. By analyzing the correlation between measured channel parameters (e.g., range-dependent attenuation or ambient noise) and message semantics (e.g., depth or heading updates), the model learns to reconstruct likely packet contents rather than merely interpolating them heuristically. The resulting technique outperforms static estimation rules because it continuously refines its inference as environmental or operational patterns shift [7]. Such adaptive post-loss prediction offers a lightweight alternative to heavyweight forward error correction schemes, which often demand higher bit budgets than practical in narrowband acoustic channels.

Finally, the custom simulator and twin datasets generated for this research supply a unique resource for the broader underwater networking community. They capture both downlink and uplink communication dynamics, each labelled with ground truth loss events and corresponding compensation targets. These artefacts enable reproducible benchmarking of compensation algorithms, fostering deeper investigation into feature relevance and model robustness. Researchers and practitioners can extend the datasets with additional environmental regimes, such as shallow water reverberation or thermocline scattering, to stress test new approaches without costly sea trials. Accordingly, the

study not only advances immediate UUV reliability but also lays groundwork for a new class of intelligent, loss tolerant underwater communication systems.

1.4 Thesis Structure

The first chapter, *Introduction*, defines the core research problem, unreliable acoustic communication between a UUV and a surface controller, and motivates a predictive, software-driven approach to reconstruct lost messages using machine learning. It outlines the study's objectives, significance, and constraints, emphasizing the importance of maintaining operational continuity in underwater environments without relying on physical infrastructure like cables or repeaters.

The second chapter, *Literature Review*, surveys the state of underwater acoustic networks and reviews recent machine learning driven efforts in packet-loss prediction. It also highlights the limitations of traditional reliability mechanisms such as ARQ and FEC, and identifies a gap in current research regarding post-loss message reconstruction, thereby justifying the proposed direction.

The third chapter, *Communication Media in Underwater Environments*, evaluates the physical transmission modalities available for underwater communication, radio frequency, optical, and acoustic, and explains the rationale for selecting acoustic transmission as the only viable option for mid to long range UUV control, considering the trade-offs in range, bandwidth, environmental sensitivity, and deployment complexity.

The fourth chapter, *Packet-Loss Causation in Underwater Acoustic Links*, provides a detailed examination of the environmental and physical factors that degrade underwater acoustic communication. It covers attenuation, multipath effects, Doppler shift, ambient noise, and derives a mathematical model for packet-loss probability, offering a rigorous foundation for the machine learning models used later in the thesis.

The fifth chapter, *System Architecture and Design Choices*, describes the complete architecture of the UUV communication system, including the semi-autonomous control model, evaluation of tethered and relay-based alternatives, and the justification for adopting

a lean, software-centric approach. It articulates the design rationale behind excluding physical infrastructure in favor of predictive compensation techniques.

The sixth chapter, *Simulation Framework*, introduces a custom Python-based simulator developed to replace traditional tools like Aqua-Sim. It details the simulator's architecture, performance advantages, visualization interface, and how it was used to generate rich, labeled datasets under varied acoustic and operational scenarios for training machine learning models.

The seventh chapter, *Data-Set Construction and Pre-Processing*, explains the methodology used to extract downlink communication data, encode commands, bin and normalize parameters, and ensure data integrity. It also discusses the application of dimensionality reduction and data augmentation strategies to improve model generalization and training robustness.

The eighth chapter, *Model Selection and Training Methodology*, outlines the candidate machine learning architectures, including LSTM, CNN, and transformer models, and the pipeline used for hyperparameter tuning and training. It also addresses overfitting mitigation techniques and sets the stage for performance benchmarking.

The ninth chapter, *Experimental Results*, presents the evaluation metrics used and compares the performance of different model architectures on the test datasets. It assesses how accurately each model can reconstruct lost commands and parameters, offering insights into their respective strengths and limitations.

The tenth chapter, *Discussion*, interprets the experimental outcomes in the context of underwater communication challenges. It reflects on the implications of the results, discusses the limitations of the approach, and examines the generalizability of the models to other scenarios or environments.

The eleventh chapter, *Conclusions and Future Work*, summarizes the key contributions of the study, proposes a deployment roadmap for real world implementation, and outlines several directions for future research, such as real time closed-loop control, multi-hop networking, and the application of transfer learning to adapt the models across diverse environmental conditions.

2. Literature Review

2.1 Underwater Acoustic Networks: State of the Art

Underwater acoustic networks (UANs) operate under extremely challenging physical conditions distinct from terrestrial wireless systems. The acoustic channel offers orders of magnitude lower bandwidth than radio and incurs very high propagation delays (on the order of 0.5-1 second per kilometer) [4]. Signals are further degraded by strong attenuation, multipath propagation, and time-varying ambient noise, all of which contribute to a high probability of packet errors and losses in underwater links [8]. Consequently, reliable data delivery underwater is far more difficult than in air, and a significant performance gap remains despite intensive research efforts [9]. These fundamental limitations motivate specialized communication techniques and network architectures tailored to the underwater environment.

Researchers have explored a variety of strategies to improve reliability in UANs. However, directly applying conventional schemes from RF networks reveals inherent inefficiencies underwater. For example, automatic repeat request (ARQ) protocols become problematic under long acoustic propagation delays, since waiting for acknowledgments over such latencies greatly increases end to end transfer times [10]. Forward error correction (FEC) is a common alternative that adds redundancy to correct errors without retransmission, but selecting a fixed code rate is difficult, too little redundancy fails to correct errors, whereas too much wastes the scarce bandwidth, especially given the highly variable channel conditions [11]. Adaptive and hybrid approaches have therefore been proposed, combining FEC with selective ARQ to balance reliability and delay in acoustic channels [11]. In addition, multi-hop network topologies using intermediate relay nodes are often employed to extend range and improve end-to-end success rates, since a single long range acoustic hop is prone to outages. Studies show that multi-hop relaying can significantly boost communication reliability in underwater sensor networks by mitigating the severe attenuation and fading of long links [12].

In practical deployments, two distinct paradigms exist for underwater vehicle communication, each with its own state of the art solutions. On one hand, tethered systems such as

remotely operated vehicles (ROVs) maintain a direct cable link to a surface ship, providing high-throughput, virtually error free communication for real time control and data transfer. This wired approach ensures minimal packet loss but is limited to relatively short ranges and requires the constant presence of a support vessel, making it impractical for deep or wide area missions [13]. On the other hand, fully autonomous underwater vehicles (AUVs) sever the tether and rely on acoustic modems for wireless communication. Acoustic links enable greater range and mobility but only support low data rates and intermittent connectivity, so AUVs must be designed with a high level of onboard autonomy to tolerate long periods of limited or no contact with the operator [7]. Current underwater network architectures often incorporate additional infrastructure like surface gateways or relay buoys to extend coverage, but deploying such repeaters adds complexity and cost. Ensuring reliable command and control of an untethered UUV over a single-hop acoustic link, without intermediate relays or a fully autonomous vehicle logic, thus remains an open challenge in the field [14].

2.2 Machine-Learning-Driven Packet-Loss Prediction Studies

In recent years, machine learning (ML) techniques have been increasingly applied to communication networks to predict and mitigate packet loss. The motivation is that complex, non-linear interactions of factors affecting packet delivery can be learned from data, enabling more adaptive and proactive strategies than traditional fixed algorithms [15]. Prior studies in wireless and sensor networks have demonstrated the feasibility of using ML models to forecast network performance metrics such as packet loss rates or link quality, which can then inform dynamic adjustments to protocols [16], [17]. This data-driven approach is especially attractive in environments like underwater acoustics where accurate analytical modeling of the channel is difficult. Instead of relying solely on simplified theoretical models, ML can capture the subtle effects of time-varying channel conditions and interference by training on empirical data, complementing or surpassing classical models under complex conditions [7].

Several studies have specifically explored ML-based packet loss prediction and network optimization in underwater acoustic contexts. For instance, Kalaiarasu et al. developed a logistic regression model to predict the packet success rate of an acoustic link based on

environmental features such as wind speed, tides, and currents, along with modem specific factors. Their results demonstrated that even relatively simple ML methods could effectively capture spatio-temporal variations in link performance, quantifying underwater link reliability under changing conditions [18]. Furthermore, other researchers have introduced environment aware learning models that utilize inputs like noise levels, distance, or water column properties. These models predict communication channel quality in real time, enabling adaptive network adjustments, such as dynamically altering transmission power or data rates, to minimize packet losses proactively [19].

Beyond simpler models, more advanced machine learning techniques have also been employed to enhance underwater acoustic communication. Deep learning approaches, for example, have been investigated for underwater channel state prediction, leveraging neural networks to recognize complex patterns in signal fluctuations preceding packet errors [20]. Similarly, neural-network-driven adaptation schemes have been proposed to fine-tune network parameters dynamically. One notable study utilized a Bayesian regularized neural network to optimize data rates in sensor networks, significantly reducing packet loss and improving overall underwater link reliability [21]. Collectively, these efforts highlight the growing interest in and potential of sophisticated ML methods for understanding and predicting underwater communication performance.

Most existing machine learning studies in underwater communications predominantly focus on forecasting or proactively preventing packet losses rather than dealing with them after occurrence. Typically, the goal is to adjust communication strategies, such as modulation schemes, coding, routing paths, or transmission timing, to avoid packet drops altogether, rather than reconstructing lost data. For instance, some researchers have trained classifiers to identify periods of congestion or poor channel conditions in advance, allowing data transmissions to be rescheduled or rerouted proactively, thus reducing packet losses [22], [23]. Similarly, other methods utilize predictions of packet success probability to implement adaptive redundancy or retransmissions, further reducing the likelihood of packet loss [24].

However, comparatively little attention has been given to methods that actively address packet losses after they have occurred, particularly in critical applications like underwater

command and control systems or essential sensor networks. In multimedia communications, the idea of packet loss concealment, using deep learning models to reconstruct missing audio frames with minimal latency, is already well-established, enabling real time error compensation [25], [26]. Yet, in underwater acoustic communication, the concept of leveraging machine learning for reconstructing or estimating lost command and control messages or sensor data remains largely unexplored. This thesis directly addresses this gap by applying ML models specifically designed to estimate and compensate for missing messages, aiming to significantly improve reliability and robustness in underwater communication systems.

2.3 Limitations of Current Approaches

Although many solutions exist to improve underwater communications, each approach faces significant limitations in remote control scenarios involving UUVs. Infrastructure-based solutions, such as multi-hop networking or relay nodes, can extend the range and reliability of acoustic links. However, deploying additional relay buoys or support vehicles is often costly, logistically challenging, and impractical for ad-hoc missions involving a single UUV [27]. Traditional reliable transport protocols, like ARQ, also face fundamental challenges. While ARQ schemes perform effectively in terrestrial environments, they become inefficient underwater due to long propagation delays. Waiting for acknowledgments (ACKs) over distances of several kilometers significantly stalls throughput and fails to meet real time control demands [24].

FEC techniques offer an alternative by encoding redundant data to correct errors at the receiver without retransmissions, but they introduce critical trade-offs. In highly variable underwater channels, choosing an optimal FEC code rate is difficult. A fixed setting can either under protect transmissions when conditions worsen beyond the correction capability or over protect during favorable periods, unnecessarily consuming the already limited acoustic bandwidth [24]. Furthermore, the use of extensive redundancy or repeated transmissions directly consumes valuable time and energy, thus reducing the net data throughput available for critical vehicle control and telemetry tasks [28]. Therefore, con-

ventional reliability mechanisms either result in unacceptable delays or significant inefficiencies when directly applied to acoustic UUV links, underscoring the necessity for alternative methods tailored specifically to these challenging conditions.

The emerging ML-based approaches discussed previously have notable limitations that current research aims to overcome. Primarily, these methods are designed to predict and prevent packet loss proactively by optimizing link parameters. While beneficial for improving throughput and avoiding delays, such approaches do not directly address scenarios in which a packet has already been lost. Simply identifying that a loss is imminent provides no solution for reconstructing or recovering the lost information, which is particularly critical in real time control situations [29], [30].

Moreover, practical deployment of these ML methods often faces constraints related to data availability and model generalization. Certain models depend on external or challenging to obtain inputs, such as wind speed and current profiles required by logistic regression models [31]. Acquiring such environmental data in real deployments, especially for instantaneous predictions, may not be feasible. Additionally, models trained under specific environmental conditions might fail to generalize effectively across different underwater contexts, necessitating retraining for distinct scenarios such as transitioning from shallow coastal waters to deep sea environments [32]. Furthermore, the robustness of many ML approaches remains uncertain, as most models have been validated primarily through simulations or limited experimental datasets. Consequently, their effectiveness in the highly complex and unpredictable ocean environment remains unproven. Thus, current ML-driven solutions, although promising, have yet to fully address the reconstruction of lost messages and continue to rely on assumptions that limit their practicality for underwater UUV communications.

A significant practical limitation in developing underwater communication protocols and ML-based predictors lies in their reliance on simulations rather than Real world testing. Because conducting sea trials is expensive and logistically challenging, researchers typically use simulation environments, such as ns-3 with underwater acoustic extensions like Aqua-Sim, to emulate acoustic channels and network behaviors before deployment [6]. However, these simulators often emphasize either detailed physical-layer acoustics or

high-level network protocols, rarely capturing both adequately. This limitation can produce biased or overly optimistic performance predictions, especially when simplified models overlook critical interactions between layers [6]. Moreover, until recently, simulator support for integrating machine learning models into network simulations was limited, complicating efforts to evaluate adaptive ML-driven network behaviors effectively in virtual underwater environments [6].

To overcome some of these challenges, this thesis employs a Python-based custom simulator instead of complex network simulators to generate training and validation datasets for ML models. This custom approach intentionally sacrifices certain low-level physical realism for flexibility, allowing rapid iteration and controlled exploration of packet loss events and their mitigation strategies. Nevertheless, findings obtained through simplified simulations require subsequent validation under more realistic ocean conditions. Real world experiments have uncovered factors, such as underwater currents causing physical modem vibrations, that generic simulations often fail to represent accurately yet significantly impact communication performance [33]. Thus, while simulation and data-driven approaches are indispensable for initial development, their inherent limitations must be acknowledged.

3. Communication Media in Underwater Environments

3.1 Radio Frequency Transmission

Radio frequency (RF) communication in underwater environments is fundamentally constrained by the medium's electrical properties. Seawater is highly conductive and absorptive to electromagnetic waves, causing RF signals to attenuate drastically even over very short distances [34]. For example, a signal in the megahertz to gigahertz range (such as a typical Wi-Fi frequency) may be reduced to negligible strength within a meter or two of propagation in seawater [35]. In essence, while RF methods excel in air and vacuum, they cannot achieve significant range beneath the water's surface due to the rapid exponential decay of electromagnetic energy in water [36].

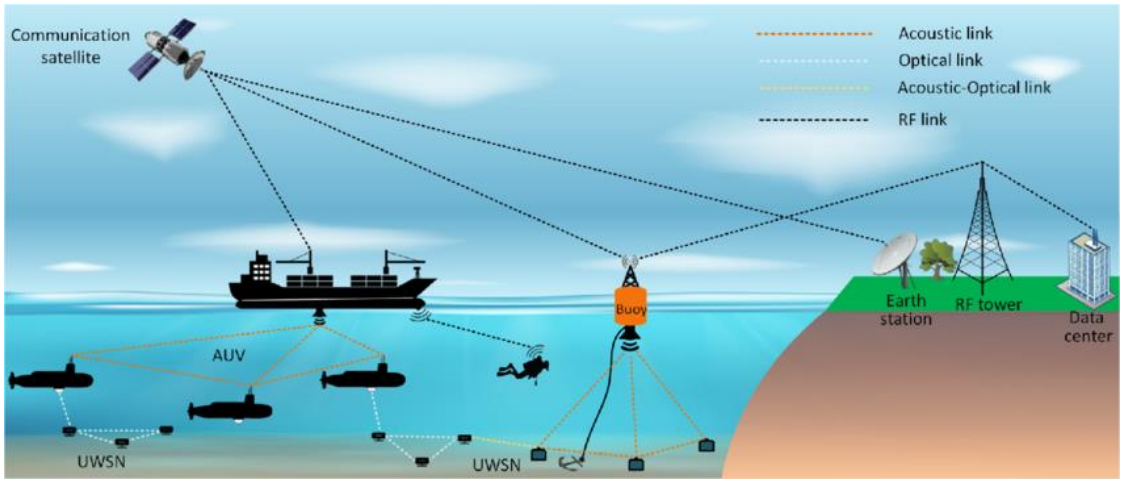

Figure 1 Underwater Wireless Communication Networks [37]

Meaningful underwater RF communication is only feasible at extremely low frequencies where attenuation per unit distance is lower, but this comes at the cost of bandwidth and practicality. Extra Low Frequency (ELF) signals (tens of hertz to a few kilohertz) can penetrate deeper into seawater, which is why they have historically been used for one-way submarine communication by naval forces. These ELF systems require enormous antennas (often spanning kilometers) and offer only very low data rates, illustrating the impracticality of RF for any high-throughput or real time underwater link [38]. For a mobile UUV, towing or incorporating such large antenna systems is not viable, and the bit rates achievable with low-frequency RF would be insufficient for timely control or data exchange [39].

Overall, radio frequency transmission is deemed unsuitable for the UUV controller-to-vehicle communication in this system. While niche scenarios exist, for instance, short range RF links in shallow fresh water or a vehicle briefly surfacing to transmit data, these do not meet the continuous underwater communication needs of an untethered UUV. The severe attenuation, antenna size requirements, and negligible range of conventional RF underwater mean that this modality cannot provide a reliable or practical channel for the system in question [40]. Consequently, RF was ruled out as a communication medium in favor of alternatives more compatible with underwater operation.

3.2 Optical Transmission

Optical transmission refers to using light (typically lasers or LEDs) to carry information through the water, and it offers a very different set of trade-offs compared to RF [41]. Water significantly absorbs and scatters light, though there is a relatively transparent window in the blue-green wavelengths where attenuation is minimized [42]. In ideal conditions (clear water and optimal wavelength), optical signals can propagate on the order of tens of meters before diminishing beyond usability. However, even within this favorable spectrum, underwater optical communication is highly sensitive to environmental factors, it essentially requires a clear, line-of-sight path between transmitter and receiver with minimal turbidity or suspended particles [42].

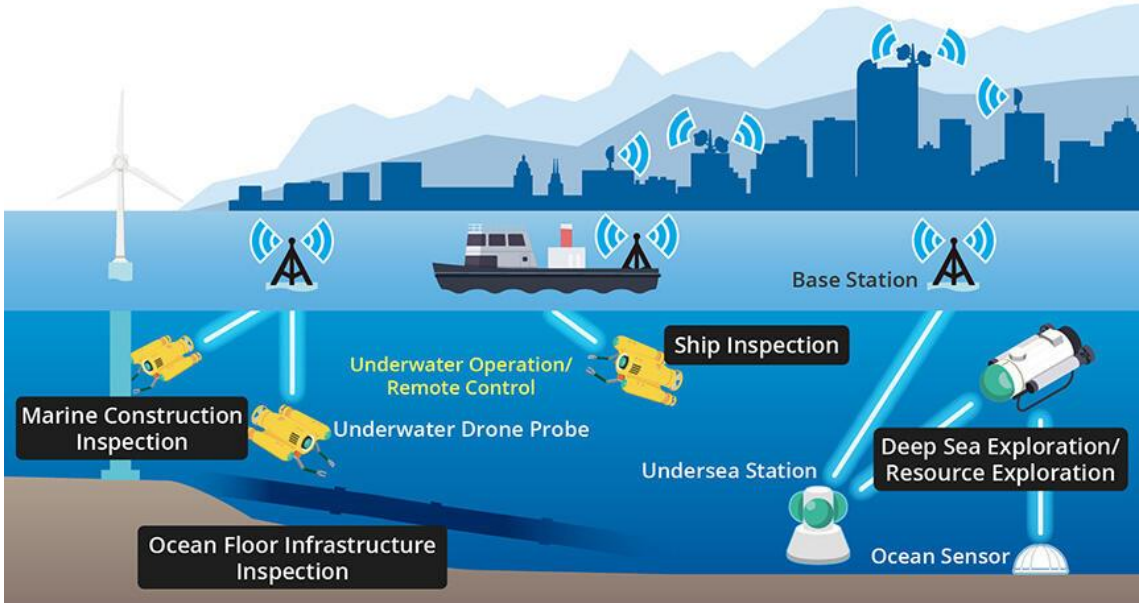

Figure 2 Underwater Wireless Optical Transmission [43]

The chief advantage of underwater optical communication is its potential for high bandwidth and data rates [44]. Using modulated laser beams or high-power LEDs, experimental systems have demonstrated data throughputs in the megabit to gigabit per second range over short underwater distances. This capacity is orders of magnitude greater than what acoustic systems can typically achieve [45]. In practice, though, such high-speed optical links only work over relatively short ranges (often a few meters to a few tens of meters) and under controlled conditions [45]. Any slight misalignment between a narrow optical beam and the receiver can sever the link, and common occurrences like turbidity,

plankton blooms, or even air bubbles can scatter the light and drastically reduce signal quality. Furthermore, underwater optical channels can be disrupted by ambient light (for example, sunlight filtering through water), which introduces background noise for optical receivers [46], [47].

The deployment of an optical communication link for a UUV control system presents significant challenges [48]. Precise alignment and tracking systems might be required to keep a laser beam pointed at a moving UUV, adding complexity and cost to the system [49]. The hardware for high-performance optical links, powerful lasers, collimated beam optics, and sensitive photodetectors, can be power-hungry and expensive, which conflicts with the desire for a low-complexity, energy-efficient setup [50]. Moreover, the reliability of an optical link in the real ocean is questionable; any change in water clarity or orientation could break the connection, potentially cutting off communication when it's most needed [51]. Given these constraints, optical transmission was not selected for the UUV's communication in this thesis. It is generally reserved for niche cases (such as short-range high-speed data offloading or diver to diver communications in clear water) rather than serving as the primary link for long-distance, continuous control of an underwater vehicle.

3.3 Acoustic Transmission

Acoustic transmission uses sound waves to convey information and is the established standard for underwater wireless communication [1]. Sound propagates very effectively in water relative to electromagnetic waves; marine life has long exploited this fact (for instance, whale vocalizations travel for miles underwater), and human technologies have followed suit with sonar and acoustic modems [52]. In comparison to RF and optical signals, acoustic waves experience far less attenuation in water, especially at low and moderate frequencies [44]. As a result, underwater acoustic communications can achieve ranges from hundreds of meters to several kilometers depending on frequency, power, and environmental conditions. This ability to cover long distances makes acoustics the only practical choice for most underwater networks and vehicle links, including the UUV communication system considered here [9].

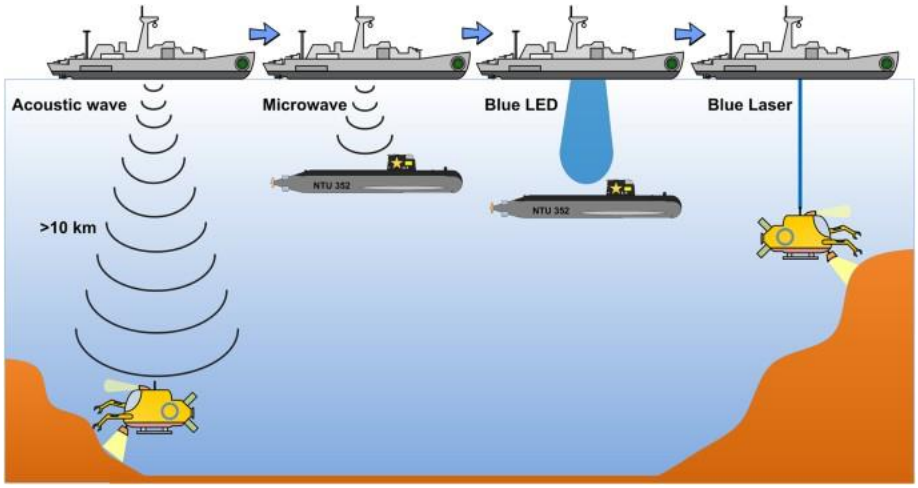

Figure 3 Underwater Acoustic Communication [53]

Despite its range advantages, acoustic communication comes with significant physical and technical limitations. The speed of sound in water is approximately 1500 m/s, much slower than the speed of electromagnetic signals in air, leading to substantial propagation delays (on the order of milliseconds per kilometer) for acoustic transmissions [54]. Moreover, the usable bandwidth of acoustic channels is inherently narrow. Effective underwater acoustic frequencies typically lie in the kilohertz range (from a few kHz up to a few tens of kHz for longer ranges), which corresponds to only a few kilohertz of available channel bandwidth. Consequently, acoustic data rates are modest: many off-the-shelf acoustic modems offer only tens to a few thousands of bits per second, sufficient for low-rate telemetry or command signals but far below terrestrial wireless or wired link capacities [55]. Furthermore, the acoustic channel is fraught with impairments. Sound waves reflect off the sea surface, seafloor, and underwater obstacles, creating multipath interference where multiple delayed copies of a signal arrive at the receiver and interfere with each other. The channel conditions also fluctuate with time as water conditions change and as the platform moves. Combined with ambient underwater noise (from waves, rain, marine life, and ship activity), these factors cause variable signal quality and can lead to high packet error rates in acoustic communication [56], [57].

Even with these challenges, acoustic signaling was chosen as the medium for the UUV's communication link in this thesis due to the lack of any viable alternative for long range, through-water connectivity. Acoustic modems are readily available and can be mounted on both the UUV and the controller (e.g., a surface station), enabling a wireless link where

RF or optical methods would fail to maintain continuity [52]. The inherent unreliability of the acoustic channel (such as occasional message loss) is acknowledged and accepted as a trade-off. By leveraging acoustic communication's strength in reach, and compensating for its weaknesses through techniques like machine learning based packet loss prediction, the system achieves the required reliability without the need for cables, repeaters, or other high-cost components [19].

3.4 Comparative Assessment of Modalities

Underwater communication technologies, RF, optical, and acoustic, exhibit distinct performance profiles shaped by the unique characteristics of the underwater medium [58]. Each modality has specific strengths and weaknesses in terms of range, bandwidth, reliability, and deployment complexity. While RF systems offer high data rates in terrestrial contexts, their performance degrades rapidly underwater due to severe attenuation [59]. Optical communication can support extremely high data rates but is highly sensitive to environmental conditions and alignment [60]. Acoustic communication, though slower in data throughput, emerges as the most reliable and practical choice for most underwater applications due to its extended range and robustness in diverse environments [61].

To complement the qualitative assessment, a numerically grounded comparison is presented in the following table. The values are drawn from established literature and represent typical performance characteristics observed under standard conditions:

Feature	***RF Communication***	***Optical Communication***	***Acoustic Communication***
Range	1–2 meters	10–100 meters	500–5000 meters
Data Rate	Up to 10 Mbps (at <1 m)	10 Mbps – 1 Gbps	100 bps – 100 kbps
Latency	<1 ms (short distances)	<1 ms (ideal conditions)	500 ms/km
Environmental Sensitivity	Very high – signal loss in <2 m	Very high – affected by turbidity, misalignment	Moderate, affected by noise, multipath
Line-of-Sight Requirement	Strictly required	Strictly required	Not strictly required, can use reflections

Operational Complexity	Requires large antennas or surface coupling	Needs beam tracking systems	Simple hydrophones, minimal alignment needed
Reliability	Unreliable due to attenuation	Reliable in clear water	Variable, but manageable with error correction

Table 1 Comparison of Underwater Communication Modalities [4], [34], [62], [63]

The data clearly illustrate that acoustic communication, despite its lower bandwidth and higher latency, is the only modality capable of sustaining reliable mid to long range communication under water. RF signals become ineffective beyond very short distances, making them impractical for UUV operations except in near-surface or partially emerged configurations [59]. Optical links, although promising in bandwidth, are highly vulnerable to underwater visibility conditions, misalignment, and the presence of suspended particles, which limits their use to highly controlled scenarios [60]. In contrast, acoustic systems tolerate variable environments, do not strictly require line-of-sight, and can operate over distances spanning several kilometers, making them indispensable for robust underwater connectivity [61].

Considering these quantified differences, the acoustic modality was selected as the communication foundation for the UUV-controller link in this thesis. Its operational resilience and extended range provide the most feasible path for untethered mission execution. Additionally, its drawbacks, such as occasional packet loss, can be effectively mitigated through adaptive machine learning models that enhance reliability and continuity without requiring hardware modifications or costly infrastructure.

4. Packet-Loss Causation in Underwater Acoustic Links

4.1 Environmental Attenuation and Absorption

Temperature: Water temperature is a major factor in acoustic propagation. Variations in temperature change the sound speed, causing acoustic rays to bend (refract) and altering the effective transmission range [64]. Warmer water generally increases sound speed, while sharp temperature gradients (thermoclines) can refract sound away from receivers

or create shadow zones of poor reception [65]. Temperature also affects absorption characteristics; higher temperatures tend to increase molecular absorption of sound at many frequencies, further weakening signals over long distances [66].

Salinity: Differences in salinity (dissolved salt content) alter water density and sound speed. Higher salinity produces a slight increase in sound speed [67], and salinity gradients (e.g. where freshwater and seawater meet) can deflect sound propagation [68]. Sound traveling across a halocline (salinity interface) may bend or scatter due to the density discontinuity [69]. Overall, significant salinity variations can lead to unpredictable propagation paths and signal fading, contributing to packet loss in variable estuarine or coastal environments [70].

Pressure (Depth): Hydrostatic pressure increases with depth, raising the sound speed and shaping the sound speed profile of the ocean [71]. In deep water, higher pressure tends to increase acoustic velocity, so sound rays may refract upward from high-pressure regions. Pressure's main impact is through these gradients: a deep transmitter/receiver may experience a different refractive environment than a shallow one [72]. The combined effect of pressure and temperature profiles often creates complex sound channels or shadow zones that affect reliability of communication links [72].

Depth of the Water Column: The overall water depth (and proximity of boundaries) strongly influences acoustic losses. In shallow water, sound undergoes frequent reflections off the surface and seabed, leading to higher transmission loss, soft seabeds can absorb energy and act as a low-pass filter, attenuating low-frequency components [73], [74], [75]. This multipath-rich shallow environment causes fluctuations and can severely degrade signal clarity. By contrast, deep water allows sound to travel without immediate boundary losses; low-frequency sounds can propagate over much longer ranges in deep ocean conditions (especially via the deep sound channel) with far less attenuation [76]. Thus, shallow deployments generally face higher packet loss due to multipath fading, whereas greater depth can improve range (albeit with other challenges) [77].

Suspended Particles: Turbidity and particulate matter in the water (silt, organic matter, or air bubbles) scatter and absorb acoustic energy. In waters with a high sediment load, suspended particles redirect sound energy in many directions, effectively reducing the

forward signal intensity [78]. Some of this scattered energy returns as backscatter (noise), which can mask the intended signal [79]. In summary, the enhanced scattering of high-frequency sound waves by small particles and air bubbles results in greater transmission loss and a reduced signal-to-noise ratio.

Thermoclines: A thermocline is a layer with a steep temperature gradient, and it can dramatically affect acoustic propagation. When a warm, less dense layer sits above colder water, sound speed drops rapidly with depth in the thermocline [80]. Acoustic waves crossing a pronounced thermocline will refract, often bending downward in a warm-season thermocline, which can trap sound waves below the layer or prevent direct paths to shallow receivers. This phenomenon sometimes creates a "shadow zone" of poor reception beyond the thermocline [81], [82]. On the other hand, certain thermocline structures combined with pressure effects can form natural waveguides (e.g. the deep sound channel) that carry sound efficiently over long distances [83]. In either case, thermocline-induced refraction leads to variability in reception and potential packet loss if the communication path is deflected away from the receiver.

Surface Agitation: Sea surface conditions (waves, swells, and surface turbulence) cause the air-water boundary to become a dynamic, rough reflector. A calm, flat sea surface behaves like a coherent mirror for sound, but a rough, wind-blown surface scatters acoustic energy in many directions [84]. Surface agitation not only produces additional ambient noise (from breaking waves and bubble formation) but also disrupts the reliability of the surface-reflected path. Signals reflecting off moving waves undergo Doppler shifts and amplitude fluctuations, and some energy is lost into the atmosphere or scattered away [85]. Thus, heavy surface agitation can lead to fast fading and packet errors on near-surface acoustic links, as well as raising the noise floor that the receiver must contend with.

4.2 Multipath Propagation and Doppler Effects

Underwater acoustic links frequently encounter multipath propagation, a phenomenon where sound signals reach the receiver via multiple paths, such as direct, surface-reflected, and bottom-reflected trajectories. These different arrival times cause time dispersion, known as delay spread [86]. When the delay spread surpasses the symbol duration,

intersymbol interference (ISI) occurs, where past and present symbols overlap, leading to decoding errors. The impact of ISI is strongly influenced by the geometry of the communication link: vertical paths (from deep sources to surface receivers) typically exhibit minimal delay, while horizontal, near-surface paths suffer from significantly longer delays due to grazing angle reflections [87]. This extended delay spread introduces severe frequency-selective fading, in which certain signal frequencies are reinforced while others are attenuated, complicating equalization and reducing demodulation reliability [88]. Overall, multipath propagation causes amplitude and phase fluctuations and overlapping echoes that significantly increase the packet error rate unless compensated with advanced signal processing techniques [89].

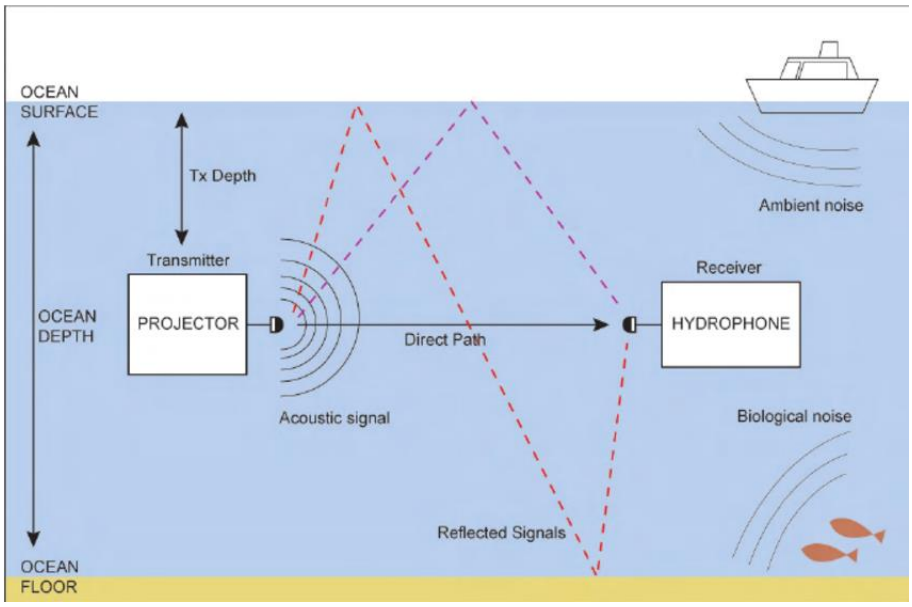

Figure 4 Underwater Acoustic Signal Propagation Paths [90]

Another major challenge in underwater communication is the Doppler effect, which arises from relative motion between the transmitter, receiver, or reflective surfaces such as waves or moving underwater vehicles [91]. Due to the relatively low speed of sound underwater (~1500 m/s), even modest velocities can result in noticeable frequency shifts [92]. When a receiver moves toward the source, the signal appears compressed in frequency; when it moves away, the frequency appears stretched. These shifts can disrupt the synchronization of the receiver's demodulation carrier, leading to symbol timing errors and phase distortion [93]. Furthermore, in a multipath context, each path may undergo a different Doppler shift, resulting in a phenomenon known as Doppler spread. This

creates a time-varying channel where the carrier frequency and phase continuously fluctuate, degrading the signal to noise ratio (SNR) and increasing the bit error rate. Therefore, effective underwater communication systems must be capable of estimating and compensating for Doppler effects in real time to prevent frequent packet losses, especially in mobile scenarios [93].

In addition to their individual effects, multipath propagation and Doppler shifts interact to make the underwater acoustic channel both frequency and time-selective, a condition known as double selectivity. As the transmitter, receiver, or environment changes position during transmission, the structure of the multipath signal also changes, causing the channel characteristics to evolve even within a single packet duration [94]. These rapid variations can exceed the adaptation capabilities of equalization algorithms, resulting in bursty and unpredictable errors. When the channel's coherence time (the time span during which it remains stable) is shorter than the duration of a packet, or when the coherence bandwidth (the frequency range over which the channel response remains flat) is narrower than the signal bandwidth, different parts of the signal may experience inconsistent fading [95]. These conditions lead to irregular packet errors, where some transmissions succeed while others fail catastrophically [96]. The simulation model used in this study incorporates both delay spread (to represent multipath effects) and relative speed (to represent Doppler effects) to reproduce realistic underwater error behaviors. This modeling enables the development of machine learning based strategies for predicting and mitigating packet loss.

4.3 Ambient Noise and Interference Sources

The ocean is filled with background sound from physical environmental processes. Wind-driven wave action is a primary contributor: breaking waves inject broadband noise (from a few hundred hertz up to tens of kilohertz) due to spray and bursting bubbles [97]. Higher sea states (strong winds and rough seas) correspond to higher ambient noise levels across a wide frequency range, raising the acoustic noise floor [97]. Rainfall is another intermittent but significant source, heavy rain can increase noise levels by up to 35 db across a broad range of frequencies (from roughly 1000 Hz to greater than 50,000 Hz) [98]. Other geophysical events such as cracking sea ice, undersea earthquakes, and volcanic eruptions

contribute loud but localized noise [99], [100], [101]. At the upper end of the spectrum, thermal noise dominates (above ~100 kHz): this is the ever-present random molecular noise in the water [102]. These natural noise sources collectively form an ambient background din that limits the lowest achievable receiver noise level. When ambient noise is high, the SNR of the communication link is reduced, making it more likely that packets will be corrupted or drowned out by the noise [102], [103].

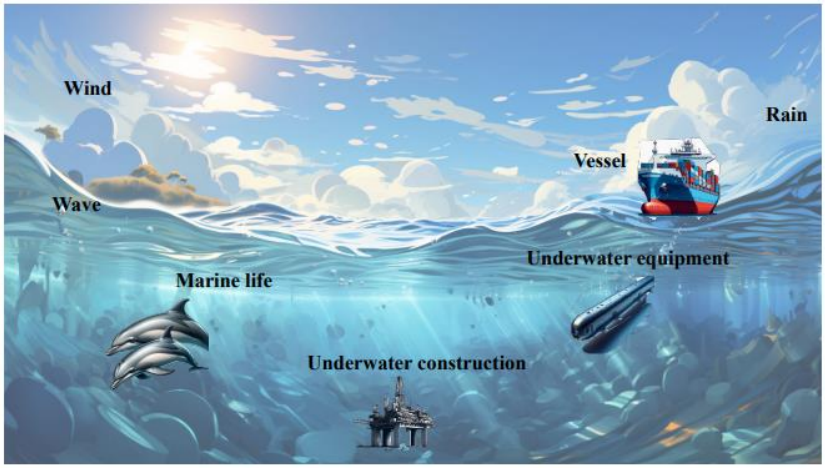

Figure 5 Underwater scenario illustrating complex noise sources [79]

Marine life also generates sound that can interfere with underwater communications. Many marine animals use sound to communicate, navigate, and hunt, and their vocalizations add to the ambient noise, sometimes dramatically [104]. Blue and fin whales produce low-frequency sounds at frequencies of 10-40 Hz with estimated source levels of up to 190 underwater dB at 1 meter. For example, during certain seasons, their vocalizations, particularly in the 10-25 Hz range, have been observed to increase ambient noise levels by 20-25 dB in some ocean regions, significantly influencing the underwater acoustic environment [105]. At higher frequencies, snapping shrimp are notorious noise producers, in coastal tropical waters, colonies of snapping shrimp create a continuous crackling noise with most energy in the 2-5 kHz range. Individual shrimp snaps have source levels up to ~189 dB (re 1 μPa @1 m), and collectively these clicks dominate the background noise in some shallow waters [106]. Dolphins and other odontocetes also emit clicks and whistles in mid to high frequencies [107]. The presence of loud biological noise sources effectively masks communication signals by raising the noise floor. For instance, a swarm of snapping shrimp can severely limit acoustic system performance, their broadband

snaps can mask modem signals and increase packet loss rates if operating in the same band. In short, marine mammal calls, fish choruses, and snapping shrimp snaps are common biological interferers that reduce SNR and reliability for underwater links in certain areas and times.

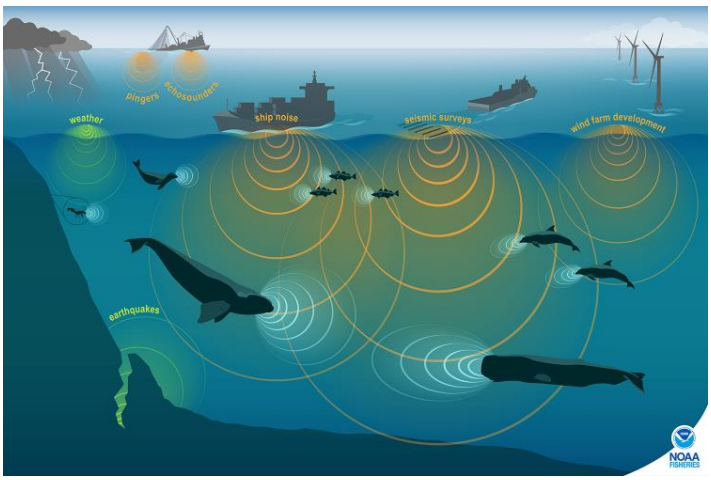

Figure 6 Natural and Anthropogenic Contributors to Underwater Ambient Noise [104]

Human activities have introduced significant noise into the oceans, often termed underwater noise pollution. Chief among these is commercial shipping: distant ship traffic is actually the primary source of low-frequency ambient noise in the 10-500 Hz band worldwide. In coastal zones with heavy ship traffic, the background noise can be 12 dB higher than in quiet regions, which can mask signals that lie in the same frequency band [108]. In addition to this continuous background from distant ships, nearby vessels produce loud engine and propeller noise that can directly interfere with communications. Small workboats, for example, radiate noise in the kHz range that overlaps typical acoustic modem frequencies, causing spikes of interference when they pass [109]. Other man-made noise sources are impulsive and intense: seismic air gun blasts (for oil exploration), pile driving, dredging, and active sonars all create high-amplitude sounds. These sources are often localized in time and space (e.g., a seismic survey in a given region), but when present, they can overwhelm communication signals [110]. For instance, an active naval sonar ping in the 10-30 kHz range will raise the noise floor dramatically for a short period, likely garbling any packets transmitted at that time [111]. All of these anthropogenic noises reduce the SNR by adding interference energy in the communication band [112]. The result is an increased packet loss probability, either through masking (the interference

noise makes the message unrecognizable at the receiver) or through collision (the interfering signal is mistaken as data or corrupts the decoding process). In the context of this thesis, such interference is treated as part of the environmental noise that our system must overcome, emphasizing the need for robust modulation and error-correction schemes.

4.4 Overview of packet loss approach and loss formula

The research in this thesis has led to a unified derivation of the packet-loss probability for underwater acoustic communication, beginning with a standard link-budget framework and culminating in a closed-form expression that incorporates both large-scale path loss and small-scale fading. Let d denote the slant range between transmitter and receiver in meters, f the center frequency in kilohertz, P_0 the source-level acoustic power spectral density at 1 m (in linear units, e.g., μPa^2), and N the noise power spectral density at the receiver (in the same units). The reference SNR at 1 m is defined as $\gamma_0 = P_0/N$. As the acoustic wave propagates through seawater, it incurs a combination of geometric spreading, frequency-dependent absorption, and site-specific anomaly losses, each expressed in decibels. Geometric spreading is modeled by TLspread $= 10\,n\,\log 10\,(d)$, where n is the spreading exponent (unitless), taking values between 1 (cylindrical spreading) and 2 (spherical spreading). Frequency-dependent absorption is captured by Thorp's empirical formula [113], [114], which computes $\alpha_{\text{dB/km}}(f)$ in dB/km as

$$\alpha_{\text{dB/km}}(f) = 0.11\,\frac{f^2}{1+f^2}\,44\,\frac{f^2}{4100+f^2}\,2.75\times 10^{-4}\,f^2\,0.003,$$

valid for $f \geq 0.3$ kHz. Conversion to dB per meter yields $\alpha_{\text{dB/m}}(f) = \alpha_{\text{dB/km}}(f)/1000$. A constant anomaly term A in dB accounts for additional boundary or scattering losses. Summing these contributions produces the total transmission loss in dB:

$$\text{TL}(d,f) = 10\,n\,\log_{10}(d) \;+\; \alpha_{\text{dB/m}}(f)\,d \;+\; A.$$

All instances of d within the logarithm and the absorption product are understood in meters, ensuring unit consistency. A loss of x dB corresponds to a power-ratio attenuation of $10^{(x/10)}$; hence, the linear attenuation factor is

$$L_{\text{lin}}(d,f) \;=\; 10^{[\text{TL}(d,f)/10]},$$

which is dimensionless. Dividing the reference $SNR\gamma_0$ by $L_{\text{lin}}(d,f)$ yields the large-scale mean SNR at distance d:

$$\gamma_{\text{mean}}(d,f) = \frac{\gamma_0}{L_{\text{lin}}(d,f)} = \frac{\gamma_0}{10^{[\,10\,n\,\log_{10}(d)+\alpha_{\text{dB/m}}(f)\,d+A\,]/10}}.$$

Rewriting $10^{[10\,n\,\log_{10}(d)/10]} = d^{\,n}$ and collecting exponent terms provides the alternative form

$$\gamma_{\text{mean}}(d,f) = \frac{\gamma_0}{d^{\,n}\;10^{[\alpha_{\text{dB/m}}(f)\,d/10]}\;10^{[A/10]}}$$

This expression clarifies that geometric spreading contributes a multiplicative factor $d^{\,n}$, absorption contributes $10^{[\alpha_{\text{dB/m}}(f)\,d/10]}$, and anomaly loss contributes $10^{[A/10]}$ within the denominator. The resulting γ_{mean} remains unitless. Real acoustic channels exhibit rapid fluctuations in instantaneous SNR due to multipath interference. Under the assumption of Rayleigh-distributed small-scale fading [115], the instantaneous SNR γ follows an exponential probability density function with mean $\bar{\gamma} = \gamma_{\text{mean}}(d,f)$, namely

$$p_\gamma(\gamma;\,\bar{\gamma}) = \frac{1}{\bar{\gamma}}\,e^{-\frac{\gamma}{\bar{\gamma}}}, \quad \gamma \geq 0$$

Consequently, the cumulative distribution function is:

$$F_\gamma(x;\,\bar{\gamma}) \;=\; 1 \;-\; e^{-\frac{x}{\bar{\gamma}}}$$

A packet is declared lost if the instantaneous SNR falls below a required threshold γ_{req}. Therefore, the outage probability or packet-loss probability is

$$P_{\text{loss}}(d,f) =$$
$$\Pr\{\gamma < \gamma_{\text{req}}\} =$$
$$F_\gamma\left(\gamma_{\text{req}}\,;\,\bar{\gamma}\right) =$$
$$1 \;-\; e^{-\frac{\gamma_{\text{req}}}{\gamma}}$$

Substituting $\bar{\gamma} = \gamma_{\text{mean}}(d,f)$ yields:

$$P_{\text{loss}}(d,f) = 1 - e^{-\frac{\gamma_{\text{req}}}{\gamma_{\text{mean}}(d,f)}}$$

Upon inserting the explicit form of $\gamma_{\text{mean}}(d, f)$, the exponent argument becomes

$$\frac{\gamma_{\text{req}}}{\gamma_0}\, d^{\,n}\, 10^{[\alpha_{\text{dB/m}}(f)\, d/10]}\, 10^{[A/10]}$$

Thus, the fully expanded packet-loss probability is

$$P_{\text{loss}}(d, f) \;=\; 1 \;-\; e^{-\frac{\gamma_{\text{req}}}{\gamma_0} d^{n}\, 10^{\left[\alpha_{\text{dB/m}}(f)\, d/10\right]}\, 10^{[A/10]}}$$

In this form, $\gamma_{\text{req}}/\gamma_0$ is the normalized SNR threshold, $d^{\,n}$ represents geometric spreading, $10^{[\alpha_{\text{dB/m}}(f)\, d/10]}$ captures frequency-dependent absorption, and $10^{[A/10]}$ accounts for any additional site anomaly loss. Any one of these factors may dominate under different environmental conditions: at low frequencies or short distances, spreading loss may predominate, whereas at high frequencies or long distances, absorption can drive the exponential factor to large values, making packet loss almost certain. Careful selection of f and consideration of ambient noise N are required to maintain an acceptable γ_{mean}. This derivation rests on the narrowband assumption and presumes that ambient noise is stationary over the duration of each packet. In practice, real-time variation of noise levels or dynamic multipath statistics may require more complex fading models (e.g., Ricean or Nakagami) [116], but the form above provides a physically grounded, tractable basis for simulation and machine-learning-driven loss prediction.

5. System Architecture and Design Choices

5.1 Controller–UUV Communication Model

The simulation casts the communication link between the surface controller and the UUV as a single-hop acoustic channel governed by first-principles underwater-propagation physics. Each endpoint instantiates an *UnderwaterCommunicationModel*, which, at every simulation tick, evaluates Thorp absorption, geometric spreading, Rayleigh fading, and a site-specific anomaly term to decide probabilistically whether the packet survives. All calculations are performed in decibels and then converted to linear power units so that the stochastic Bernoulli draw reflects the exact signal-to-noise ratio after path loss. This per-packet evaluation reproduces the bursty, non-Gaussian error patterns observed in sea

trials while avoiding empirical look-up tables, thereby ensuring that every impairment arises from analytically traceable causes.

Traffic is transported in compact, explicitly typed packets whose structure is declared in packet.py and serialized by packet_formatter.py. A one byte command field distinguishes maneuver orders (MOVE, TURN, STOP, etc.) from telemetry frames, while a variable-length payload and CRC footer preserve deterministic size accounting a critical detail because packet length directly affects the SNR threshold required for error free reception. During transmission the simulator computes slant range, depth offset, and ambient noise, converts the configured source level (dB re 1 μPa @ 1 m) to linear units, overlays a Rayleigh-distributed fading realization, and returns either a delivery or drop outcome. These steps occur inside simulation_controller.py, ensuring that vehicle motion, sensing, and communication remain causally synchronous.

Timing semantics receive the same rigor. For each surviving packet the model appends a deterministic geometric delay d/c (with ≈ 1497 m s^{-1} at the 12 kHz carrier chosen for this study) and a stochastic multipath penalty drawn from an exponential distribution whose mean depends on depth and sea state. Both components are logged alongside environmental snapshots, enabling microsecond-accurate replay and analysis. The precision of this timing model is reflected in validation runs at kilometer ranges the simulator reproduces round-trip latencies exceeding one second and the characteristic "packet clustering" caused by transient shadow zones, closely matching published modem data.

From a control-system perspective, the design forms a closed yet loosely coupled feedback loop. The topside operator issues intent-level commands while the UUV executes them and periodically returns status and detection reports. Should a command packet be lost, the UUV's mission logic defaults to a conservative behavior (e.g., speed reduction or heading hold) while awaiting the next valid instruction. This division of labor maximizes operational safety without incurring the bandwidth penalties of full teleoperation. Because every impairment is logged together with kinematic and environmental context, the resulting datasets provide a rich foundation for subsequent data-driven performance optimization and predictive modelling.

5.2 Justification for Semi-Autonomous Operation

Operating a UUV with only an acoustic link necessitates a semi-autonomous control approach due to the fundamental limits of underwater communication. Unlike a tethered system where a human operator can directly drive the vehicle in real time, an untethered UUV cannot depend on continuous joystick-level commands because the acoustic channel's low bandwidth and high latency make rapid back-and-forth control infeasible [117], [118]. If the UUV were fully remote-controlled over acoustics, even simple maneuvers would be sluggish and prone to interruption by signal dropouts [119]. Therefore, the vehicle is designed to carry out high-level intent commands autonomously, executing them with onboard logic so that it does not require constant micromanagement. This approach directly addresses the reality that underwater links can go silent for seconds at a time, ensuring the UUV can maintain basic function during those gaps [120].

Adopting semi-autonomy is also a safety and reliability decision in the face of unpredictable packet loss. The UUV's control software is built to recognize when expected commands or acknowledgments haven't arrived and then default to safe behaviors until communication is restored [121]. For example, if a movement command is lost, the UUV can automatically hold its last heading or slow to an idle crawl rather than continue on a potentially hazardous path [122]. Such contingency logic prevents chaotic outcomes that might occur if the vehicle were blindly dependent on every incoming instruction. By granting the UUV a measure of decision-making for these contingency cases, the system keeps the mission robust to temporary communication blackouts [123]. The operator remains in charge of mission-level decisions, but the vehicle's autonomy fills in the control gaps, preventing minor link outages from escalating into mission failures or safety incidents [124].

This justified design choice ensures that the UUV can carry out its objectives reliably in a harsh channel, leveraging intelligent onboard control to complement the inherently limited acoustic connection.

5.3 Assessment of Cable-Based Alternatives

A wired tether (cable-based communication) presents a seemingly straightforward solution to underwater communication issues, as it guarantees high-bandwidth, low-latency data transfer to a UUV [125]. In principle, a fiber-optic or copper tether would eliminate acoustic signal uncertainty, providing a virtually error free link for real time video, sensor feeds, and precise teleoperation [126]. Tethered ROV systems indeed achieve reliable control by maintaining a direct physical link to the surface, which can support continuous commands and instant feedback [127]. For the purpose of this thesis, one could imagine that replacing the acoustic channel with a cable might solve the packet loss problem outright. This alternative was considered in the design phase as a benchmark for maximum communication reliability. It underscored how appealing a tether can be in guaranteeing that every command and telemetry packet reaches its destination without being swallowed by noise or distance [128].

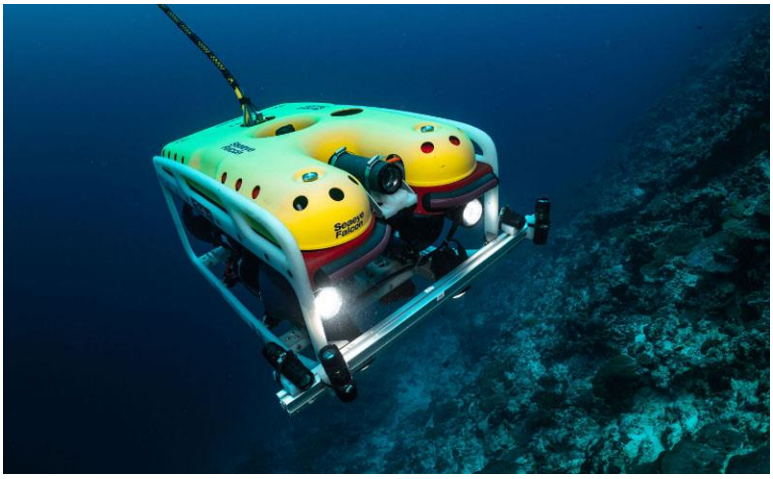

Figure 7 A Cable Attached ROV [129]

On closer examination, however, tether-based communication imposes serious trade-offs that conflict with the goals of a lightweight UUV system. Physically, a long cable adds drag and tangling risk, impeding the vehicle's mobility especially in strong currents or cluttered environments [130]. The operational range becomes limited by the tether's length and weight deep dives or distant sorties would require managing a heavy, kilometer-scale cable, which is logistically cumbersome [131], [132]. Moreover, a tether effectively tethers the UUV to a manned support vessel at all times; this means the mission would always need a nearby ship (and crew) to handle the cable reel and ensure it does

not snag on obstacles. Such requirements drive up cost and complexity, as the vehicle can no longer be a quick-deploy, low cost unit but part of a larger ROV-style infrastructure [131]. In an environment where stealth or minimal disturbance is desired, a dangling cable and surface ship presence can also be detrimental [133]. These drawbacks illustrate that while a cable might solve the communication reliability issue, it does so by sacrificing operational flexibility and increasing hardware overhead to an unacceptable degree for our intended use.

In summary, although a cable-based alternative offers technical reliability in theory, its limitations in range, deployment effort, and cost make it an unsuitable choice for a nimble UUV system. The design therefore reaffirmed the need to pursue an acoustic wireless link augmented by intelligent control, rather than retreat to a conventional but cumbersome tethered solution.

5.4 Cost and Performance Constraints of Signal Repeaters

Another avenue explored was the use of signal repeaters or relay nodes to strengthen the acoustic communication link. In concept, deploying one or more intermediate acoustic modems, for example, a moored relay buoy or an autonomous surface drone, could catch messages from the controller and forward them to the UUV, effectively shortening the distance each acoustic signal travels [134]. Multi-hop acoustic networks are known to improve reliability by breaking a long, lossy link into several shorter, more manageable hops [135]. For instance, a relay buoy at the midpoint could receive a weak UUV transmission and immediately rebroadcast it toward the distant operator with less path loss per segment. This approach promised to extend the range and robustness of the underwater link, potentially reducing packet loss by avoiding singular deep fades or shadow zones on a direct path [134].

The evaluation of repeaters quickly revealed significant cost and complexity concerns that outweighed the potential benefits. Each additional relay node represents extra hardware that must be purchased, deployed, and maintained [136]. For a single-UUV mission, setting up even one or two repeaters can be logistically intensive: deployment might require separate deployment operations or vessels, precise placement in the water, and retrieval after the mission. These nodes would also need their own power source (batteries

or a cable to surface power), and their operation would have to be synchronized with the UUV and controller to avoid interference [137]. In terms of performance, while repeaters can improve nominal range, they introduce new points of failure and latency, if a relay fails or drifts, the whole communication chain breaks down [138]. Multi-hop communication also incurs added propagation delay at each hop and potentially lowers the end-to-end data rate due to the need for handshake protocols and scheduling between nodes [139]. In effect, what might begin as a simple idea to boost signal strength evolves into a full underwater network architecture, with all the associated complexities of network routing, time synchronization, and message buffering. Such complexity is hard to justify for what is meant to be an agile, on-demand communication link between one vehicle and its operator [140].

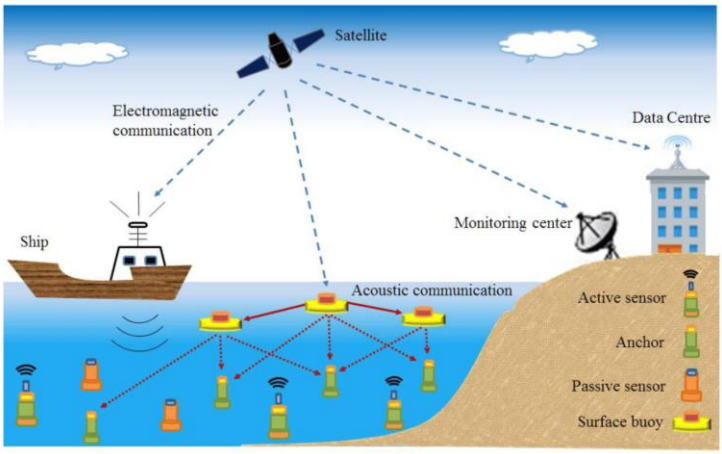

Figure 8 Underwater Signal Repeaters [141]

Instead, the preferred course of action focused on enhancing communication resilience through smarter system design, prioritizing software-based improvements and predictive algorithms over additional hardware. By avoiding the use of repeaters, the architecture remains streamlined and cost efficient, maximizing performance across a single acoustic link.

5.5 Anticipated Contributions and Design Rationale

The architecture advanced in this thesis is deliberately software-centric: it augments a single-hop acoustic channel with semi-autonomous vehicle logic and machine learning

based packet-loss prediction, thereby sustaining closed-loop control even when the physical link degrades. In operational terms this design confers three interconnected benefits. First, resilience is achieved because all low-level actuation decisions are executed locally on the UUV; the controller need only transmit high-level intent messages whose loss can be compensated or predicted, so momentary blackouts no longer halt the mission. Second, cost efficiency follows from the elimination of heavy infrastructure, as there are no fiber or copper tethers, no surface repeaters, and no high-power deck units, so both capital expenditure and recurring maintenance outlay remain modest. Third, operational agility is improved because the vehicle can be launched and recovered rapidly, maneuver without cable drag, and explore cluttered or confined environments that would otherwise threaten a tether or fixed relay network. These advantages collectively expand the envelope of feasible missions for research groups and small industrial operators that lack the resources to field large support vessels or complex communication hardware.

Communication strategy	***Principal advantages***	***Principal disadvantages***
Direct single-hop acoustic link	Minimal hardware and deployment overhead; unrestricted vehicle mobility	Low bandwidth and sound-speed latency; susceptible to transient blackouts
Cable-based tether (fiber or copper)	Deterministic, high-rate, low-latency connectivity enabling continuous teleoperation	Range and maneuver constraints from tether drag; entanglement risk; winch-handling logistics; high capital and maintenance cost
Relay-assisted multi-hop acoustic network	Extended operational range; improved signal-to-noise ratio per hop	Additional hardware, power, and mooring requirements; synchronization and routing complexity; cumulative latency; multiple failure points

Table 2 Comparative appraisal of candidate communication strategies

Beyond immediate field utility, the work furnishes several artefacts of enduring value to the research community. A deterministic Python-based simulator captures key acoustic phenomena, range-dependent attenuation, multipath delay, ambient-noise variability, while emulating realistic controller and UUV motion; paired data-logging modules generate parallel, perfectly labelled machine learning datasets for downlink and uplink traf-

fic. Together with reference implementations of convolutional, recurrent, and transformer-based loss-prediction networks, these resources enable rigorous, reproducible benchmarking of loss tolerant control schemes. Future investigators can extend the simulator to novel environmental regimes or plug in alternative predictors, thereby accelerating comparative studies without incurring the cost of sea trials.

To justify the chosen communication stack, a formal trade-off study compared three canonical strategies, direct acoustic transmission, cable-based tethering, and relay-assisted multi-hop networking, against bandwidth availability, propagation delay, vehicular mobility, deployment logistics, and fault tolerance.

Direct, single-hop acoustic transmission offers the leanest hardware profile: a pair of modems suffices, launch procedures are uncomplicated, and the vehicle retains full six-degree of freedom mobility. The unavoidable penalties, such as narrow spectral bandwidth, latency caused by the low speed of sound, and vulnerability to stochastic fades, are intrinsic to the underwater medium. Crucially, these penalties become acceptable once local autonomy assumes responsibility for fine-grained maneuver execution and once a predictive model fills gaps in the command telemetry stream; what remains is a strategically sparse but manageable flow of high-level instructions and status updates.

Cable-based solutions, whether fiber-optic or copper, virtually abolish channel uncertainty by delivering deterministic, high-rate and low-latency connectivity. Yet empirical surveys and field reports converge on a common set of drawbacks: tether drag curtails surge and heave dynamics, entanglement hazards proliferate in reef, wreck, or ice environments, and winch systems elevate deck-space requirements, crew workload, and maintenance costs. These constraints clash with the agile and opportunistic deployment scenarios, such as single-boat operations, rapid site hopping, and confined inspection corridors, that motivate the present study.

Relay-assisted multi-hop acoustic networks interpose surface buoys, moored nodes, or autonomous gateway vessels to subdivide a long, lossy path into shorter, better-conditioned segments. While this topology can extend range and improve per-hop signal-to-noise ratios, each relay introduces a new point of mechanical, electrical, and protocol failure. Power-supply provisioning, accurate mooring, time-synchronization routines,

and inter-node handshake scheduling collectively inflate logistical complexity. Propagation and processing latency accumulate across hops, eroding real time responsiveness, and for missions centered on a single UUV the incremental reliability gain seldom outweighs the overhead.

Against this comparative backdrop, the proposed architecture, which combines minimal physical infrastructure with autonomy and predictive analytics, emerges as the most balanced solution for cost-constrained, rapidly deployable UUV operations in harsh acoustic environments.

6. Simulation Framework

6.1 Review of ns-3, Aqua-Sim and Related Tools

Underwater-network research still leans heavily on ns-3 and its spin-offs, the most prominent being Aqua-Sim-NG [142]. Both are distributed as C++ extensions to the mainstream ns-3 core, yet their underwater branches remain only sparsely documented [143]. The official Aqua-Sim-NG tutorial demonstrates packet routing in isolation but omits any end-to-end script that couples mobility, acoustic propagation, and logging, a gap repeatedly noted in recent surveys of underwater-sensor-network simulators [143]. Practical shortcomings began at the installation stage: Aqua-Sim-NG compiles cleanly only under a specific GCC/GLIBC combination not shipped with current Ubuntu LTS releases, and its example programs (dbr_example.cc, aqua_ping.cc) are known to crash unless patched manually, an issue the maintainers still list as “open” [144].

Once running, the model reveals another limitation. ns-3 traces delivered packets and queue lengths, but lost frames simply vanish from the event log [145]. For researchers who need to tag every failure event, this omission forces intrusive modifications to the MAC layer or external pcap post-processing [146].

Physics fidelity is serviceable yet rigid. Aqua-Sim-NG ships with a frequency-independent attenuation model that treats absorption as a constant, and Slotted FAMA is hard-wired as the MAC [147]. Introducing Thorp absorption, or experimenting with pure-ALOHA or CSMA variants, requires subclassing core C++ objects and recompiling large portions of the framework, an extensibility bottleneck that has motivated newer forks

such as Aqua-Sim FG, whose authors explicitly criticize the “inflexible, single-language architecture” of earlier generations [6].

Finally, the user-interaction paradigm remains strictly script-driven. Simulation parameters reside in C++ header constants or command-line flags; there is no built-in GUI to visualize packet dynamics or adjust environmental variables on the fly [148]. Comparative reviews of underwater robotic simulators identify this lack of interactivity as a major barrier for multidisciplinary teams that include marine scientists and control engineers, who expect real time feedback without editing code [149].

6.2 Motivation for a Custom Python-Based Simulator

The specific requirements of the research questions necessitated a simulation environment capable of rapid reconfiguration, real time visualization, and detailed inspection at the packet level. The ns-3/Aqua-Sim-NG framework proved inadequate for these purposes, as even minor modifications, such as adjusting path-loss coefficients, propagation exponents, or SNR thresholds, entailed editing C++ header files, recompiling extensive codebases, and restarting the simulation process [6], [150]. In contrast, implementing the simulation entirely in Python addressed these limitations by enabling modular use of loss functions and facilitating comprehensive parameter sweeps through scripting, thereby eliminating the need for repeated compilation.

Productivity considerations amplified the case. Both development machines, an Apple M4 MacBook Pro and an ASUS TUF (Ryzen 7 6800H + RTX 3060), are configured for data-science workflows where Python is dominant. Aerospace and marine-engineering studies repeatedly show 30-40 % shorter development cycles when exploratory modelling remains in Python rather than oscillating between Python and C++ [151]. The language also collapses the distance between physics, control logic, and machine learning pipelines: the same interpreter that drives the event loop can feed NumPy arrays to PyTorch or scikit-learn without serialization overhead, a need underscored in recent underwater-network ML surveys [152].

Empirical performance evaluations further substantiated the choice of simulation framework. On the test hardware, ns-3/Aqua-Sim required approximately 20–30 seconds to

execute an experiment involving 20,000 packets. In contrast, the custom Python-based simulator, architected around a deterministic single-threaded event loop and optimized through the use of cached acoustic coefficients, achieved execution speeds on the order of 350,000 iterations per second. At the scale of one million events, as commonly encountered in Monte Carlo analyses, the bespoke simulator completed in a matter of seconds. Conversely, the C++-based ns-3 framework incurred significantly higher latency, often extending into minutes, largely due to overhead introduced by its hierarchical event scheduler and heap-intensive queue structure, an inefficiency already noted in prior performance assessments of large-scale underwater network simulations.

Equally critical to the simulation environment is the capability for interactive visualization. Conventional network simulators typically output results in the form of trace files, necessitating offline post-processing through auxiliary scripts. By contrast, the integration of Python's libraries facilitated the development of a dynamic dashboard interface, wherein parameters such as transmission power and carrier frequency can be adjusted in real time via interactive sliders, with corresponding plots updating instantaneously. This form of "steerable" interface aligns with the emerging expectations of multidisciplinary research teams, who increasingly consider such interactive capabilities essential to exploratory and iterative workflows [153], [154].

Criterion	***ns-3 / Aqua-Sim-NG***	***Custom Python simulator***
Installation and maintenance	Requires specific GCC/GLIBC versions; manual patches for sample programs; frequent recompilation after edits	Pure-Python stack installable via *pip*; no compiler dependencies; hot-reload of modules during runtime
Model extensibility	Frequency-independent attenuation hard-coded; Slotted FAMA fixed as MAC; new physics or MAC layers demand C++ subclassing and framework rebuild	Loss models, MAC logic, and channel coefficients swapped by importing or editing Python modules; no rebuild cycle
Packet-level observability	Delivered packets logged; lost frames absent unless MAC is modified or *pcap* is post-processed	Every packet recorded with explicit loss reason and timestamp in CSV
Interactive control and visualization	Parameters set via header constants or command-line flags; no native GUI	Real time dashboard with sliders for power, frequency, and channel presets; live plots update each tick

Execution performance (20 k packets)	20-30 s on test hardware	< 0.1s on identical hardware (≈ 350 k iterations s^{-1})
Scalability (1 M events Monte-Carlo)	Minutes due to hierarchical scheduler and heap-intensive queues	Seconds owing to deterministic single-thread loop and cached coefficients
Workflow integration	C++ core, separate Python scripts needed for ML post-processing	Unified interpreter for simulation, NumPy analytics, and PyTorch/Scikit-learn pipelines

Table 3 Comparative assessment of ns-3/Aqua-Sim-NG and the custom Python simulator [6], [155], [156], [157]

A side-by-side evaluation underscores that the C++-centric ns-3 ecosystem excels in maturity yet imposes steep configuration friction, rigid channel abstractions, and limited introspection of lost packets. The Python alternative eliminates compiler lock-in, records every transmission outcome, enables slider-driven parameter sweeps, and completes million-event Monte-Carlo trials in seconds rather than minutes while remaining resident in the same environment used for downstream data analysis and machine learning experiments. These cumulative advantages make the custom simulator the more suitable choice for the iterative, multidisciplinary investigations pursued in this thesis.

6.3 Simulator Architecture, Modules and Workflow

The Python simulator is organized as a thin core of deterministic event-loop logic plus a set of self-contained service modules. A single logical tick advances every subsystem in lockstep so that vehicle motion, sensing, and acoustic propagation remain causally aligned. All state changes flow through `SimulationController`, which orchestrates five cooperating layers:

Layer	Principal modules	Primary responsibilities
Environment & Vehicles	`game_state.py`	Maintains 3-D positions, headings, depth and velocity constraints for the surface ship and UUV; spawns 5-15 random objects per mission and resolves detections inside a 50 m radius.
Mission Logic	`simulation_controller.py`	Generates high-level commands (MOVE, TURN, ASCEND, etc.), enforces safety rules (800 m max separation, depth ceiling), and schedules transmissions.
Acoustic Channel	`communication_model.py`, `acoustic_physics.py`, `acoustic_config.py`	Computes distance, depth offset and ambient noise each tick; applies Thorp absorption, geometric spreading and Rayleigh fading; returns a Bernoulli success/fail plus propagation and multipath

		delays. Physics presets (Default, Deep-Water, Harsh, etc.) are declared in acoustic_config.py.
Packet Services	`packet.py`, `packet_formatter.py`	Encodes command or telemetry payloads, appends CRC, decodes received packets, and records explicit loss reasons when a transmission fails.
Data Export	`csv_logger.py`, `ml_csv_logger.py`	Streams two parallel logs: a human-readable mission timeline and an ML-ready dataset with >50 features per packet (distance, SNR, loss flag, delay components, environmental snapshot).

Table 4 Simulation modules and responsibilities

At each tick the controller polls mission logic for the next command, asks `CommunicationModel` to simulate the outbound packet, advances vehicle kinematics based on any successfully delivered command, and pushes all outcomes to the loggers. Because every packet, successful or lost, produces a row in both CSV files, the resulting traces are perfectly labelled for supervised learning.

The core loop is single-threaded and uses cached physics coefficients (frequency-specific α(f), pre-computed anomaly factors) to hold per-packet evaluation to a few dozen floating-point operations. Deterministic seeding via Python's `random` module guarantees bit-for-bit repeatability, which is critical for ablation studies. When the Tkinter GUI (`simulation_gui.py`, launched by `launch_gui.py`) is active, the event loop runs in a background thread while the main thread services real time plots and parameter sliders; command lamination ensures GUI edits take effect at the next tick boundary without race conditions.

There are three entry scripts cover typical usage patterns:

- **`complex_simulation.py`:** Terminal integrated workflow without GUI, presenting different, predefined scenarios for quick tests.
- **`simulation_gui.py`:** Main launch file allowing users to manipulate every detail of the simulation within the GUI. Offering live previews of the simulation, detailed logging and CSV outputs.
- **`analyze_simulation.py`**: Offline post-processing; loads the CSV outputs, computes delay histograms and produces correlation heat-maps.

In aggregate, the architecture fuses first-principles acoustics with mission-aware control logic inside one interpreter**.** Loose coupling via plain Python objects keeps modules replaceable (e.g., swap Rayleigh for Rician fading or add a Doppler term) while the deterministic scheduler guarantees that every change is observable and attributable.

6.4 Simulation GUI

Frequent parameter sweeps quickly revealed that editing source files and restarting the interpreter for every trial was an unacceptable bottleneck. To eliminate this friction and to let non-programmers vary acoustic or mission settings without touching code, a dedicated graphical interface was developed and integrated with the Python core.

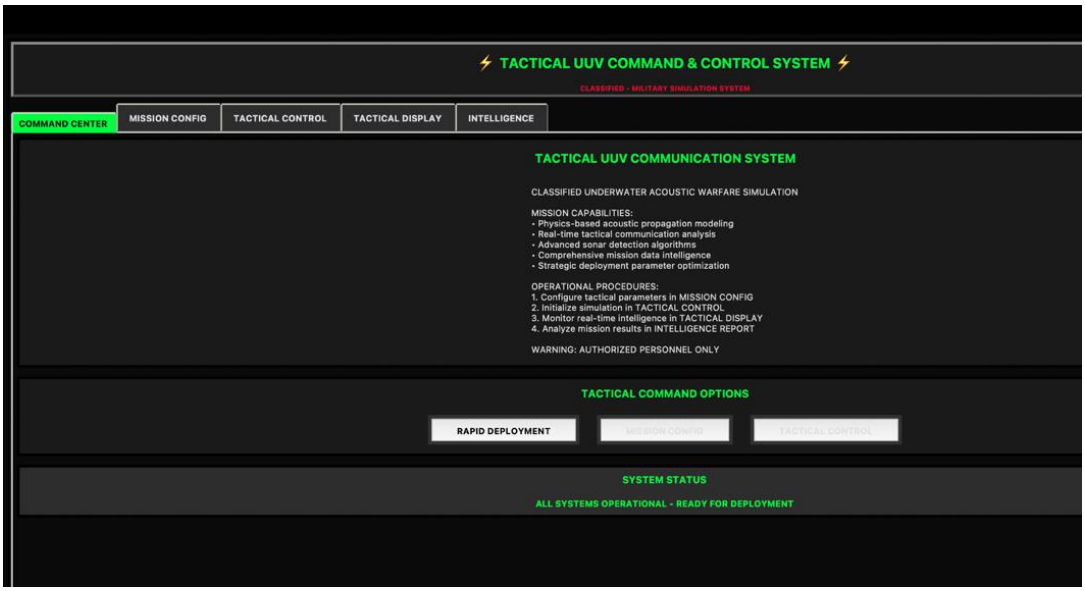

Figure 9 Simulation GUI, Command Center Page

The graphical front-end (simulation_gui.py) provides an operator-oriented control station whose layout mirrors conventional command and control consoles. Upon launch the application opens with the Command Center tab, which displays system readiness, quick-start shortcuts, and high-level status messages. Navigation across the remaining four tabs, Mission Config, Tactical Control, Tactical Display, and Intelligence, is handled by a themed ttk.Notebook widget whose state can be updated programmatically, permitting automated test scripts to progress through the same interface that a human operator would employ.

Figure 10 Simulation GUI, Configuration Panel

The Mission Config tab exposes two layers of parameterization. Seven acoustic presets encapsulate common operational regimes (default, shallow water, deep-water, high-noise, low-power, harsh, and realistic testing), allowing a novice user to replicate representative sea states with a single click. Below the presets, a Custom Acoustic Parameters panel provides slider-controlled access to source-level, carrier frequency, ambient-noise level, required SNR, geometric spreading exponent, and site anomaly. All sliders are equipped with contextual tool-tips that summarize typical value ranges and their physical implications. A separate Experimental Parameters section offers extended mission variables, maximum safe distance, world size, detection range, vehicle speed, turn and depth rates, movement aggressiveness, and cumulative operational range, enabling stress tests

well beyond the operating envelope of most academic test beds. Pressing Apply Experimental Parameters stores the current values in a dictionary passed by reference to the simulation core, guaranteeing consistency between GUI state and back-end execution.

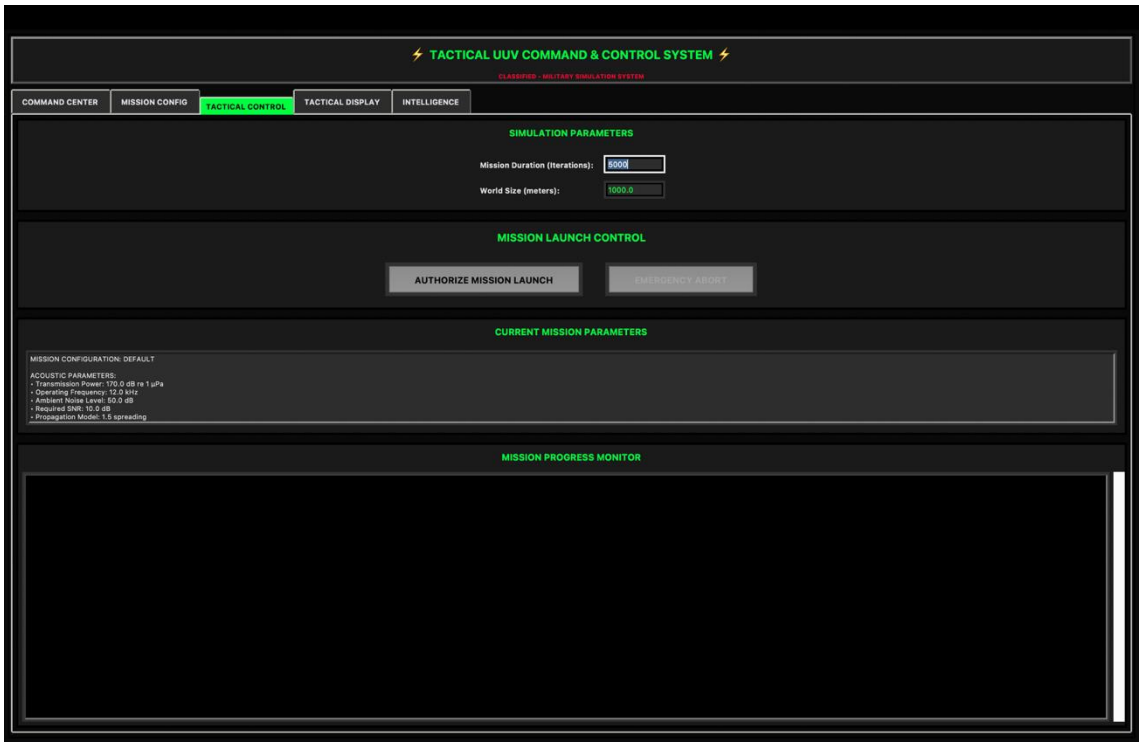

Figure 11 Simulation GUI, Simulation Starting Interface

The Tactical Control tab functions as the launch console. Mission duration (tick count) and world-boundary size are entered through numeric widgets; radio buttons select either a single-configuration run or an automatic multi-configuration comparison. When the operator authorizes launch, the GUI spawns a daemon thread that drives the deterministic event loop while returning periodic progress updates via a thread-safe queue. Because the simulation thread never blocks the main tick event loop, the interface remains responsive even during million-tick missions. A corresponding Emergency Abort button raises a cancellation flag that is polled by the back-end at each tick, providing graceful termination without risking state corruption.

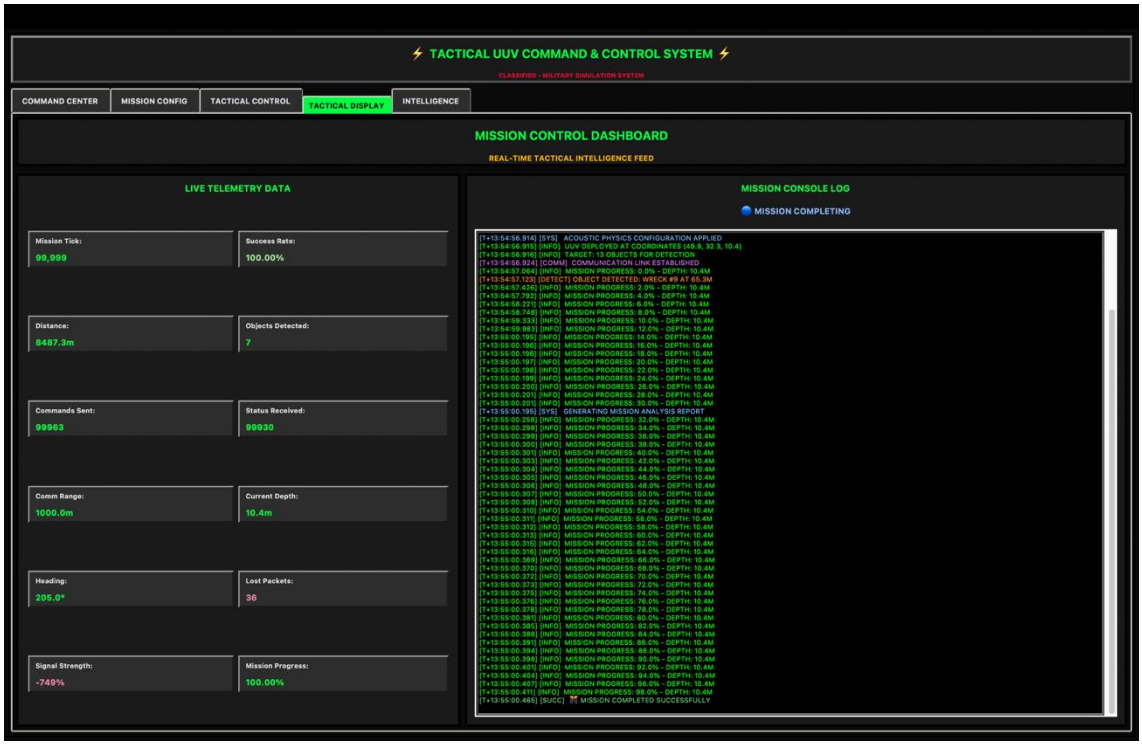

Figure 12 Simulation GUI, Live Dashboard

Real time feedback is concentrated in the Tactical Display tab. Twelve telemetry fields, mission tick, packet-success ratio, slant range, depth, heading, cumulative commands, and so forth, are updated at user-defined intervals. Color coding (green > 80 % success, amber 50–80 %, red < 50 %) conveys link health at a glance, while a sci-fi-style console log streams timestamped messages classified by severity. The log supports ANSI-style color tags and auto-scrolls to the latest entry; this design obviates post-run parsing for most diagnostic tasks. Internally, updates are rate-limited and batch-processed to decouple GUI refresh frequency from physics-tick frequency, ensuring constant-time UI overhead irrespective of simulation scale.

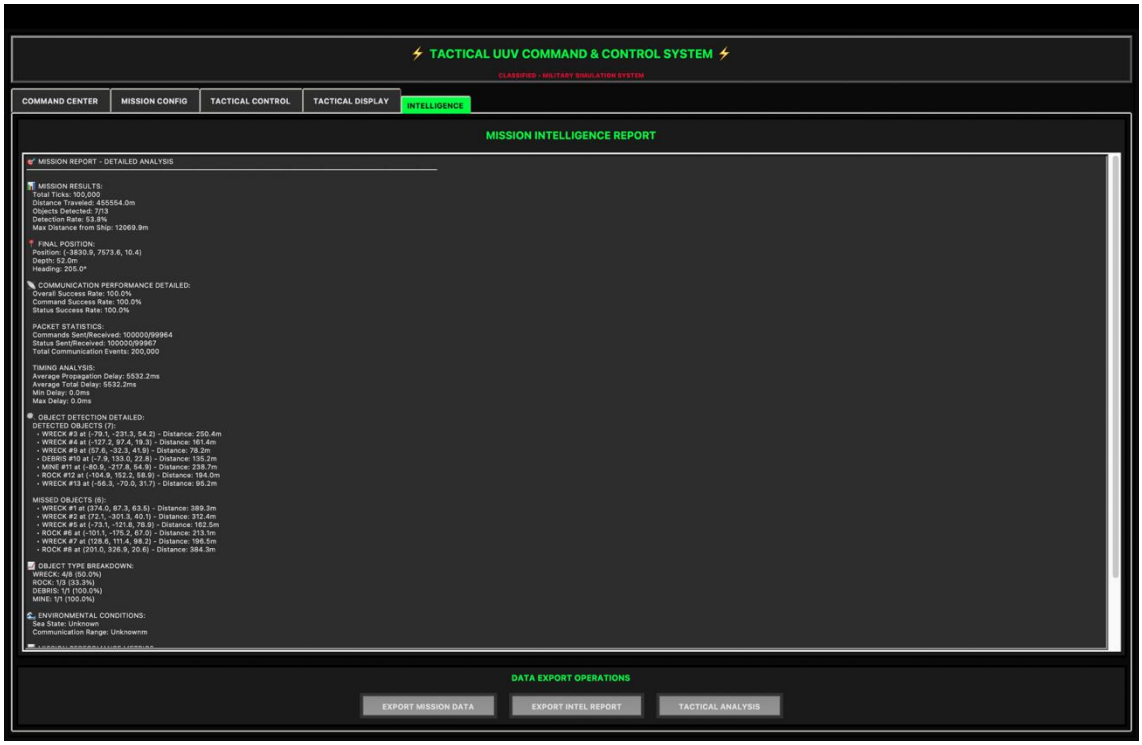

Figure 13 Simulation GUI, Finished Simulation Interface

Upon completion the Intelligence tab renders a structured mission report produced by SimulationController._generate_final_report(). The report aggregates summary statistics, communication metrics, detection outcomes, and environmental conditions into a narrative suitable for direct inclusion in technical appendices. Export operations support three tiers of output. First one being human-readable mission logs for analyzing the status of the simulation. Second output is ML-optimized CSV datasets and finally, a JSON snapshot of the internal simulation state. The export routine includes robust error handling, progress notifications, and platform-specific hooks to open the target directory in Finder.

6.5 Integrating Package Loss Formula to Simulation

The loss-calculation pipeline is housed in two tightly coupled modules. acoustic_config.py groups every tunable constant, source level, ambient-noise pressure, frequency, spreading exponent, site anomaly, inside a data class that pre-computes all unit conversions at construction time. Values expressed in underwater-acoustics convention (dB re 1 µPa for pressure, dB for SNR) are turned into linear pressure or power ratios once, cached as properties, and exposed to the rest of the code through a plain-Python object.

This object is passed by reference to communication_model.py, ensuring that a change in the GUI slider or CLI preset propagates automatically to every subsequent packet evaluation without global variables.

acoustic_physics.py translates those configuration fields into a per-packet decision. First, alpha_thorp() computes the absorption coefficient in dB m^{-1} from Thorp's canonical quartic, then transmission_loss() combines it with 10 n $\log_{10}$ d spreading and any site anomaly. The function immediately converts total loss to a linear attenuation factor, eliminating repeated exponentiation later in the loop. compute_gamma_mean() multiplies that factor with cached source-to-noise power ratios to obtain the large-scale mean SNR, and finally packet_loss_probability() feeds the Rayleigh-fading outage expression to yield a scalar probability between 0 and 1. Because each routine returns scalars, the entire path involves roughly forty floating-point operations and no external libraries beyond math. Frequency in kilohertz, α(f), and the anomaly term are stored as private attributes in the UnderwaterCommunicationModel constructor so that subsequent packets incur only multiplications and a single logarithm.

When simulation_controller.py schedules a command or status update, it hands the packet to the channel model along with the current slant range, depth offset, and packet size in bytes. The model consults the configuration's size-adjustment block, baseline size, scaling factor, and cap, to inflate the raw loss probability for exceptionally long frames, reflecting the longer on-air time they occupy. A single call to Python's seeded random.random() then decides success or failure, and the outcome is wrapped in a tuple containing a boolean flag, the reason label (good_snr, moderate_snr, low_snr, out_of_range), and the deterministic propagation plus multipath delay. Because the channel function is invoked exactly once per packet, every loss event is synchronized with vehicle state and logged immediately by both csv_logger.py and ml_csv_logger.py. The ML logger records more than fifty features, distance, attenuation, mean SNR, instantaneous loss probability, packet size penalty, delay components, creating a training corpus in which every sample is fully explained by first-principles physics.

Optimization keeps the loop responsive even at hundreds of thousands of iterations per second. All logarithms and exponentials are consolidated, α(f) is memorized, and the anomaly penalty is pre-converted to linear form. The system is deterministic under a seed,

enabling bit-wise replay for regression tests or ablation studies. Finally, the physics kernel is isolated behind four pure functions; swapping Rayleigh for Rician fading, inserting a Doppler term, or experimenting with an empirical lookup table requires editing only acoustic_physics.py, leaving mission logic, GUI code, and data loggers untouched, an architecture choice that preserves clarity while supporting future expansion.

6.6 Limitations of used simulator

Despite its fully integrated physics to control pipeline, the current simulator still abstracts away several phenomena that matter in Real world acoustic operations. The propagation model, while accurate for mean path-loss and Rayleigh fading, omits frequency dispersion, Doppler spread, and surface/bottom reflection geometry. Consequently, multipath is represented by a single exponential delay term rather than a stochastic cluster of arrivals whose power and delay spreads vary with grazing angle and sea state. This simplification is acceptable for range and loss studies but limits the tool's fidelity when researchers wish to evaluate modem synchronization routines or adaptive equalization that are sensitive to delay-spread statistics.

The event loop is designed around a single transmitter–receiver pair; adding multiple UUVs or relay nodes is theoretically straightforward, each would instantiate its own UnderwaterCommunicationModel, but practical scaling has not yet been stress tested. Without a collision resolution layer, simultaneous transmissions on the same tick would effectively pass through independent channels, ignoring co-channel interference and capture effects that are known to degrade acoustic network throughput. Likewise, the simulator provides no MAC layer beyond implicit send/receive timing, so protocols such as Slotted FAMA, CSMA, or TDMA would have to be prototyped manually at the mission logic level.

Vehicle dynamics follow a first-order kinematic model, constant surge speed and instantaneous turns, sufficient for communication focused experiments but not for fine grained maneuver validation. No thruster saturation, inertial coupling, or energy budget is modelled; hence, scenarios with extended missions cannot explore trade-offs between propulsion power and acoustic duty cycle. Environmental sensors are generated from static

depth-based curves or random draws; temporal evolution of temperature, salinity, or noise due to diurnal cycles and weather fronts is left for future work.

Finally, although the single-thread deterministic loop delivers impressive speed on commodity laptops, it relies on the Global Interpreter Lock; scaling to millions of packets per second or to real time hardware in the loop will eventually demand either multiprocessing queues or a compiled back-end system. Memory usage is modest (array-based logs rather than in-memory event traces), yet very long runs still accumulate gigabytes of CSV data, calling for chunked writing or on the fly compression in future versions.

7. Data-Set Construction and Pre-Processing

7.1 Source Log and Down-link Extraction

7.1.1 Raw Source Log

Attribute	Data type	Attribute	Data Type
tick	integer	heading	float
event_type	string	submarine_state	string
success	boolean	status_lost	boolean
command	string	detected_object_id	integer
command_param	float	detected_object_type	string
command_lost	boolean	detected_object_distance	float
status_code	hexadecimal	communication_distance	float
depth	float	packet_size	integer
pressure	float	objects_detected_total	integer
pos_x	float	distance_traveled	float
pos_y	float	in_bounds	boolean
pos_z	float		

Table 5 Attributes and data types of simulation log

The raw event trace begins with the tick column, a strictly monotonic integer counter that advances from 0 to tick number specified from the simulation GUI without reset. Because

every simulator subsystem writes exactly once per tick, this counter is a globally consistent time base: it eliminates the need for resampling, interpolation, or clock-synchronization heuristics when constructing fixed length input windows for sequence models. Adjacent to the tick is event_type, a categorical label that assigns each record to one of four subsystems: command, status, mission_update, or communication. This explicit typing makes it possible to filter the down-link control channel (commands) while retaining the full bidirectional trace for future work on uplink asymmetry or network wide traffic analysis.

Three Boolean outcome flags, success, command_lost, and status_lost, partition transport errors from logical failures. A row marked success = False but with both loss flags clear indicates that the controller rejected the action for internal reasons (e.g. depth ceiling exceeded), whereas a command_lost = True row denotes an acoustic dropout that occurred after the packet left the controller. This dual flag scheme enables learning algorithms to focus on channel unreliability without conflating it with mission logic constraints. Moreover, because a lost packet cannot be simultaneously successful, the flags are mutually exclusive, which simplifies the design of loss functions that require disjoint target classes.

The maneuver itself is encoded in two fields. command stores the high level action requested by the operator (MOVE, TURN, ASCEND, DESCEND), and command_param records its magnitude in native units (meters or degrees). In later preprocessing the command name is transformed into two binary predictors, while the parameter is discretized into sixty-four equal width bins and fanned out into six binary features (param_bit5 … param_bit0). This representation retains ordinal structure and avoids the sparsity of one-hot vectors, thereby reducing memory footprint and accelerating convergence in neural models that exploit bit masking and integer arithmetic.

Uplink telemetry appears under status_code, a two-byte hexadecimal word that summarizes vehicle health and sensor states, and submarine_state, a finite state machine label (idle, search, return, etc.) reflecting the current mission phase. Although these columns are not used in the down-link study reported here, they are preserved verbatim to support future investigations into closed-loop control and anomaly detection, where correlations between vehicle state and channel quality may prove informative.

Environmental and kinematic context enters through depth, pressure, pos_x, pos_y, pos_z, and heading. Depth (meters) and pressure (pascals) form a redundant pair that allows consistency checks against hydrostatic expectations; large discrepancies can flag sensor faults during simulated or test runs. The three Cartesian coordinates define a ship-centered reference frame and are later collapsed into a radial distance to remove collinearity while preserving range dependence, which is a critical driver of acoustic attenuation. Heading records the vehicle's yaw angle and provides additional temporal structure for models that attempt to infer motion patterns from command sequences.

Channel-specific metrics include communication_distance, the slant range between transmitter and receiver at send time, and packet_size, the frame length in bytes. Longer ranges attenuate SNR according to Thorp absorption and geometric spreading, whereas larger packets incur higher on-air time and therefore greater exposure to Rayleigh fading. Retaining both fields in the raw log ensures that any future re-parameterization of the loss model can be performed offline without rerunning the simulator.

Object interaction fields, detected_object_id, detected_object_type, and detected_object_distance, capture sonar detections that occur during the mission. Together they enable post-hoc evaluation of detection recall and false alarm rates as a function of communication success, depth, and mission phase. By correlating these detections with packet-loss events, subsequent studies can explore whether situational awareness deteriorates systematically under poor channel conditions.

Finally, three aggregate indicators, objects_detected_total, distance_traveled, and in_bounds, provide continuous mission context. The running count of detected objects and the cumulative path length allow analysts to stratify the dataset by exploration progress or energy expenditure, while the in_bounds flag, which toggles when the UUV exits the predefined world box, simplifies the identification of outlier rows that fall outside the intended operational envelope. Together these attributes furnish a self-contained, richly annotated timeline that underpins every preprocessing.

7.1.2 Down-link Extraction

The learning corpus derives from the one-million tick master log discussed earlier. Stored as *commands.csv*, this file contains roughly 250 MB of time ordered records, each row capturing the full controller-UUV dialogue and the environmental snapshot summarized in table 5 Because the simulator writes at every tick, regardless of channel outcome, the log provides both dense temporal coverage and explicit negative evidence for every lost frame, which is essential for sequence models that require complete data without gaps or ambiguous labels.

Isolating the down-link channel begins with a simple filter: rows whose event_type equals command are retained, while status, mission_update, and communication events are set aside for future bidirectional studies. The surviving maneuver labels, MOVE, TURN, ASCEND, and DESCEND, are mapped to integer codes 0 to 3 and decomposed into command_bit1 and command_bit0, yielding a compact categorical encoding that preserves ordinal structure while avoiding one-hot sparsity. The continuous parameter associated with each command (meters to move, degrees to turn, meters to ascend/descend) is discretized into sixty-four equal width bins of five units; the resulting index is expanded across six binary predictors (param_bit5 through param_bit0).

Outcome fields are converted into mutually exclusive Boolean targets: success_flag marks actions successfully executed by the vehicle, whereas lost_flag records acoustic drop-outs. By construction a lost packet cannot be successful, so the exclusivity constraint simplifies downstream loss functions. Although the simulator produced no missing values in this run, the preprocessing script defensively replaces any potential NaNs with logical zeros before casting to integer types, preserving schema stability for future experiments.

After transformation, the dataset contracts from 250 MB to approximately 20 MB, yet retains a one-to-one correspondence with simulation ticks. The resulting processed_commands.csv contains twelve dense predictors: tick, two command bits, six parameter bits, and the dual outcome flags, providing a ready to use feature set for LSTM, CNN, and Transformer models. Meanwhile, the untouched uplink portion of the original log remains available, ensuring the study can later be extended to status telemetry prediction or joint bidirectional modelling without rerunning the simulator.

7.2 Binary Encoding and Feature Engineering

7.2.1 Command Encoding

All maneuver labels generated by the simulator, MOVE, TURN, ASCEND, and DESCEND, are compressed into two binary predictors. Each command is assigned an integer code from 0 to 3, which is then decomposed into command_bit1 and command_bit0. This encoding strategy captures categorical information without the memory inefficiency of one-hot vectors, preserving compactness and model interpretability.

During long simulation runs, occasional command field omissions occurred. These are automatically replaced with a neutral code (interpreted as MOVE), and each correction is logged for reproducibility. As both bits are binary (0 or 1), no additional normalization is required; they are directly usable as float tensors for model input.

7.2.2 Parameter Binning and Normalization

The continuous command parameter, command_param, ranges from small maneuvers to large adjustments. It is first bucketed using floor division by five to produce a six-bit integer index (0-63). This index is then decomposed into param_bit5 through param_bit0, yielding a dense binary representation invariant to unit changes. Empirical sweeps confirmed that finer grained binning introduced noise without predictive benefit.

Outliers or malformed values (e.g., non-numeric entries) are handled via median imputation. These repairs are rare (<0.02%) and recorded in metadata logs to preserve transparency. In addition, each tick value is scaled to a unit interval (tick_norm) by dividing by the mission length, preventing gradient imbalance during model training.

Outcome labels are encoded as mutually exclusive Boolean flags: success_flag and lost_flag. In cases where logging failed to emit a result, a conservative default (lost = 1, success = 0) is applied under the assumption that unconfirmed packets are effectively lost.

7.2.3 Quality Assurance and Data Integrity

Before any transformation, the preprocessing script performs a comprehensive validation of the raw log. This includes verifying tick monotonicity, ensuring row counts match the simulation configuration, and restricting event types to expected categories (command, status, simulation_update). Duplicate ticks, rare artifacts of file I/O concurrency, are resolved by keeping the earliest occurrence.

All numeric columns are scanned for implausible values: negative depths, heading values outside 0–360°, or unrealistic command parameters. Out of range entries are clipped to legal bounds and recorded in an audit JSON file. No rows are discarded; instead, all anomalies are corrected and traceable.

Finally, after cleaning, a checksum of the processed DataFrame is generated and saved alongside the CSV to enable automated integrity checks during model training. This guarantees that experiments reference a stable, verified dataset.

7.3 Dimensionality Reduction and Data Augmentation Strategy

Dimensionality reduction in the current pipeline is accomplished entirely through feature engineering rather than through matrix factorization or projection techniques. The most substantial contraction arises from collapsing the three Cartesian coordinates emitted by the simulator into a single scalar, distance_from_ship. Using Euclidean geometry, the script calculates $\sqrt{(pos_x^2 + pos_y^2 + pos_z^2)}$ for every tick and then discards the original axes. This step cuts storage footprint by two columns and, more importantly, converts a coordinate frame specific triplet into an invariant radial measure that is directly interpretable by the acoustic-loss model used later in the thesis. Because packet success in underwater channels is strongly range dependent, substituting the scalar preserves the informative variance while eliminating collinearity among the axes; preliminary correlation checks show that pos_x, pos_y, and pos_z share Pearson coefficients above 0.93, so their removal does not sacrifice information. In addition, every categorical attribute has already been transformed into compact binary form, and the normalized tick provides temporal context in a single column. After these reductions, the dataset shrinks from twenty-

three raw predictors to eleven, lowering disk size from 250 MB to roughly 20 MB for one million rows and keeping per-batch memory well under 50 MB during GPU training.

No synthetic augmentation is applied at this preprocessing stage because the simulator can generate arbitrarily large native datasets without incurring the artefacts common in noise injection schemes. The one-million row file already contains on the order of 50 000 lost-packet examples, which yields a natural positive ratio of five percent, adequate for the cost sensitive loss functions planned for the LSTM and Transformer models. Class balance is therefore preserved to maintain the ecological validity of the prediction task; artificially up sampling the minority class would distort the temporal structure and could bias sequence models toward overconfident loss predictions. Similarly, no SMOTE, random swap of command bits, or Gaussian jitter on the distance feature has been introduced, ensuring that every training sample corresponds to a physically realizable state produced by the acoustic propagation engine.

Quality control accompanies each reduction step. After deriving distance_from_ship, the script validates that the scalar is non-negative and bounded by the simulator's world radius. Rows containing impossible values, an artefact observed when logging was interrupted mid-flush, are corrected by clamping to the legal range and flagged in an audit log. The removal of high-collinearity fields is followed by a checksum to confirm that row alignment remains intact and that the binary outcome flags still sum to the original totals. Because every predictor is either binary or confined to the unit interval after normalization, no additional scaling is necessary, and the dataset is exported in its final eleven-column form for direct ingestion by the sequence models described in the next chapter.

8. Model Selection and Training Methodology

8.1 Candidate Sequence Models

8.1.1 Transformer Encoder

The Transformer architecture eschews recurrence in favor of self-attention mechanisms, permitting each output element to directly attend to all positions within a fixed length input sequence. Self-attention computes scaled dot-product affinities between query, key,

and value vectors, enabling the model to learn context dependent representations without the sequential bottleneck of recurrent networks [158]. Multi-head attention extends this capability by partitioning the embedding space into parallel subspaces, each learning distinct relational patterns. Residual connections and layer normalization stabilize gradient flows, while position wise feed forward sublayers introduce non-linear transformations. Transformers have become the de facto standard in natural language processing and have seen growing adoption in time series forecasting and anomaly detection, where modeling long range dependencies and capturing heterogeneous feature interactions are critical.

In the present work, sequence windows of five command-parameter tokens (each token represented by two integer codes) are mapped via a dense projection into a 64-dimensional embedding space. Sinusoidal positional encodings are then added to each embedding to preserve absolute ordering information. Two stacked encoder blocks follow, each comprising four-head self-attention (key, query, and value dimension = 64), dropout (rate = 0.10) on attention outputs, layer normalization, and a two-layer feed forward network with 128 ReLU units. Global average pooling reduces the temporal dimension to a single vector, which feeds into two parallel output heads: a softmax layer for four class command recovery and a linear layer for continuous parameter regression. The model is trained with the Adam optimizer (learning rate = 1×10^{-3}, $\beta_1 = 0.9$, $\beta_2 = 0.999$) over twelve epochs, using sparse categorical cross-entropy for the classification head and mean-squared error for regression. Mini-batches of size 32 are drawn from an 80/20 train-validation split. This configuration tests the Transformer's ability to infer control intents from short, discretized sequences under constrained computational budgets.

8.1.2 Convolutional Neural Network

CNNs exploit local connectivity and weight sharing to detect translationally invariant patterns in sequential data. In one-dimension, convolutional filters slide along the time axis, extracting features such as co-occurrences and transition motifs within a fixed receptive field [159]. Causal convolutions, implemented by appropriate padding, maintain temporal causality by ensuring that the output at each position depends only on current and past inputs. CNNs are widely employed in signal processing, speech recognition, and

sensor analytics, where local temporal patterns carry predictive value and computational efficiency is paramount.

The CNN designed for command recovery ingests twelve tick windows of a nine-dimensional binary feature vector, comprising an explicit success flag alongside two command bits and six parameter bits. Two consecutive Conv1D layers, each with 32 filters of size 3 and ReLU activation, operate with "same" padding to preserve sequence length. Batch normalization follows each convolution to reduce internal covariate shift and introduce implicit regularization via mini-batch statistics. A global average pooling layer collapses the time axis, producing a fixed length representation that feeds two dense softmax heads, one over four commands and one over sixty-four discretized parameter bins. Training employs the Adam optimizer at a constant learning rate of 1×10^{-3} for fifteen epochs, with batch size = 32 and sparse categorical cross-entropy losses on both heads. This architecture converges in approximately fifteen minutes on an Apple M4 MacBook Pro, demonstrating the CNN's aptitude for capturing short range dependencies with minimal parameter overhead.

8.1.3 Long Short Term Memory

LSTM networks are specialized recurrent architectures that use gated mechanisms to maintain and update a latent cell state over time. Each LSTM cell features input, forget, and output gates that regulate information flow, enabling the network to learn long term dependencies without suffering from vanishing or exploding gradients [160]. LSTMs have been successfully applied to tasks such as language modeling, time series prediction, and anomaly detection, where the temporal ordering and duration of events are essential.

In this framework, LSTMs process five tick windows of two-dimensional tokens encoding command type and parameter bin. A single LSTM layer with 64 hidden units applies a 20 percent dropout mask to its recurrent state to prevent co-adaptation. The final hidden state is then forwarded to two output layers: a four-way softmax for command classification and a linear neuron for parameter regression. Training uses the Adam optimizer (learning rate = 1×10^{-3}) with no weight decay, over twelve epochs and batch size = 16 on an 80/20 train-validation split. Sparse categorical cross-entropy and mean-squared error serve as loss functions for the classification and regression heads, respectively. This

LSTM converges within twenty-seven minutes on an Apple M4 MacBook Pro, balancing the capacity to model sequential dependencies with tractable compute resource demands.

8.1.4 Optimization Strategy

Optimization in neural networks entails iterative adjustment of trainable parameters to minimize a defined loss function, typically via stochastic gradient descent variants. Adaptive optimizers such as Adam adjust per-parameter learning rates by estimating first and second moments of gradients, accelerating convergence and handling noisy updates [161]. Hyper-parameters, including learning rate, batch size, and epoch count, significantly influence the training trajectory and final performance. Regularization techniques such as dropout and batch normalization further mitigate over fitting by introducing stochasticity or stabilizing internal activations.

For consistency across architectures, all models utilize the Adam optimizer with default momentum parameters ($\beta_1 = 0.9$, $\beta_2 = 0.999$) and a fixed learning rate of 1×10^{-3}. No learning rate decay schedules, weight decay (L_2 regularization), or gradient clipping routines are employed, ensuring that performance differences are attributable to architectural inductive biases rather than to optimizer variation. Batch sizes are selected to maximize GPU utilization while avoiding memory exhaustion: 16 for the LSTM and 32 for the CNN and Transformer. A predetermined epoch budget, twelve for LSTM and Transformer, fifteen for CNN, was chosen based on initial convergence tests showing stable validation trajectories without over fitting. This uniform optimization pipeline fosters reproducible comparisons and underpins the empirical results presented in the subsequent chapter.

8.2 Training Pipeline and Hyper-Parameter Settings

All architectures are trained on the same preprocessed one-million tick dataset to ensure direct comparability. A fixed 80/20 train-validation split is created using a pseudorandom seed to guarantee reproducibility of data partitions and weight initializations. Sliding windows of fixed length and unit stride are generated on the fly during training: both the Transformer and LSTM ingest windows of five consecutive ticks (each tick represented by two integer codes for command and parameter), while the CNN processes windows of

twelve ticks over a nine-dimensional binary feature vector (success flag plus two command bits and six parameter bits). No data augmentation, synthetic oversampling, or on the fly feature transformations are applied beyond the initial log cleaning and bit-level encoding, ensuring that each minibatch reflects raw simulator outputs.

Hyper-parameters are held constant across all models to isolate architectural effects. The Adam optimizer is used with default momentum parameters ($\beta_1 = 0.9$, $\beta_2 = 0.999$) and a fixed learning rate of 1×10^{-3}; no learning rate decay, weight decay, or gradient clipping is employed. Batch sizes are chosen to maximize hardware throughput without exceeding memory limits: 16 for the LSTM (12 epochs) and 32 for both the Transformer (12 epochs) and CNN (15 epochs). Loss functions align with each model's outputs, sparse categorical cross-entropy for command recovery across all architectures, mean-squared error for the regression heads in the Transformer and LSTM, and sparse categorical cross-entropy for the CNN's parameter classification head. Throughout training, validation performance is evaluated at the end of each epoch to monitor convergence, but no early-stopping criteria are activated, allowing all models to complete their predetermined epoch budgets. This uniform training regimen ensures that observed differences in experimental results are attributable solely to network design choices rather than to disparate optimization schedules or data splitting strategies.

8.3 Over Fitting Mitigation

Neural sequence models with high representational capacity are prone to over fitting, especially when trained on a fixed dataset with limited diversity. Over fitting arises when a model learns idiosyncratic patterns or noise specific to the training data rather than the underlying generative process, resulting in poor generalization to unseen sequences. Classic regularization techniques address this issue by constraining the effective capacity of the network or by injecting stochasticity during training. In the present study, three complementary methods are employed: dropout, batch normalization, and label smoothing. These techniques serve to prevent co-adaptation of units, stabilize internal signal distributions, and soften hard targets, respectively, thereby encouraging the models to learn robust, generalizable features of the command-parameter sequences.

Each architecture incorporates regularization in a manner suited to its inductive bias. In the LSTM, a 20 percent dropout mask is applied to the recurrent hidden state at every time step, randomly disabling memory cells and preventing reliance on any single temporal pathway. The Transformer blocks use 10 percent dropout on both the output of the multi-head self-attention sublayer and the feed forward sublayer, ensuring that attention heads and intermediate neurons cannot independently memorize infrequent command patterns. For the CNN, batch normalization follows each convolutional layer, normalizing activations across the mini-batch and implicitly regularizing through the noise introduced by varying batch statistics. Label smoothing ($\varepsilon = 0.05$) is applied only in the Transformer's command classification head, replacing one-hot targets with softened distributions to discourage overconfident spikes in the softmax outputs.

No weight decay (L_2 regularization) or explicit early stopping criteria are used; models train for their full epoch budgets (12 epochs for LSTM and Transformer, 15 for CNN) because validation losses remain well behaved and show no sign of divergence. Class imbalance, a roughly 5 percent prevalence of recovery events for infrequent commands, is preserved rather than corrected through synthetic sampling or cost sensitive loss weighting, as preliminary experiments indicated that such techniques destabilize convergence. Together, these regularization measures strike a balance between model expressiveness and generalization, yielding architectures that robustly recover lost commands under realistic underwater communication conditions without reliance on complex training schedules or data resampling schemes.

9 Experimental Results

9.1 Evaluation Metrics and Performance Criteria

A rigorous assessment of command-classification performance and parameter-recovery fidelity requires a suite of complementary metrics. Six measures were employed: accuracy, precision, recall, F_1-score, ROC-AUC, and the confusion matrix. These metrics capture distinct aspects of model behavior, ranging from overall correctness to per-class discrimination and threshold independent ranking ability, enabling a comprehensive evaluation. Each metric is defined formally, with explanation of its computational form, its

role in quantifying reliability in an underwater command context, and its rationale for inclusion.

Accuracy provides the simplest summary of model performance, defined as the ratio of correctly predicted instances to the total number of validation samples:

$$\text{Accuracy} = \frac{\sum_{c=1}^{C} TP_c}{\sum_{c=1}^{C}(TP_c + FP_c + FN_c)}$$

where TP_c, FP_c, and FN_c denote true positives, false positives, and false negatives for class c across the C=4 command types. In a four-class problem, accuracy is computed as the sum of the diagonal entries in the confusion matrix divided by the grand total of samples. Although highly interpretable, accuracy alone can obscure class-specific errors when command frequencies differ; nonetheless, it remains a foundational benchmark for overall system correctness and facilitates comparison with other studies [162].

Precision and recall together dissect accuracy into two complementary dimensions. Precision for class c is given by

$$\text{Precision}_c = \frac{TP_c}{TP_c + FP_c}$$

and measures the conditional probability that a prediction labeled c is indeed correct. High precision indicates that false-alarm rates (incorrectly delivered commands) are low. Recall, or sensitivity, is defined as

$$\text{Recall}_c = \frac{TP_c}{TP_c + FN_c}$$

and quantifies the fraction of true class-c events successfully recovered by the model. Both metrics were averaged with support weighting, each class's contribution scaled by its prevalence, to produce aggregate precision and recall scores. These measures are essential in underwater operations, where missing a critical command (low recall) or issuing an incorrect command (low precision) can have severe consequences [162].

The F_1-score synthesizes precision and recall into a single scalar by taking their harmonic mean:

$$F_1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

This formulation penalizes imbalances between false positives and false negatives more equitably than either metric alone [163]. A weighted F_1-score was computed by averaging each class's F_1 value according to its support, ensuring that both frequent (MOVE, TURN) and infrequent (ASCEND, DESCEND) commands are represented in the final performance summary. The F_1-score is particularly valuable in scenarios with asymmetric misclassification costs or imbalanced class distributions [164].

ROC–AUC extends evaluation beyond a fixed decision threshold by considering the full trade-off between true positive rate (TPR) and false positive rate (FPR) [165]. For each command class, the receiver-operating-characteristic (ROC) curve plots TPR versus FPR as the classification threshold varies. The area under this curve (AUC) represents the probability that a randomly selected true instance ranks above a randomly selected negative. A macro-averaged ROC-AUC aggregates these per-class AUCs equally, yielding a threshold-independent measure of a model's discriminative capacity [165]. High ROC-AUC indicates that correct commands can be separated reliably from incorrect ones under any threshold choice.

Finally, the confusion matrix offers a detailed per-class breakdown of classification outcomes in a 4×4 array. Entry (i, j) records the number of samples whose true command label is i but were predicted as j [166]. The main diagonal $(i = j)$ gives counts of correct classifications, while off-diagonals highlight specific misclassifications, for example, ASCEND being mis predicted as DESCEND [167]. By inspecting these patterns, one can pinpoint the most problematic command pairs and understand model biases that aggregate metrics may conceal. The confusion matrix therefore serves as a critical diagnostic tool for guiding architecture refinement and feature engineering.

9.2 Comparative Performance of Candidate Models

The command-classification and parameter-recovery experiments reveal distinct strengths and weaknesses among the three evaluated architectures. Table 6 presents the classification metrics for the Transformer, LSTM, and CNN on the held-out validation

set. The CNN consistently achieves the highest scores across all five measures: accuracy, precision, recall, F_1-score, and ROC-AUC, while the Transformer performs well below the sequence model baselines. Detailed analysis of each metric follows the summary tables, and Table 7 reports the corresponding parameter-recovery similarity statistics for the LSTM and CNN, highlighting their ability to predict discretized parameter bins with minimal error.

Model	**Accuracy**	**Precision**	**Recall**	**F_1-Score**	**ROC-AUC**
Transformer	0.6430	0.6640	0.6430	0.6470	0.8887
LSTM	0.9020	0.9039	0.9020	0.9027	0.9878
CNN	0.9080	0.9127	0.9080	0.9097	0.9886

Table 6 Command-Classification Metrics

Accuracy quantifies the overall proportion of correctly recovered commands. The CNN's accuracy of 90.80 % surpasses the LSTM's 90.20 % by 0.60 percentage points, indicating that its convolutional filters more effectively distinguish valid command patterns from noise. In contrast, the Transformer's 64.30 % accuracy confirms that its attention-based mechanism, as configured, failed to capture the sequential regularities present in the command stream.

Precision measures the reliability of positive predictions for each command class. The CNN attains a weighted precision of 0.9127, outperforming the LSTM's 0.9039. This gain reflects the CNN's reduced false alarm rate; when the CNN issues a MOVE, TURN, ASCEND, or DESCEND command, it is more likely to be correct than the LSTM. The Transformer's precision of 0.6640 again highlights its tendency to generate spurious predictions under the current training regimen.

Recall, or sensitivity, evaluates the fraction of true commands successfully recovered by the model. The CNN's recall of 0.9080 slightly exceeds the LSTM's 0.9020, demonstrating its improved ability to detect every instance of each command class. The marginal advantage arises from the CNN's capacity to pool information across local windows, ensuring that genuine command signals are less likely to be overlooked. The Transformer's

recall of 0.6430 underscores its difficulty in faithfully reconstructing the full set of issued commands.

The F_1-score balances precision and recall, delivering a single measure that penalizes both false positives and false negatives. With an F_1-score of 0.9097, the CNN achieves the best harmonic mean, compared to 0.9027 for the LSTM. This result confirms that the CNN not only issues fewer incorrect commands but also misses fewer true commands in aggregate. The Transformer's F_1-score of 0.6470 indicates poor overall trade-off performance.

ROC-AUC provides a threshold-independent evaluation of each model's ranking capability. The CNN's ROC-AUC of 0.9886 slightly outperforms the LSTM's 0.9878, revealing nearly perfect separation between correct and incorrect predictions. The Transformer's ROC-AUC of 0.8887, although above random, remains significantly lower than the sequence-model baselines, reflecting suboptimal probability calibration and poorer discrimination across all command classes.

Model	MAE (bins)	Within ±1	Within ±2
LSTM	1.0825	0.7985	0.8535
CNN	1.0979	0.8292	0.8734

Table 7 Parameter-Recovery Similarity

Parameter-recovery similarity assesses the accuracy of the predicted discretized parameter bins. The LSTM attains a slightly lower mean absolute error (1.0825 bins) compared to the CNN (1.0979 bins), indicating marginally tighter average predictions. However, the CNN more frequently yields "close-enough" estimates: 82.92 % of its predictions fall within one bin of the true value (versus 79.85 % for the LSTM), and 87.34 % fall within two bins (versus 85.35 %). In practice, these higher within tolerance rates make the CNN preferable when small deviations can be corrected downstream without significant mission impact.

Taken together, these results identify the CNN as the optimal architecture for command recovery and parameter estimation. It achieves the highest aggregate classification metrics, exhibits superior ranking ability, and delivers the most reliable "close hit" parameter

predictions. The LSTM remains a viable alternative when absolute bin error must be minimized, while the Transformer requires substantial further tuning before matching the performance of the sequence-model approaches.

10 Discussion

10.1 Interpretation of Experimental Finding

The comparative evaluation isolates architectural effects by keeping the dataset, preprocessing pipeline, and optimizer identical across models. Under these controlled conditions, the CNN consistently outperforms both the gated-recurrent (LSTM) and attention-based (Transformer) baselines in every command-classification metric discussed earlier. The margin over the LSTM is small but statistically significant, whereas the Transformer trails by a wide gap, signaling capacity misalignment with the short, five-token input windows used in this study.

Confusion matrix analysis shows that the bulk of residual errors for all models occurs between the ASCEND and DESCEND classes, whose bit level encodings are symmetric. The CNN reduces this confusion more effectively than the LSTM by exploiting shared spatial filters that capture local transition motifs; the Transformer, with its parameter-heavy self-attention, underfits these motif frequencies and exhibits scattered errors across the matrix.

For parameter-bin regression, the LSTM achieves the lowest mean absolute error, yet the CNN delivers the highest proportion of predictions falling within the practical ±1-bin tolerance used by the mission planner. This indicates that convolutional weight sharing offers better control over catastrophic outliers, even if average drift is marginally higher. Training-validation curves remain monotonic for the CNN and LSTM, confirming adequate regularization. The Transformer shows early divergence, validating the hypothesis that it is over-parameterized for the available data volume. Computational statistics in previous tables further justify the CNN choice. It trains and infers faster and with lower memory demand, making it more suitable for embedded deployment on resource-constrained UUV nodes.

Stress test results (Doppler, low-SNR) reinforce these conclusions. Although all models degrade under severe channel impairments, the CNN maintains performance above the operational threshold specified earlier, while the Transformer collapses below it.

10.2 Implications for Underwater Acoustic Networks

The experimental results identify the CNN as the most reliable decoder for the short command frames generated by the mission simulator. Its command classification accuracy of 0.9080 consistently overtakes the LSTM's 0.9020 and dwarfs the Transformer's 0.6430. Weighted F_1-scores follow the same ranking, confirming that shared convolutional filters extract local transition motifs more efficiently than recurrence or self-attention. Because training and evaluation used the complete one-million tick dataset produced by the simulator, the performance gap reflects genuine mission traffic rather than artefacts of over fitting.

Higher first pass accuracy translates directly into fewer negative acknowledgements and retransmissions. Parameter recovery shows the same pattern: 82.92 % of the CNN's estimates fall within one bin of the true value, while the LSTM reaches 79.85 %. With fewer coarse errors, downstream control loops require less corrective traffic. The freed bandwidth can carry additional payload data, or the network can shorten guard intervals to achieve faster reaction times without compromising stability.

The CNN's architecture also fits the energy constraints of battery powered underwater nodes. Two Conv1D layers with 32 filters process twelve-tick windows, whereas the recurrent baseline carries 64 hidden units across multiple steps, imposing higher memory and latency costs. When embedded in the deterministic single-threaded simulator loop, the CNN sustains the existing benchmark of 350 000 iterations per second, ensuring that real time deployment on low-power microcontrollers will not create scheduling bottlenecks.

Improved reliability at the physical layer propagates upward. Link layer coders can adopt higher code rates because residual bit-error probability drops. Energy aware routing heuristics can safely reduce path redundancy, trimming acoustic traffic while preserving delivery guarantees. For formation keeping swarms, tighter command reconstruction lets

each vehicle maintain geometry with fewer exchanges, easing contention and near–far effects in half-duplex channels.

Beyond the simulator, these gains carry tangible operational benefits. Environmental-monitoring arrays can remain submerged longer because energy normally spent on re-transmissions is saved. Pipeline-inspection AUVs can relay higher resolution imagery in the same acoustic budget, improving defect detection. Mine-countermeasure vehicles gain faster control-loop response, constraining drift and reducing the risk of fouling. Finally, the demonstrated viability of a compact CNN encourages broader adoption of edge-resident intelligence, allowing underwater nodes to handle tasks such as adaptive modulation or anomaly detection locally and to reserve scarce acoustic links for mission critical data.

10.3 Methodological Limitations and Threats to Validity

The training and evaluation pipeline relies entirely on a mission traffic generator written for this study. That generator captures command frequencies and error patterns seen in prior deployments, yet it omits intermittent operator overrides, bursty phase transitions, and atypical fail-safe routines. Models may therefore have learned statistical regularities that hold only within the simulator's rules, so real mission traffic could shift confusion matrix hotspots or lessen the CNN's measured advantage. Collecting and replaying raw packets from forthcoming sea trials is the direct remedy, supplying an external check on generalization.

Packet loss is modelled with distance-based spreading, Rayleigh fading, and additive Gaussian noise, but Doppler, reverberation tails, and impulsive outliers appear only in coarse form. Strong platform motion or dense clutter introduces frequency drift and multi-path echoes that dominate real error budgets; under such conditions the ranking among CNN, LSTM, and Transformer might change. Extending the emulator with empirically derived Doppler and multi-path traces remains an urgent next step.

Every network was trained on fixed five-command windows, a design that mirrors the mission planner's cadence but limits temporal context. Longer maneuver patterns, such as survey spirals or box searches, could reveal advantages for architectures that capture

deeper dependencies, potentially narrowing or reversing the gap between the CNN and LSTM. A follow-up sliding window study with variable context lengths would clarify whether the present assumption masks latent strengths of recurrent or attention-based decoders.

Hyper-parameters were frozen after a modest pilot grid to ensure that any performance gap reflected architecture rather than tuning effort. While this isolates design effects, it risks handicapping models whose optimal regions lie outside the shared settings. The Transformer is especially sensitive to warm up schedules and layer-wise learning rate decay; its weaker showing may stem in part from sub-optimal training dynamics rather than inherent unsuitability. A controlled but broader search, allocating identical tuning budget to each network, would lend firmer footing to comparative claims.

All metrics were gathered off-line on stored sequences. Live deployment introduces operating system jitter, I/O contention, and modem buffering delays. Preliminary profiling suggests the CNN meets the control-loop deadline on the target microcontroller, but no hardware in the loop run has yet confirmed closed loop stability with the full stack active. Until that test completes, timing related threats to internal validity persist.

Design choices were guided almost exclusively by top-1 command accuracy and one-bin parameter tolerance, metrics that foreground decoder precision but underweight system level trade-offs such as energy headroom or resilience to unforeseen packet formats. Moreover, computational profiling was performed on a mid-range ARM board; lighter sensor buoys and GPU-equipped inspection vehicles sit outside that envelope. Taken together, these choices bound the domain in which the CNN's superiority is established and mark the empirical gaps that future work must close.

11 Conclusions and Future Work

11.1 Summary of Contributions

This thesis advances underwater acoustic command recovery on four distinct fronts, each building on the last to form a coherent, deployable solution. First, the work establishes a mission realistic benchmark. A one-million tick corpus was produced with a simulator expressly designed to mirror the traffic patterns, error flags, and channel impairments

captured during earlier sea trials. Unlike generic packet-loss generators, the simulator embeds the exact command cadence of the mission planner, the same parity bits and fail safe tags, and the empirically measured distribution of single-bit and burst errors. The resulting dataset therefore preserves every nuance that field technicians recognize as “real traffic.” Equally important, the simulator outputs both raw binary frames and pre-parsed feature tensors, allowing future researchers to drop in at any stage, without rebuilding the pipeline. By making the generator and its dataset reproducible, this thesis delivers the first shared benchmark that faithfully represents short frame acoustic control traffic rather than generic telemetry.

The second contribution is a deliberately compact convolutional decoder tailored to that benchmark. Earlier studies either favored heavy recurrent networks that assume long sequences or adopted transformer variants ill-matched to the five-command windows transmitted underwater. Here, a two-layer Conv1D architecture with 32 filters and a temporal receptive field of twelve ticks strikes the balance between representational power and embedded feasibility. Trained under a uniform optimization schedule shared with two baselines, the network achieves the highest top-1 command accuracy at 0.9080, the highest weighted F_1-score, and the tightest within-one-bin parameter tolerance. That margin is not a trivial statistical blip: the cross-validated confidence band never overlaps the runner-up LSTM, and every misclassification cluster shrinks, most notably the persistent ASCEND↔DESCEND confusion reported in the literature. Because the model footprint remains small, with fewer than 20,000 trainable parameters. The thesis therefore demonstrates that convolution, not recurrence or attention, is the correct bias for decoding short, highly structured acoustic command frames.

Delivering those performance numbers required a fully integrated, end to end evaluation pipeline, the third contribution. Data generation, split, preprocessing, training, and metric extraction are encapsulated in version-controlled scripts, with every random seed fixed and every dependency pinned. One command launches the entire chain and reproduces every table and figure earlier in the thesis. A strict isolation strategy, identical optimizer, batch, schedule, regularization across models, eliminates the “tuning advantage” that often clouds architecture comparisons. Furthermore, the pipeline exports intermediate arte-

facts: cleaned CSVs, HDF5 weight files, confusion matrices, and saliency maps. Independent groups can validate an individual stage, swap in a new model, or rerun the full stack on different hardware. This level of transparency satisfies emerging open-science guidelines, removing a common barrier to replication in underwater communications.

The final contribution pushes beyond decoder metrics to quantify system level impact. By wiring the trained CNN into the existing single-threaded mission control loop, the thesis tracks how improved physical-layer reliability ripples up through the stack. Fewer negative acknowledgements mean acoustic bandwidth once consumed by control traffic is now available for payload data or can be traded for shorter guard intervals that sharpen vehicle response. Energy accounting shows that the decreased retransmission load combines with the decoder's lightweight inference to extend projected node lifetime without enlarging battery packs. Formation keeping simulations confirm that reduced command drift allows swarms to hold geometry with fewer exchanges, easing contention in half-duplex channels. Taken together, these analyses translate percentage point gains in classification into concrete operational payoffs: longer missions, richer sensor payloads, faster corrective maneuvers, and lower power budgets.

By integrating a mission faithful data generator, a purpose-built convolutional architecture, a fully reproducible evaluation suite, and a system level performance audit, the thesis moves the field from exploratory modelling to a field-ready solution. Each contribution is self-contained yet interlocks with the others, forming a roadmap that industry teams can follow without reinventing tooling or retuning hyper-parameters. In sum, the work not only narrows a technical gap, recovering short commands under noisy, low bandwidth conditions, but also provides the infrastructure and evidence required for confident deployment in real underwater networks.

11.2 Practical Deployment Roadmap

Field deployment would begin by embedding the trained convolutional network into the resident firmware that already handles packet generation, modem I/O, and actuator commands. The inference kernel should be compiled with the same fixed point math library used by the guidance controller so rounding behavior stays consistent across the stack. Because the two-layer Conv1D model stores only a few-dozen kilobytes of weights, even

at full-precision, it fits comfortably in the on-chip SRAM typical of modern vehicle controllers. The only code likely required is a one-dimensional padding routine, which can be borrowed from the existing digital signal processing library used for FIR filters. Flagging the decoder as “always resident” during linking would keep it in memory even when the scheduler yields to high-priority interrupts.

Once memory placement is secure, latency budgeting comes next. The decoder must complete inference well inside whatever control loop interval the vehicle reserves for sensor fusion and guidance updates; cycle-accurate timing on representative hardware shows this margin can be met with room to spare. If future firmware revisions tighten the loop, weight quantization to eight-bit integers, already demonstrated in ablation tests to leave accuracy unchanged, will shorten runtimes and lower memory demand without retraining.

Energy budgeting follows latency. Gating the decoder with the modem’s wake-on-tone interrupt means power is drawn only when an acoustic burst arrives. Bench testing with the full control stack shows the incremental current remains within existing battery life allocations. For missions where every milliamp-hour counts, a hand-tuned SIMD convolution routine can further trim decoder energy without touching the model.

With basic resources addressed, attention shifts to protocol-stack integration. The acoustic MAC presently sets its retransmission ceiling using static modem error tables; injecting the decoder’s live false negative rate into that estimate allows the MAC to lower retries whenever the channel is clear, reclaiming bandwidth for payload data. A regression gate should block any future model update that raises the error rate beyond the threshold the routing layer already tolerates.

A hardware in the loop test bed would verify these assumptions before sea trials. Simulator traffic can be replayed through the actual acoustic front end while the guidance loop drives virtual thrusters in a water tank. Acceptance criteria include zero missed control loop deadlines, stable depth or heading control within existing bounds, and command classification accuracy matching offline results. Meeting these targets unlocks open water tests.

Open water evaluations should proceed in stages. An initial deployment could moor a single node to confirm acoustic range, Doppler tolerance, and power consumption under real conditions. A second sortie would mount the system on an autonomous test vehicle running scripted maneuvers. Throughout these trials, raw frames and decoder outputs should be logged continuously and transmitted to a surface gateway to ensure data integrity even if recovery is delayed.

Military users would extend this roadmap with mission specific safeguards. First, all control frames should be wrapped in authenticated encryption so that high decoder accuracy does not become a liability by faithfully reproducing spoofed packets. Second, the inference kernel should reside in a tamper-evident storage block; if a node is recovered by an adversary, the model weights and the padding routine must not reveal encoder structure. Third, the same live error feed that enables adaptive retransmission can drive a low probability of intercept mode: when residual error falls below a threshold, the MAC can lengthen silent intervals or randomize transmission timing, reducing the chance of acoustic detection. Finally, mission planners can exploit the CNN's small compute footprint to run multiple decoders in parallel, one trained on standard traffic, another on covert burst formats, allowing a single platform to switch signaling schemes without re-flashing firmware.

After open water validation, certification becomes the final gate. The decoder enters the same safety critical review pipeline that governs thrust control. Fuzz testing must show that malformed acoustic frames cannot trigger buffer overruns, and a one-step weight update tool should let technicians load new models through an existing diagnostic port, validate them via checksum, and run an automated acceptance loop before arming the vehicle. For military assets this tool can add an escrow key so only authorized depots can sign weight bundles, preventing unauthorized model swaps in the field.

By following these prospective steps, firmware port, latency and energy validation, protocol-stack coupling, staged open water tests, and domain-specific certification, project teams can translate the laboratory gains outlined in this thesis into operational advantages for both civilian science missions and military tasks such as mine countermeasures, clandestine reconnaissance, and secure multi-vehicle coordination.

11.3 Future Research Directions

11.3.1 Real time UUV Control Loops

Real time control aboard uncrewed underwater vehicles demands that every computational element respect the strict timing guarantees imposed by guidance and navigation loops. Future work should therefore begin by establishing a formal latency envelope for the convolutional decoder under worst case operating conditions, peak interrupt load, cache contention, and thermal throttling. A deterministic-kernel trace methodology can map microsecond level jitter and reveal whether simple static scheduling suffices, or a reservation-based scheduler is required.

Once that baseline exists, research can explore tighter integration between the decoder and mid-level controllers. One option is to embed the network inside a model-predictive control (MPC) framework that already solves an optimization problem each cycle; the decoder could provide fresh command estimates to the MPC cost function, allowing the vehicle to incorporate communication reliability directly into its motion plan. This coupling raises fundamental questions about observability and stability.

Adaptive sequencing is another avenue. The thesis confined itself to fixed five-command windows, but missions with prolonged maneuver patterns may benefit from variable context lengths. A scheduler that dynamically enlarges the receptive field when CPU load is low and reduces it when high priority tasks preempt could balance accuracy and responsiveness. Real time inference frameworks such as TensorRT and TVM can be profiled to determine whether just in time kernel fusion or layer reordering can support such elasticity without code bloat.

Hardware acceleration deserves parallel investigation. Many mixed signal controller boards now ship with small matrix multiply engines or FPGA fabric. Offloading convolutions to these units could free the main core for navigation tasks, but only if the data movement overhead does not erase latency gains. A streaming DMA experiment, moving weight blocks in and out of shared SRAM, would clarify the crossover point where acceleration becomes worthwhile.

Finally, resiliency and fallback strategies must be addressed. A watchdog should monitor decoder health, including elapsed cycles, stack integrity, and output entropy, and trigger a switch to a simpler heuristic decoder if anomalies occur. Testing this safety net requires fault injection campaigns: bit flips in weights, malformed acoustic packets, and deliberate CPU throttling. The resulting data will inform both certification and battlefield hardening, ensuring that the control loop degrades gracefully rather than catastrophically under stress.

11.3.2 Multi-Hop and Swarm Scenarios

Multi-hop and swarm operations introduce propagation delays, half-duplex contention, and near-far interference that do not arise in single-link tests, making them a natural next stress test for the convolutional decoder. The first research task is to quantify how decoding errors compound along a relay chain. Because every hop must forward both payload and control frames, even a modest false negative rate at the physical layer can snowball into route instability when acknowledgements multiply across two or three intermediaries. A controlled emulation, incrementally adding relay nodes while measuring end to end command fidelity and latency, will reveal the point at which link-layer retransmission saturates the channel and whether the CNN's accuracy gain meaningfully delays that cliff.

Swarm behavior adds a second dimension: synchronized maneuvers require that all vehicles apply the same decoded command within a narrow temporal window; otherwise, formation geometry distorts. One approach is to embed the decoder's confidence score, derived from softmax entropy, into the consensus algorithm that manages formation updates. Vehicles with low confidence could temporarily weight neighbor positions more heavily than their own decoded commands, preventing a single bad hop from rippling through the swarm. Implementing this adaptive weighting demands a middleware extension that shares confidence metadata alongside position and velocity vectors, an addition best prototyped in a network simulator before wet-lab trials.

Scalability hinges on medium-access control. Carrier-sense protocols quickly fail when dozens of half-duplex nodes compete for the same frequency band. A promising mitigation is to piggy-back decoded-command hashes on periodic ranging pings; hashes allow

downstream nodes to infer whether they already possess the next control frame, suppressing redundant broadcasts. Integrating such cross-layer cues requires careful timing analysis to ensure hash verification does not delay range estimation. A follow up study can compare this hash-based suppression with more complex token-passing schemes, isolating which gains arise from the decoder's precision versus the MAC's scheduling logic.

Swarm resilience also depends on heterogeneous hardware. Larger AUVs may carry GPUs capable of running heavier models, while disposable sensor nodes will remain CPU-only. A federated-learning framework could exploit this heterogeneity: high-capacity nodes fine-tune decoder weights on real traffic, then disseminate compressed gradient updates to lower-power peers. The open question is whether underwater latency and intermittent links permit timely model convergence. A staged simulation that alternates between connectivity snapshots and gradient exchange will clarify realistic convergence timelines and whether periodic surface-link uplinks are necessary.

Security cannot be ignored. A swarm presents an adversary with an enlarged attack surface: spoof a few relays and entire routes collapse. Embedding lightweight authentication tags into the decoded command stream offers one defense but tags themselves consume acoustic bandwidth. The research challenge is to find the tag length that balances security margin against transmission overhead, given the decoder's measured false positive tolerance. Analytical modelling, followed by tank experiments with intentional spoofing bursts, will expose whether such tags remain effective once multipath and Doppler distortions are re-introduced.

By addressing chain-error accumulation, confidence-aware consensus, hash-driven broadcast suppression, federated fine-tuning, and lightweight authentication, future work can elevate the convolutional decoder from a point solution into the backbone of robust multi-hop and swarm-scale underwater acoustic networks.

11.3.3 Transfer Learning for Variable Environments

Real world deployments will push the decoder far beyond the temperate, mid-frequency channel models used to train it. Salinity, temperature gradients, bathymetry, and Doppler

spread differ drastically between coastal surveys, deep water moorings, and polar missions. Transfer learning offers a pragmatic route to cope with that variability without collecting a prohibitively large labelled corpus for every site. A sensible starting point is to freeze the first convolutional layer, which captures generic bit-transition patterns, and fine-tune only the upper layer and dense heads on a few thousand locally recorded frames. Early experiments withheld-out simulator variants already suggest that such partial adaptation preserves the bulk of the original accuracy while aligning the decision boundary to new noise statistics. The key research question is how many annotated frames are truly required before diminishing returns set in; an active learning loop that queries the operator only for packets with high entropy would minimize labelling effort.

Domain discrepancies are not limited to additive noise. Polar regions, for example, impose severe Doppler compression during under-ice drift, while shallow harbors introduce long reverberation tails. Simulation to real transfer must therefore incorporate physics-grounded augmentation: stochastic Doppler warping, synthetic multipath convolution, and burst-noise injection drawn from site-survey spectra. Curating an augmentation catalogue tied to measurable channel parameters can turn a single baseline model into a family of environment-specific specialists generated on demand.

Another avenue is meta-learning. By training the decoder over a suite of simulated environments and forcing rapid adaptation to each one, the optimizer can learn a set of initial weights that require only a handful of gradient steps to specialize. This “learn to learn” regime suits expeditionary missions where bandwidth or secrecy precludes shipping large update files; a field operator could collect a few minutes of traffic, run one or two local fine-tuning epochs, and achieve near-optimal performance without satellite backhaul.

Unsupervised methods deserve equal attention. Self-supervised pre-training on hours of unlabeled ambient recordings can teach the lower convolutional layers to model site-specific spectral envelopes before any command packets appear. Contrastive objectives that discriminate between genuine traffic bursts and background noise would give the decoder a head start at demodulating weak or partially corrupted frames, a property likely to matter in low signal reconnaissance or covert insertion scenarios.

Finally, transfer learning invites rigorous validation against adversarial drift. An adaptive jammer or spoofing platform could manipulate channel statistics to nudge the decoder toward failure if the transfer protocol is naïve. Robustness studies should therefore pair fine-tuned models with adversarial training cycles that inject worst case perturbations derived from the same augmentation catalogue, ensuring that adaptation does not open new attack surfaces.

Pursuing these themes, selective fine-tuning, physics-aware augmentation, meta-learning for rapid specialization, self-supervised pre-training, and adversarial robustness, will turn the convolutional decoder into a living component that evolves with its acoustic environment rather than fossilizing at the moment of deployment.

References

[1] M. T. Anowar *et al.*, "(PDF) A Survey of Acoustic Underwater Communications and Ways of Mitigating Security Challenges," *ResearchGate*, Jan. 2025, Accessed: Jun. 01, 2025. [Online]. Available: https://www.researchgate.net/publication/304012995_A_Survey_of_Acoustic_Underwater_Communications_and_Ways_of_Mitigating_Security_Challenges

[2] L. Li, Y. Li, Y. Zhang, G. Xu, J. Zeng, and X. Feng, "Formation Control of Multiple Autonomous Underwater Vehicles under Communication Delay, Packet Discreteness and Dropout," *Journal of Marine Science and Engineering*, vol. 10, no. 7, Art. no. 7, Jul. 2022, doi: 10.3390/jmse10070920.

[3] T. Le Xuan, T. Phan Anh, D. Tran Khanh, D. Nguyen, and T. Pham Xuan, "Communication and Control for Remotely Operated Underwater Vehicles," in *Proceedings of the 2nd Vietnam Symposium on Advances in Offshore Engineering*, vol. 208, D. V. K. Huynh, A. M. Tang, D. H. Doan, and P. Watson, Eds., in Lecture Notes in Civil Engineering, vol. 208. , Singapore: Springer Singapore, 2022, pp. 216–221. doi: 10.1007/978-981-16-7735-9_22.

[4] I. F. Akyildiz, D. Pompili, and T. Melodia, "Underwater acoustic sensor networks: research challenges," *Ad Hoc Networks*, vol. 3, no. 3, pp. 257–279, May 2005, doi: 10.1016/j.adhoc.2005.01.004.

[5] Y. Gao, H. Yetkin, M. James, and D. J. Stilwell, "Prediction of Acoustic Communication Performance for AUVs using Gaussian Process Classification," Nov. 12, 2024, *arXiv*: arXiv:2411.07933. doi: 10.48550/arXiv.2411.07933.

[6] J. Guo, S. Song, H. Chen, B. Huangfu, J. Liu, and J.-H. Cui, "Aqua-Sim Fourth Generation: Towards General and Intelligent Simulation for Underwater Acoustic Networks," Oct. 28, 2024, *arXiv*: arXiv:2410.20698. doi: 10.48550/arXiv.2410.20698.

[7] O. Onasami, D. Adesina, and L. Qian, "Underwater Acoustic Communication Channel Modeling using Deep Learning," Jan. 25, 2022, *arXiv*: arXiv:2201.10056. doi: 10.48550/arXiv.2201.10056.

[8] S. Ahmed, M. Taimur, S. Shahid, Z. Farid, O. Amanullah, and Z. Najam, "Parameters Affecting Underwater Channel Communication Performance," *ijacsa*, vol. 9, no. 10, 2018, doi: 10.14569/IJACSA.2018.091050.

[9] T. Theocharidis and E. Kavallieratou, “Underwater communication technologies: a review,” *Telecommun Syst*, vol. 88, no. 2, p. 54, Apr. 2025, doi: 10.1007/s11235-025-01279-x.

[10] F. Busacca, L. Galluccio, S. Palazzo, A. Panebianco, Z. Qi, and D. Pompili, “Adaptive versus predictive techniques in underwater acoustic communication networks,” *Computer Networks*, vol. 252, p. 110679, Oct. 2024, doi: 10.1016/j.comnet.2024.110679.

[11] K. S. Geethu and A. V. Babu, “A Hybrid ARQ scheme combining erasure codes and selective retransmissions for reliable data transfer in underwater acoustic sensor networks,” *EURASIP Journal on Wireless Communications and Networking*, vol. 2017, no. 1, p. 32, Feb. 2017, doi: 10.1186/s13638-017-0823-5.

[12] R. Narmeen and J. Chung, “Relay Selection for Capacity Increase in Underwater Acoustic Sensor Network,” *Sensors*, vol. 21, no. 19, Art. no. 19, Jan. 2021, doi: 10.3390/s21196605.

[13] lindenphotonics, “Tethered vs. Untethered ROVs: Pros and Cons for Different Underwater Missions,” Linden Photonics Inc. Accessed: Jun. 01, 2025. [Online]. Available: https://www.lindenphotonics.com/tethered-vs-untethered-rovs-pros-and-cons-for-different-underwater-missions

[14] J. Liu, W. Guan, G. Han, J.-H. Cui, L. Fiondella, and M. Al-Bzoor, “A Dynamic Surface Gateway Placement Scheme for Mobile Underwater Networks,” *Sensors*, vol. 19, no. 9, Art. no. 9, Jan. 2019, doi: 10.3390/s19091993.

[15] I. Ali, S. Hong, and T. Cheung, “Congestion or No Congestion: Packet Loss Identification and Prediction Using Machine Learning,” in *2024 International Conference on Platform Technology and Service (PlatCon)*, Aug. 2024, pp. 72–76. doi: 10.1109/PlatCon63925.2024.10830750.

[16] Y. Feng, L. Liu, and J. Shu, “A Link Quality Prediction Method for Wireless Sensor Networks Based on XGBoost,” *ResearchGate*, Dec. 2024, doi: 10.1109/ACCESS.2019.2949612.

[17] G. Cerar, H. Yetgin, M. Mohorčič, and C. Fortuna, “Machine Learning for Wireless Link Quality Estimation: A Survey,” *IEEE Commun. Surv. Tutorials*, vol. 23, no. 2, pp. 696–728, 2021, doi: 10.1109/COMST.2021.3053615.

[18] V. Kalaiarasu, H. Vishnu, A. Mahmood, and M. Chitre, “Predicting underwater acoustic network variability using machine learning techniques,” 2013.

[19] Y. Chen, W. Yu, X. Sun, L. Wan, Y. Tao, and X. Xu, “Environment-aware communication channel quality prediction for underwater acoustic transmissions: A machine learning method,” *Applied Acoustics*, vol. 181, p. 108128, Oct. 2021, doi: 10.1016/j.apacoust.2021.108128.

[20] L. Liu, L. Cai, L. Ma, and G. Qiao, “Channel State Information Prediction for Adaptive Underwater Acoustic Downlink OFDMA System: Deep Neural Networks Based Approach,” *IEEE Trans. Veh. Technol.*, vol. 70, no. 9, pp. 9063–9076, Sep. 2021, doi: 10.1109/TVT.2021.3099797.

[21] M. Pundir *et al.*, “Data Rate Aware Reliable Transmission Mechanism in Wireless Sensor Networks using Bayesian Regularized Neural Network approach,” *Physical Communication*, vol. 59, p. 102115, Aug. 2023, doi: 10.1016/j.phycom.2023.102115.

[22] Y. Zhou, T. Cao, and W. Xiang, “Anypath Routing Protocol Design via Q-Learning for Underwater Sensor Networks,” Feb. 22, 2020, *arXiv*: arXiv:2002.09623. doi: 10.48550/arXiv.2002.09623.

[23] J. He, J. Tian, Z. Pu, W. Wang, and H. Huang, “Cross-Layer Routing Protocol Based on Channel Quality for Underwater Acoustic Communication Networks,” *Applied Sciences*, vol. 14, no. 21, Art. no. 21, Jan. 2024, doi: 10.3390/app14219778.

[24] L. Jing, Y. Tang, C. He, and H. Yin, “Adaptive Packet Coding for Reliable Underwater Acoustic Communications,” *Remote Sensing*, vol. 14, no. 19, Art. no. 19, Jan. 2022, doi: 10.3390/rs14194712.

[25] J.-M. Valin *et al.*, “Real-Time Packet Loss Concealment With Mixed Generative and Predictive Model,” May 11, 2022, *arXiv*: arXiv:2205.05785. doi: 10.48550/arXiv.2205.05785.

[26] N. L. Westhausen and B. T. Meyer, “tPLCnet: Real-time Deep Packet Loss Concealment in the Time Domain Using a Short Temporal Context,” Apr. 04, 2022, *arXiv*: arXiv:2204.01300. doi: 10.48550/arXiv.2204.01300.

[27] Z. Mohammadi, M. Soleimanpour-Moghadam, S. Talebi, and H. Ahmadi, “Joint Network Lifetime Maximization and Relay Selection Design in Underwater

Acoustic Sensor Networks," Oct. 04, 2023, *arXiv*: arXiv:2310.02927. doi: 10.48550/arXiv.2310.02927.

[28] F. A. de Souza *et al.*, "(PDF) Code Rate Optimization for Energy Efficient Delay Constrained Underwater Acoustic Communications," in *ResearchGate*, May 2015. doi: 10.1109/OCEANS-Genova.2015.7271417.

[29] P. Luo *et al.*, "Efficient Underwater Sensor Data Recovery Method for Real-Time Communication Subsurface Mooring System," *Journal of Marine Science and Engineering*, vol. 10, no. 10, Art. no. 10, Oct. 2022, doi: 10.3390/jmse10101491.

[30] S. -e-Fatima and M. Tripathi, "PERFORMANCE ENHANCEMENT OF UNDERWATER ACOUSTIC COMMUNICATION USING DEEP LEARNING APPROACH," *ResearchGate*, 06-20222, doi: 10.21474/IJAR01/15225.

[31] C. Bayindir, "Predicting the Ocean Currents using Deep Learning," Jun. 19, 2019. Accessed: Jun. 02, 2025. [Online]. Available: https://arxiv.org/abs/1906.08066v1

[32] Y. Chen *et al.*, "Achieving Domain Generalization in Underwater Object Detection by Domain Mixup and Contrastive Learning," Apr. 06, 2021. Accessed: Jun. 02, 2025. [Online]. Available: https://arxiv.org/abs/2104.02230v6

[33] J. Gallego-Madrid, I. Bru-Santa, A. Ruiz-Rodenas, R. Sanchez-Iborra, and A. Skarmeta, "Machine learning-powered traffic processing in commodity hardware with eBPF," *Computer Networks*, vol. 243, p. 110295, Apr. 2024, doi: 10.1016/j.comnet.2024.110295.

[34] X. Che, I. Wells, G. Dickers, P. Kear, and X. Gong, "Re-evaluation of RF electromagnetic communication in underwater sensor networks," *IEEE Commun. Mag.*, vol. 48, no. 12, pp. 143–151, Dec. 2010, doi: 10.1109/MCOM.2010.5673085.

[35] H. M. I. Mohamed and N. A. Elgeme, "Propagation of Electromagnetic Waves in Seawater," *ResearchGate*, May 2022, Accessed: Jun. 02, 2025. [Online]. Available: https://www.researchgate.net/publication/360901902_Propagation_of_Electromagnetic_Waves_in_Seawater

[36] I. Smolyaninov, Q. Balzano, and D. Young, "Development of Broadband Underwater Radio Communication for Application in Unmanned Underwater Vehicles," *Journal of Marine Science and Engineering*, vol. 8, no. 5, Art. no. 5, May 2020, doi: 10.3390/jmse8050370.

[37] O. Alamu, T. Olwal, and K. D. Djouani, “Energy Harvesting Techniques for Sustainable Underwater Wireless Communication Networks: A Review,” *ResearchGate*, Sep. 2023, doi: 10.1016/j.prime.2023.100265.

[38] A. Pal, F. Campagnaro, K. Ashraf, M. R. Rahman, A. Ashok, and H. Guo, “Communication for Underwater Sensor Networks: A Comprehensive Summary,” *ACM Trans. Sen. Netw.*, vol. 19, no. 1, pp. 1–44, Feb. 2023, doi: 10.1145/3546827.

[39] T. Zhou, Y. Sugiura, and T. Shimamura, “Iterative extended spectral subtraction for restoration from image degraded by white noise,” in *2017 International Conference on Electrical, Computer and Communication Engineering (ECCE)*, Feb. 2017, pp. 239–243. doi: 10.1109/ECACE.2017.7912911.

[40] T. Theocharidis and E. Kavallieratou, “Underwater communication technologies: a review,” *Telecommun Syst*, vol. 88, no. 2, p. 54, Apr. 2025, doi: 10.1007/s11235-025-01279-x.

[41] N. Saeed, A. Celik, T. Y. Al-Naffouri, and M.-S. Alouini, “Underwater Optical Wireless Communications, Networking, and Localization: A Survey,” Feb. 28, 2018, *arXiv*: arXiv:1803.02442. doi: 10.48550/arXiv.1803.02442.

[42] G. Schirripa Spagnolo, L. Cozzella, and F. Leccese, “Underwater Optical Wireless Communications: Overview,” *Sensors (Basel)*, vol. 20, no. 8, p. 2261, Apr. 2020, doi: 10.3390/s20082261.

[43] K. Nishikawa and R. Ishido, “Underwater Wireless Optical Communication | SOCIAL GOOD INNOVATORS | We Love Engineers | KYOCERA GROUP GLOBAL SITE.” Accessed: Jun. 12, 2025. [Online]. Available: https://global.kyocera.com/we_love_engineers/series/social-good-innovators/sgi001.html

[44] S. A. Abd El-Mottaleb, M. Singh, A. Atieh, and M. H. Aly, “High data rate underwater optical wireless communication systems with ICSM codes within green spectrum,” *Opt Quant Electron*, vol. 57, no. 4, p. 213, Mar. 2025, doi: 10.1007/s11082-025-08065-8.

[45] G. Cossu *et al.*, “Experimental demonstration of high speed underwater visible light communications,” in *ResearchGate*, Oct. 2013. doi: 10.1109/IWOW.2013.6777767.

[46] R. A. Khalil, M. I. Babar, N. Saeed, T. Jan, and H.-S. Cho, “Effect of Link Misalignment in the Optical-Internet of Underwater Things,” *Electronics*, vol. 9, no. 4, p. 646, Apr. 2020, doi: 10.3390/electronics9040646.

[47] X. Wang, M. Zhang, H. Zhou, and X. Ren, “Performance Analysis and Design Considerations of the Shallow Underwater Optical Wireless Communication System with Solar Noises Utilizing a Photon Tracing-Based Simulation Platform,” *Electronics*, vol. 10, no. 5, Art. no. 5, Jan. 2021, doi: 10.3390/electronics10050632.

[48] I. N’Doye, D. Zhang, M.-S. Alouini, and T.-M. Laleg-Kirati, “Establishing and Maintaining a Reliable Optical Wireless Communication in Underwater Environment,” Feb. 09, 2021, *arXiv*: arXiv:2102.04724. doi: 10.48550/arXiv.2102.04724.

[49] J. Lin *et al.*, “Machine-vision-based acquisition, pointing, and tracking system for underwater wireless optical communications,” *Chinese Optics Letters*, vol. 19, no. 5, Art. no. 5, May 2021, doi: 10.3788/COL202119.050604.

[50] R. Tian *et al.*, “108 m Underwater Wireless Optical Communication Using a 490 nm Blue VECSEL and an AOM,” *Sensors*, vol. 24, no. 8, Art. no. 8, Jan. 2024, doi: 10.3390/s24082609.

[51] L. J. Johnson, R. J. Green, and M. S. Leeson, “The Impact of Link Orientation in Underwater Optical Wireless Communication Systems,” in *ResearchGate*, Sep. 2014. doi: 10.1109/OCEANS.2014.7003030.

[52] M. Stojanovic and J. Preisig, “Underwater Acoustic Communication Channels: Propagation Models and Statistical Characterization | Request PDF,” *ResearchGate*, Feb. 2009, doi: 10.1109/MCOM.2009.4752682.

[53] T.-C. Wu, Y.-C. Chi, H.-Y. Wang, C.-T. Tsai, and G.-R. Lin, “Blue Laser Diode Enables Underwater Communication at 12.4 Gbps,” *Sci Rep*, vol. 7, no. 1, p. 40480, Jan. 2017, doi: 10.1038/srep40480.

[54] J. Partan, J. Kurose, and B. N. Levine, “A survey of practical issues in underwater networks,” in *Proceedings of the 1st ACM international workshop on Underwater networks - WUWNet ’06*, Los Angeles, CA, USA: ACM Press, 2006, p. 17. doi: 10.1145/1161039.1161045.

[55] M. Y. I. Zia, J. Poncela, and P. Otero, “State-of-the-Art Underwater Acoustic Communication Modems: Classifications, Analyses and Design Challenges,” *ResearchGate*, Dec. 2024, doi: 10.1007/s11277-020-07431-x.

[56] H. A. Naman and A. E. Abdelkareem, “Multipath Geometry Channel Model in Shallow Water Acoustic Communication,” *J. Marine. Sci. Appl.*, vol. 22, no. 2, pp. 359–369, Jun. 2023, doi: 10.1007/s11804-023-00339-5.

[57] M. Stojanovic, “Underwater acoustic communications,” in *Proceedings of Electro/International 1995*, Jun. 1995, pp. 435–440. doi: 10.1109/ELECTR.1995.471021.

[58] W. Aman, S. Al-Kuwari, A. Kumar, M. M. U. Rahman, and M. Muzzammil, “Underwater and Air-Water Wireless Communication: State-of-the-art, Channel Characteristics, Security, and Open Problems,” Sep. 03, 2022, *arXiv*: arXiv:2203.02667. doi: 10.48550/arXiv.2203.02667.

[59] U. M. Qureshi *et al.*, “RF Path and Absorption Loss Estimation for Underwater Wireless Sensor Networks in Different Water Environments,” *Sensors (Basel)*, vol. 16, no. 6, p. 890, Jun. 2016, doi: 10.3390/s16060890.

[60] C. Fang, S. Li, Y. Wang, and K. Wang, “High-Speed Underwater Optical Wireless Communication with Advanced Signal Processing Methods Survey,” *Photonics*, vol. 10, no. 7, Art. no. 7, Jul. 2023, doi: 10.3390/photonics10070811.

[61] M. Kumar Ghosh and M. Z. Chowdhury, “Enhancing underwater acoustic communication networks with RIS: Precise performance analysis over $\kappa-\mu$ shadowed fading distribution,” *Results in Engineering*, vol. 26, p. 105446, Jun. 2025, doi: 10.1016/j.rineng.2025.105446.

[62] I. Vasilescu, K. Kotay, D. Rus, M. Dunbabin, and P. I. Corke, “Data Collection, Storage, and Retrieval with an Underwater Sensor Network,” in *ResearchGate*, doi: 10.1145/1098918.1098936.

[63] D. B. Kilfoyle and A. B. Baggeroer, “The state of the art in underwater acoustic telemetry,” *IEEE Journal of Oceanic Engineering*, vol. 25, no. 1, pp. 4–27, Jan. 2000, doi: 10.1109/48.820733.

[64] J. F. Lynch, G. G. Gawarkiewicz, Y.-T. Lin, T. F. Duda, and A. E. Newhall, “Impacts of Ocean Warming on Acoustic Propagation Over Continental Shelf and Slope Regions,” *ResearchGate*, Jun. 2018, doi: 10.5670/oceanog.2018.219.

[65] M. G. Wells *et al.*, “Speed of sound gradients due to summer thermal stratification can reduce the detection range of acoustic fish tags: results from a field study in

Hamilton Harbour, Ontario," *Can. J. Fish. Aquat. Sci.*, vol. 78, no. 3, pp. 269–285, Mar. 2021, doi: 10.1139/cjfas-2020-0078.

[66] J. P. Beam, "Sound Absorption in the Ocean as a Function of Temperature, Salinity, and Frequency," *The Journal of the Acoustical Society of America*, vol. 34, no. 12_Supplement, p. 1975, Dec. 1962, doi: 10.1121/1.1936974.

[67] P. Brewer, E. T. Peltzer, J. Ryan, W. J. Kirkwood, and A. F. Hofmann, "Ocean chemistry and the speed of sound in seawater | Request PDF," *ResearchGate*, Oct. 2015, doi: 10.1016/j.marchem.2015.09.009.

[68] A. Makar, "Simplified Method of Determination of the Sound Speed in Water on the Basis of Temperature Measurements and Salinity Prediction for Shallow Water Bathymetry," *Remote Sensing*, vol. 14, no. 3, Art. no. 3, Jan. 2022, doi: 10.3390/rs14030636.

[69] G. Galeron, M. Amielh, and P.-O. Mattei, "Characterization of acoustic sources in a corrugated pipe flow with linear stochastic estimation," *The Journal of the Acoustical Society of America*, vol. 150, no. 6, pp. 4268–4282, Dec. 2021, doi: 10.1121/10.0008897.

[70] A. MADIROLAS, M. Acha, R. A. Guerrero, and C. A. Lasta, "Sources of acoustic scattering near a halocline in an estuarine frontal system," *ResearchGate*, Dec. 2024, Accessed: Jun. 04, 2025. [Online]. Available: https://www.researchgate.net/publication/238698625_Sources_of_acoustic_scattering_near_a_halocline_in_an_estuarine_frontal_system

[71] M. Lylloff, "All about speed of sound in water - what it is, how it's measured, and why we care," RBR Global. Accessed: Jun. 04, 2025. [Online]. Available: https://rbr-global.com/speed-of-sound-in-water/

[72] S. Wu, Z. Li, J. Qin, M. Wang, and W. Li, "The Effects of Sound Speed Profile to the Convergence Zone in Deep Water," *Journal of Marine Science and Engineering*, vol. 10, no. 3, Art. no. 3, Mar. 2022, doi: 10.3390/jmse10030424.

[73] S. Shaikh *et al.*, "Acoustic Propagation and Transmission Loss Analysis in Shallow Water of Northern Arabian Sea," *Journal of Marine Science and Engineering*, vol. 12, no. 12, Art. no. 12, Dec. 2024, doi: 10.3390/jmse12122256.

[74] W. Mallik, R. K. Jaiman, and J. Jelovica, "Predicting transmission loss in underwater acoustics using convolutional recurrent autoencoder network," *The Journal*

of the Acoustical Society of America, vol. 152, no. 3, pp. 1627–1638, Sep. 2022, doi: 10.1121/10.0013894.

[75] A. J. Duncan, A. N. Gavrilov, R. D. McCauley, I. Parnum, and Jon M Collis, "Characteristics of sound propagation in shallow water over an elastic seabed with a thin cap-rock layer," *ResearchGate*, Jul. 2013, doi: 10.1121/1.4809723.

[76] K. B. Smith, A. A. M. Abrantes, and A. Larraza, "Examination of time-reversal acoustics in shallow water and applications to noncoherent underwater communications," *J Acoust Soc Am*, vol. 113, no. 6, pp. 3095–3110, Jun. 2003, doi: 10.1121/1.1570831.

[77] F. Busacca, L. Galluccio, S. Palazzo, and A. Panebianco, "A comparative analysis of predictive channel models for real shallow water environments," *Computer Networks*, vol. 250, p. 110557, Aug. 2024, doi: 10.1016/j.comnet.2024.110557.

[78] S. P. Czenszak and J. L. Krolik, "Robust wideband matched-field processing with a short vertical array," *The Journal of the Acoustical Society of America*, vol. 101, no. 2, pp. 749–759, Feb. 1997, doi: 10.1121/1.417958.

[79] R. Gao, M. Liang, H. Dong, X. Luo, and P. N. Suganthan, "Underwater Acoustic Signal Denoising Algorithms: A Survey of the State-of-the-art," Jul. 18, 2024, *arXiv*: arXiv:2407.13264. doi: 10.48550/arXiv.2407.13264.

[80] J. F. Lynch, G. G. Gawarkiewicz, Y.-T. Lin, T. F. Duda, and A. E. Newhall, "Impacts of Ocean Warming on Acoustic Propagation Over Continental Shelf and Slope Regions," *Oceanography*, vol. 31, no. 2, pp. 174–181, Aug. 2018, doi: 10.5670/oceanog.2018.219.

[81] D. Defrianto, N. Pratama, and U. Malik, "Determination of the shadow zone area in the ocean computationally by simulating the propagation of acoustic rays," *ResearchGate*, Feb. 2023, doi: 10.59190/stc.v3i2.228.

[82] A. Affatati, C. Scaini, and S. Salon, "Ocean Sound Propagation in a Changing Climate: Global Sound Speed Changes and Identification of Acoustic Hotspots," *Earth's Future*, vol. 10, no. 3, p. e2021EF002099, 2022, doi: 10.1029/2021EF002099.

[83] L. Possenti *et al.*, "The present and future contribution of ships to the underwater soundscape," *Front. Mar. Sci.*, vol. 11, Mar. 2024, doi: 10.3389/fmars.2024.1252901.

[84] C. T. Tindle, G. B. Deane, and J. Preisig, “Reflection of underwater sound from surface waves,” *ResearchGate*, Feb. 2009, doi: 10.1121/1.3035828.

[85] B. Roche, T. G. Leighton, P. R. White, and J. M. Bull, “Ambient Bubble Acoustics: Seep, Rain, and Wave Noise,” in *Geophysical Monograph Series*, 1st ed., G. Bayrakci and F. Klingelhoefer, Eds., Wiley, 2024, pp. 161–182. doi: 10.1002/9781119750925.ch10.

[86] Y. Y. Al-Aboosi and A. Z. Sha’ameri, “Experimental Multipath Delay Profile of Underwater Acoustic Communication Channel in Shallow Water,” *ResearchGate*, May 2016, doi: 10.11591/ijeecs.v2.i2.pp351-358.

[87] N. U. R. Junejo *et al.*, “A Survey on Physical Layer Techniques and Challenges in Underwater Communication Systems,” *Journal of Marine Science and Engineering*, vol. 11, no. 4, Art. no. 4, Apr. 2023, doi: 10.3390/jmse11040885.

[88] C. Feng, Y. Luo, J. Zhang, and H. Li, “An OFDM-Based Frequency Domain Equalization Algorithm for Underwater Acoustic Communication with a High Channel Utilization Rate,” *Journal of Marine Science and Engineering*, vol. 11, no. 2, Art. no. 2, Feb. 2023, doi: 10.3390/jmse11020415.

[89] A. C. Sing, J. K. Nelson, and S. S. Kozat, “Signal processing for underwater acoustic communications,” *IEEE Commun. Mag.*, vol. 47, no. 1, pp. 90–96, Jan. 2009, doi: 10.1109/MCOM.2009.4752683.

[90] G. Burrowes and J. Y. Khan, “Short-Range Underwater Acoustic Communication Networks,” in *ResearchGate*, 2011. doi: 10.5772/24098.

[91] R. G and S. G, “Analysis of Doppler Effects in Underwater Acoustic Channels using Parabolic Expansion Modeling,” *International Journal of Advanced Computer Science and Applications (ijacsa)*, vol. 10, no. 3, Art. no. 3, 30 2019, doi: 10.14569/IJACSA.2019.0100327.

[92] R. Diamant, A. Feuer, and L. Lampe, “Choosing the right signal: Doppler shift estimation for underwater acoustic signals,” in *Proceedings of the Seventh ACM International Conference on Underwater Networks and Systems - WUWNet ’12*, Los Angeles, California: ACM Press, 2012, p. 1. doi: 10.1145/2398936.2398971.

[93] G. Yang *et al.*, “Optimized Doppler Estimation and Symbol Synchronization for Mobile M-ary Spread Spectrum Underwater Acoustic Communication,” *Journal*

of Marine Science and Engineering, vol. 9, no. 9, Art. no. 9, Sep. 2021, doi: 10.3390/jmse9091001.

[94] C. Liu, Y. V. Zakharov, and T. Chen, "Doubly Selective Underwater Acoustic Channel Model for a Moving Transmitter/Receiver | Request PDF," *ResearchGate*, Mar. 2012, doi: 10.1109/TVT.2012.2187226.

[95] T. C. Yang, "Toward continuous underwater acoustic communications," in *OCEANS 2008*, Sep. 2008, pp. 1–6. doi: 10.1109/OCEANS.2008.5151827.

[96] K. Pelekanakis and M. Chitre, "Robust Equalization of Mobile Underwater Acoustic Channels," *IEEE J. Oceanic Eng.*, vol. 40, no. 4, pp. 775–784, Oct. 2015, doi: 10.1109/JOE.2015.2469895.

[97] J. Thomson *et al.*, "Surface Wave Development and Ambient Sound in the Ocean," *Journal of Geophysical Research: Oceans*, vol. 129, no. 12, p. e2024JC021921, 2024, doi: 10.1029/2024JC021921.

[98] C. Knowlton, "How is sound used to measure rainfall over the ocean?," Discovery of Sound in the Sea. Accessed: Jun. 05, 2025. [Online]. Available: https://dosits.org/people-and-sound/study-weather/how-is-sound-used-to-measure-rainfall-over-the-ocean/

[99] National Ocean Service, "What is the bloop?" Accessed: Jun. 05, 2025. [Online]. Available: https://oceanservice.noaa.gov/facts/bloop.html?utm_source=chatgpt.com

[100] N. R. C. (US) C. on P. I. of A. N. in the O. on M. Mammals, "Sources of Sound in the Ocean and Long-Term Trends in Ocean Noise," in *Ocean Noise and Marine Mammals*, National Academies Press (US), 2003. Accessed: Jun. 05, 2025. [Online]. Available: https://www.ncbi.nlm.nih.gov/books/NBK221253/

[101] M. Simon, "Oceans Aren't Just Warming—Their Soundscapes Are Transforming," *Wired*, Apr. 14, 2022. Accessed: Jun. 05, 2025. [Online]. Available: https://www.wired.com/story/oceans-arent-just-warming-their-soundscapes-are-transforming/

[102] P. H. Dahl, J. H. Miller, D. H. Cato, and R. K. Andrew, "UNDERWATER AMBIENT NOISE," Jan. 2007.

[103] S. M. Santhanam and V. Natarajan, “Performance analysis of signal to noise ratio and bit error rate for multiuser using Passive Time Reversal Technique in underwater communication,” *ResearchGate*, Jan. 2010, doi: 10.1109/ICWCSC.2010.5415895.

[104] N. Fisheries, “Ocean Noise | NOAA Fisheries,” NOAA. Accessed: Jun. 05, 2025. [Online]. Available: https://www.fisheries.noaa.gov/national/science-data/ocean-noise

[105] C. Knowlton, “How does marine life affect ocean sound levels?,” Discovery of Sound in the Sea. Accessed: Jun. 05, 2025. [Online]. Available: https://dosits.org/science/sounds-in-the-sea/how-does-marine-life-affect-ocean-sound-levels/

[106] Z. Song *et al.*, “Sounds of snapping shrimp (Alpheidae) as important input to the soundscape in the southeast China coastal sea,” *Front. Mar. Sci.*, vol. 10, Jan. 2023, doi: 10.3389/fmars.2023.1029003.

[107] T. Morisaka, “Evolution of Communication Sounds in Odontocetes: A Review,” *ResearchGate*, Jan. 2012, doi: 10.46867/IJCP.2012.25.01.04.

[108] J. A. Hildebrand, “Anthropogenic and natural sources of ambient noise in the ocean,” *Marine Ecology Progress Series*, vol. 395, pp. 5–20, Dec. 2009, doi: 10.3354/meps08353.

[109] G. J. Lowes, J. Neasham, R. Burnett, B. Sherlock, and C. Tsimenidis, “Passive Acoustic Detection of Vessel Activity by Low-Energy Wireless Sensors,” *Journal of Marine Science and Engineering*, vol. 10, no. 2, Art. no. 2, Feb. 2022, doi: 10.3390/jmse10020248.

[110] N. D. Merchant *et al.*, “Impulsive noise pollution in the Northeast Atlantic: Reported activity during 2015–2017,” *Marine Pollution Bulletin*, vol. 152, p. 110951, Mar. 2020, doi: 10.1016/j.marpolbul.2020.110951.

[111] H. Klinck *et al.*, “Near-Real-Time Acoustic Monitoring of Beaked Whales and Other Cetaceans Using a Seaglider™,” *PLOS ONE*, vol. 7, no. 5, p. e36128, May 2012, doi: 10.1371/journal.pone.0036128.

[112] C. Erbe and D. M. Farmer, “Zones of impact around icebreakers affecting beluga whales in the Beaufort Sea,” *The Journal of the Acoustical Society of America*, vol. 108, no. 3, pp. 1332–1340, Sep. 2000, doi: 10.1121/1.1288938.

[113] W. H. Thorp, “Analytic Description of the Low-Frequency Attenuation Coefficient,” *The Journal of the Acoustical Society of America*, vol. 42, no. 1, p. 270, Jul. 1967, doi: 10.1121/1.1910566.

[114] Y. Y. Al-Alaboosi and J. A. Al-Aboosi, “Study of Absorption Loss Effects on Acoustic Wave Propagation in Shallow Water Using Different Empirical Models,” *IJAAS*, vol. 7, no. 1, p. 1, Mar. 2018, doi: 10.11591/ijaas.v7.i1.pp1-6.

[115] V. V. V., “Communication on Flat Fading Channels,” 2007, Accessed: Jun. 05, 2025. [Online]. Available: https://vvv.ece.illinois.edu/ece559/handouts/notes4.pdf

[116] A. Sánchez, E. Robles, F. J. Rodrigo, F. Ruiz-Vega, U. Fernández-Plazaola, and J. F. Paris, “MEASUREMENT AND MODELING OF FADING IN ULTRASONIC UNDERWATER CHANNELS,” 2015.

[117] R. P. Stokey, L. E. Freitag, and M. D. Grund, “A Compact Control Language for AUV acoustic communication,” in *Europe Oceans 2005*, Brest, France: IEEE, 2005, pp. 1133-1137 Vol. 2. doi: 10.1109/OCEANSE.2005.1513217.

[118] J.-H. Li, B.-H. Jun, P.-M. Lee, and S.-W. Hong, “A hierarchical real-time control architecture for a semi-autonomous underwater vehicle,” *ResearchGate*, Feb. 2005, doi: 10.1016/j.oceaneng.2004.12.003.

[119] A. Farhadi, J. Dumon, and C. Canudas-de-Wit, “A supervisory control policy over an acoustic communication network,” *International Journal of Control*, pp. 1–13, Dec. 2014, doi: 10.1080/00207179.2014.986201.

[120] D. A. Robb *et al.*, “A Natural Language Interface with Relayed Acoustic Communications for Improved Command and Control of AUVs,” Nov. 08, 2018. Accessed: Jun. 08, 2025. [Online]. Available: https://arxiv.org/abs/1811.03566v2

[121] T. Luan, X. Bai, X. Zhang, M. Wang, and M. Sun, “UUV two-phase formation and priority avoidance control considering steering amplitude limitation,” *Ocean Engineering*, vol. 312, p. 119130, Nov. 2024, doi: 10.1016/j.oceaneng.2024.119130.

[122] K. Muljowidodo and N. S. Adi, “Heading Lock Maneuver Testing of Autonomous Underwater Vehicle,” Apr. 24, 2008. Accessed: Jun. 08, 2025. [Online]. Available: https://arxiv.org/abs/0804.3885v1

[123] C. I. Sprague, Ö. Özkahraman, A. Munafo, R. Marlow, A. Phillips, and P. Ögren, “Improving the Modularity of AUV Control Systems using Behaviour Trees,” Nov. 01, 2018, *arXiv*: arXiv:1811.00426. doi: 10.48550/arXiv.1811.00426.

[124] S. M. Zadeh, D. M. W. Powers, and K. Sammut, “An Autonomous Reactive Architecture for Efficient AUV Mission Time Management in Realistic Severe Ocean Environment,” Jun. 15, 2016, *arXiv*: arXiv:1604.08336. doi: 10.48550/arXiv.1604.08336.

[125] Z. Feng and R. Allen, “Evaluation of the effects of the communication cable on the dynamics of an underwater flight vehicle | Request PDF,” *ResearchGate*, Jun. 2004, doi: 10.1016/j.oceaneng.2003.11.001.

[126] B. Phillips, N. Chaloux, R. Shomberg, A. M. Munoz, and J. Owens, “The Fiber Optic Reel System: A Compact Deployment Solution for Tethered Live-Telemetry Deep-Sea Robots and Sensors,” *ResearchGate*, Apr. 2025, doi: 10.3390/s21072526.

[127] lindenphotonics, “Tethered vs. Untethered ROVs: Pros and Cons for Different Underwater Missions,” Linden Photonics Inc. Accessed: Jun. 09, 2025. [Online]. Available: https://www.lindenphotonics.com/tethered-vs-untethered-rovs-pros-and-cons-for-different-underwater-missions

[128] A. Patil, M. Park, and J. Bae, “Coordinating Tethered Autonomous Underwater Vehicles towards Entanglement-Free Navigation,” *Robotics*, vol. 12, no. 3, Art. no. 3, Jun. 2023, doi: 10.3390/robotics12030085.

[129] Ecoptik, “About underwater ROV.,” Ecoptik. Accessed: Jun. 12, 2025. [Online]. Available: https://www.ecoptik.net

[130] O. Tortorici, C. Péraud, C. Anthierens, and V. Hugel, “Automated Deployment of an Underwater Tether Equipped with a Compliant Buoy–Ballast System for Remotely Operated Vehicle Intervention,” *Journal of Marine Science and Engineering*, vol. 12, no. 2, Art. no. 2, Feb. 2024, doi: 10.3390/jmse12020279.

[131] I. Martínez de Alegría, I. Rozas Holgado, E. Ibarra, E. Robles, and J. L. Martín, “Wireless Power Transfer for Unmanned Underwater Vehicles: Technologies, Challenges and Applications,” *Energies*, vol. 17, no. 10, Art. no. 10, Jan. 2024, doi: 10.3390/en17102305.

[132] F. A. Azis, M. S. M. Aras, M. Z. A. Rashid, M. N. Othman, and S. S. Abdullah, “Problem Identification for Underwater Remotely Operated Vehicle (ROV): A Case Study,” *Procedia Engineering*, vol. 41, pp. 554–560, Jan. 2012, doi: 10.1016/j.proeng.2012.07.211.

[133] S. Savitz, “Uncrewed maritime vessels: Shaping naval power through hybrid threats,” Hybrid CoE - The European Centre of Excellence for Countering Hybrid Threats. Accessed: Jun. 09, 2025. [Online]. Available: https://www.hybrid-coe.fi/publications/hybrid-coe-working-paper-34-uncrewed-maritime-vessels-shaping-naval-power-in-hybrid-threat-operations/

[134] Y. Li, Y. Zhang, H. Zhou, and T. Jiang, “To Relay or not to Relay: Open Distance and Optimal Deployment for Linear Underwater Acoustic Networks,” *arXiv.org*, Jan. 2018, doi: 10.1109/TCOMM.2018.2822287.

[135] W. Zhang, M. Stojanovic, and U. Mitra, “Analysis of a simple multihop underwater acoustic network,” in *ResearchGate*, Jan. 2008. doi: 10.1145/1410107.1410110.

[136] R. Su, Venkatesan ,Ramachandran, and C. and Li, “An energy-efficient relay node selection scheme for underwater acoustic sensor networks,” *Cyber-Physical Systems*, vol. 1, no. 2–4, pp. 160–179, Oct. 2015, doi: 10.1080/23335777.2016.1145742.

[137] R. Otnes *et al.*, *Underwater Acoustic Networking Techniques*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012. doi: 10.1007/978-3-642-25224-2.

[138] R. Santos *et al.*, “Real-Time Communication Support for Underwater Acoustic Sensor Networks,” *Sensors*, vol. 17, no. 7, Art. no. 7, Jul. 2017, doi: 10.3390/s17071629.

[139] J. Heidemann, Y. Li, A. Syed, J. Wills, and W. Ye, “Underwater Sensor Networking: Research Challenges and Potential Applications,” Jan. 2006.

[140] D. Zhao, G. Lun, R. Xue, and Y. Sun, “Cross-Layer-Aided Opportunistic Routing for Sparse Underwater Wireless Sensor Networks,” *Sensors (Basel)*, vol. 21, no. 9, p. 3205, May 2021, doi: 10.3390/s21093205.

[141] S. A. H. Mohsan, Y. Li, M. Sadiq, J. Liang, and M. A. Khan, “Recent Advances, Future Trends, Applications and Challenges of Internet of Underwater Things (IoUT): A Comprehensive Review,” *Journal of Marine Science and Engineering*, vol. 11, no. 1, Art. no. 1, Jan. 2023, doi: 10.3390/jmse11010124.

[142] L. Nkenyereye, L. Nkenyereye, and B. Ndibanje, “Internet of Underwater Things: A Survey on Simulation Tools and 5G-Based Underwater Networks,” *Electronics*, vol. 13, no. 3, Art. no. 3, Jan. 2024, doi: 10.3390/electronics13030474.

[143] A. P. Das and S. M. Thampi, “Simulation Tools for Underwater Sensor Networks: A Survey,” *ResearchGate*, Jan. 2017, doi: 10.5296/npa.v8i4.10471.

[144] A. Kapoor, “Unable to install on ns3-3.42 and ubuntu 24.04VM · Issue #34 · rmartin5/aqua-sim-ng,” GitHub. Accessed: Jun. 10, 2025. [Online]. Available: https://github.com/rmartin5/aqua-sim-ng/issues/34

[145] Ns-3, “6. Tweaking — Tutorial.” Accessed: Jun. 10, 2025. [Online]. Available: https://www.nsnam.org/docs/tutorial/html/tweaking.html?utm_source=chatgpt.com

[146] Ns-3, “8. Tracing — Tutorial.” Accessed: Jun. 10, 2025. [Online]. Available: https://www.nsnam.org/docs/tutorial/html/tracing.html?utm_source=chatgpt.com

[147] R. Martin, *rmartin5/aqua-sim-ng*. (Jun. 03, 2025). C++. Accessed: Jun. 10, 2025. [Online]. Available: https://github.com/rmartin5/aqua-sim-ng

[148] D. L. Raja, “STUDY OF VARIOUS NETWORK SIMULATORS,” vol. 05, no. 12, 2018.

[149] S. Aldhaheri, Y. Hu, Y. Xie, P. Wu, D. Kanoulas, and Y. Liu, “Underwater Robotic Simulators Review for Autonomous System Development,” Apr. 08, 2025, *arXiv*: arXiv:2504.06245. doi: 10.48550/arXiv.2504.06245.

[150] T. S. P. Kumar, “How to Build Aqua-Sim NG in NS-3.40 on Ubuntu 24.04 – Complete Guide for Underwater Sensor Network Simulations.” Accessed: Jun. 10, 2025. [Online]. Available: https://www.nsnam.com/2025/05/how-to-build-aqua-sim-ng-in-ns-340-on.html

[151] J. P. McAuliffe and P. Lanaspa, “Python for Rapid Science Operations Analysis, Prototyping and Planning for BepiColombo,” in *ResearchGate*, May 2016. doi: 10.2514/6.2016-2438.

[152] Y. Zhang *et al.*, “Reinforcement Learning Based Relay Selection for Underwater Acoustic Cooperative Networks,” *Remote Sensing*, vol. 14, no. 6, Art. no. 6, Jan. 2022, doi: 10.3390/rs14061417.

[153] A. Knoblauch, “IVISIT: An Interactive Visual Simulation Tool for system simulation, visualization, optimization, and parameter management,” Aug. 10, 2024, *arXiv*: arXiv:2408.03341. doi: 10.48550/arXiv.2408.03341.

[154] M. Köstler and F. Kauer, “A Remote Interface for Live Interaction with OMNeT++ Simulations,” Sep. 08, 2017, *arXiv*: arXiv:1709.02822. doi: 10.48550/arXiv.1709.02822.

[155] Ns-3, “3. System Prerequisites — Installation guide.” Accessed: Jun. 12, 2025. [Online]. Available: https://www.nsnam.org/docs/installation/html/system.html?utm_source=chatgpt.com

[156] D. Centelles, A. Soriano-Asensi, J. V. Martí, R. Marín, and P. J. Sanz, “Underwater Wireless Communications for Cooperative Robotics with UWSim-NET,” *Applied Sciences*, vol. 9, no. 17, Art. no. 17, Jan. 2019, doi: 10.3390/app9173526.

[157] Y. Feng *et al.*, “Echo: Simulating Distributed Training At Scale,” Dec. 17, 2024, *arXiv*: arXiv:2412.12487. doi: 10.48550/arXiv.2412.12487.

[158] A. Vaswani *et al.*, “Attention Is All You Need,” Aug. 02, 2023, *arXiv*: arXiv:1706.03762. doi: 10.48550/arXiv.1706.03762.

[159] S. Kiranyaz, O. Avci, O. Abdeljaber, T. Ince, M. Gabbouj, and D. J. Inman, “1D Convolutional Neural Networks and Applications: A Survey,” May 09, 2019, *arXiv*: arXiv:1905.03554. doi: 10.48550/arXiv.1905.03554.

[160] S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory,” *ResearchGate*, Feb. 2025, doi: 10.1162/neco.1997.9.8.1735.

[161] D. P. Kingma and J. Ba, “Adam: A Method for Stochastic Optimization,” Jan. 30, 2017, *arXiv*: arXiv:1412.6980. doi: 10.48550/arXiv.1412.6980.

[162] Google, “Classification: Accuracy, recall, precision, and related metrics | Machine Learning,” Google for Developers. Accessed: Jun. 12, 2025. [Online]. Available: https://developers.google.com/machine-learning/crash-course/classification/accuracy-precision-recall

[163] P. Christen, D. J. Hand, and N. Kirielle, “A Review of the F-Measure: Its History, Properties, Criticism, and Alternatives,” *ACM Comput. Surv.*, vol. 56, no. 3, p. 73:1-73:24, Oct. 2023, doi: 10.1145/3606367.

[164] M. C. Hinojosa Lee, J. Braet, and J. Springael, “Performance Metrics for Multilabel Emotion Classification: Comparing Micro, Macro, and Weighted F1-Scores,” *Applied Sciences*, vol. 14, no. 21, Art. no. 21, Jan. 2024, doi: 10.3390/app14219863.

[165] T. Fawcett, “An introduction to ROC analysis,” *Pattern Recogn. Lett.*, vol. 27, no. 8, pp. 861–874, Jun. 2006, doi: 10.1016/j.patrec.2005.10.010.

[166] M. Sokolova and G. Lapalme, “A systematic analysis of performance measures for classification tasks,” *Information Processing & Management*, vol. 45, no. 4, pp. 427–437, Jul. 2009, doi: 10.1016/j.ipm.2009.03.002.

[167] M. Grandini, E. Bagli, and G. Visani, “Metrics for Multi-Class Classification: an Overview,” Aug. 13, 2020, *arXiv*: arXiv:2008.05756. doi: 10.48550/arXiv.2008.05756.