Tafila Technical University
College of Engineering

Department of Computer Engineering and Communication

2025/2026 Winter Semester

Graduation Project 2

Monitoring Submarine Robot (MSR)

Supervised by:
Dr. Murad Al-Aqtash

Committee Member Names:

Dr. Naeem Al-Odat

Dr.Ahmad Al-jaafreh

Dr. Murad Al-Aqtash

Student Name	**Mohammad Al-Saaideh**
Student ID Number	**320210112015**
Student Name	**Rashad Jarbou**
Student ID Number	**320210112007**
Student Name	**Abdellrahman Al-Hanaqtah**
Student ID Number	**320210112011**

ACKNOWLEDGEMENT

First, all praise to Allah for granting us the strength to complete this work.
We would like to express our sincere gratitude to our supervisor, Dr. Murad Al-Aqtash, for his invaluable guidance and support throughout the development of the Monitoring Submarine Robot (MSR).
We also extend our appreciation to the Department of Computer Engineering and Communication at Tafila Technical University for providing the necessary resources.
Finally, heartfelt thanks to our parents and families for their endless encouragement and sacrifice during our studies.

Pledge

This is to declare that the project entitled “Monitoring Submarine Robot (MSR)” is an original work done by the undersigned, in partial fulfillment of the requirements for the bachelor's degree at the Department of Computer Engineering and Communication, College of Engineering, Tafila Technical University.

All the analysis, design, and system development have been accomplished by the undersigned. Moreover, this project has not been submitted to any other college or university.

Student Name: Mohammad Al-Saaideh

Student Name: Rashad Jarbou

Student Name: Abdellrahman Al-Hanaqtah

Monitoring Submarine Robot (MSR)

CHAPTER 1: Introduction

1.1 Overview

The Gulf of Aqaba hosts one of the most unique and resilient coral reef ecosystems in the world, supporting biodiversity, tourism, and the livelihoods of local communities. Despite their resilience, these reefs are not immune to environmental stressors, and their preservation requires proactive monitoring and management. Advances in robotics, sensing technologies, and artificial intelligence now make it possible to design systems capable of collecting ecological and environmental data with high accuracy. This project leverages these technologies to develop a Smart Underwater Vehicle (ROV) specifically designed to monitor marine ecosystems in the Gulf of Aqaba.

1.2 Project Motivation

Coral reef conservation in Aqaba is essential not only for protecting marine biodiversity but also for sustaining tourism and the local economy. However, conventional monitoring methods, such as diver-based surveys, are costly, time-intensive, and provide only limited snapshots of reef conditions. Emerging technologies in remotely operated robotics and computer vision offer a promising alternative by enabling real-time, precise, and safe monitoring of coral ecosystems. By integrating a tethered ROV system with advanced object detection, this project seeks to provide conservation authorities with a reliable tool for data-driven decision-making.

1.3 Problem Statement

The coral reefs of Aqaba are among the most resilient worldwide, yet their survival is increasingly threatened by uncontrolled tourism, rising pollution levels, and the severe impacts of climate change. Current monitoring methods are limited in scale and continuity. Conservation authorities require a robust, remotely controlled monitoring system capable of navigating currents and providing real-time insights into reef health without the risks associated with human divers.

1.4 Project Aim and Objectives

Aim: To design and implement a smart Remotely Operated Vehicle (ROV) for real-time coral reef monitoring in Aqaba.

Objectives:

1. Monitors coral reef health status (healthy, stressed, bleached, or dead) over time.
2. Provides continuous data logging and visualization to aid researchers and local environmental authorities in decision-making.
3. Navigates the marine environment using a stable, remotely controlled propulsion system to facilitate targeted underwater inspections.
4. Executes real-time object detection to identify and categorize ecosystem conditions using a custom-trained computer vision model.

1.5 Project Scope

This project involves designing and building a smart ROV to monitor the coral reefs of Aqaba. The vehicle integrates a CSI v1.3 camera interfaced with a Jetson Nano (4GB) and an Arduino-based motor control system.

The video feed is transmitted via a Cat 6 Ethernet cable to a surface laptop, where a YOLO v11 model processes the data. The project focuses on the development of a fully integrated prototype, comprehensive wiring, and software implementation for detection and control. Deep-sea exploration beyond the cable length and fully autonomous navigation are beyond the scope of this work.

1.6 Project Software and Hardware Requirements

Hardware:

- **Subsea Unit:** Jetson Nano (4GB RAM), CSI v1.3 Camera, Arduino (programmed in C), 3x Brushless Motors with ESCs (2 side, 1 vertical), 4S 18Ah Battery (100A continuous/burst), 2x 36V-to-5V Buck Converters.
- **Surface Unit:** Laptop, USB Joystick, Cat 6 Ethernet Cable.

Software:

- **Embedded:** Arduino IDE (C), Linux (Jetson Nano).
- **Surface/Processing:** Python (VS Code), YOLO v11 (Object Detection), Streamlit (Video/Result Display), Grafana (Dashboard), InfluxDB (Database).

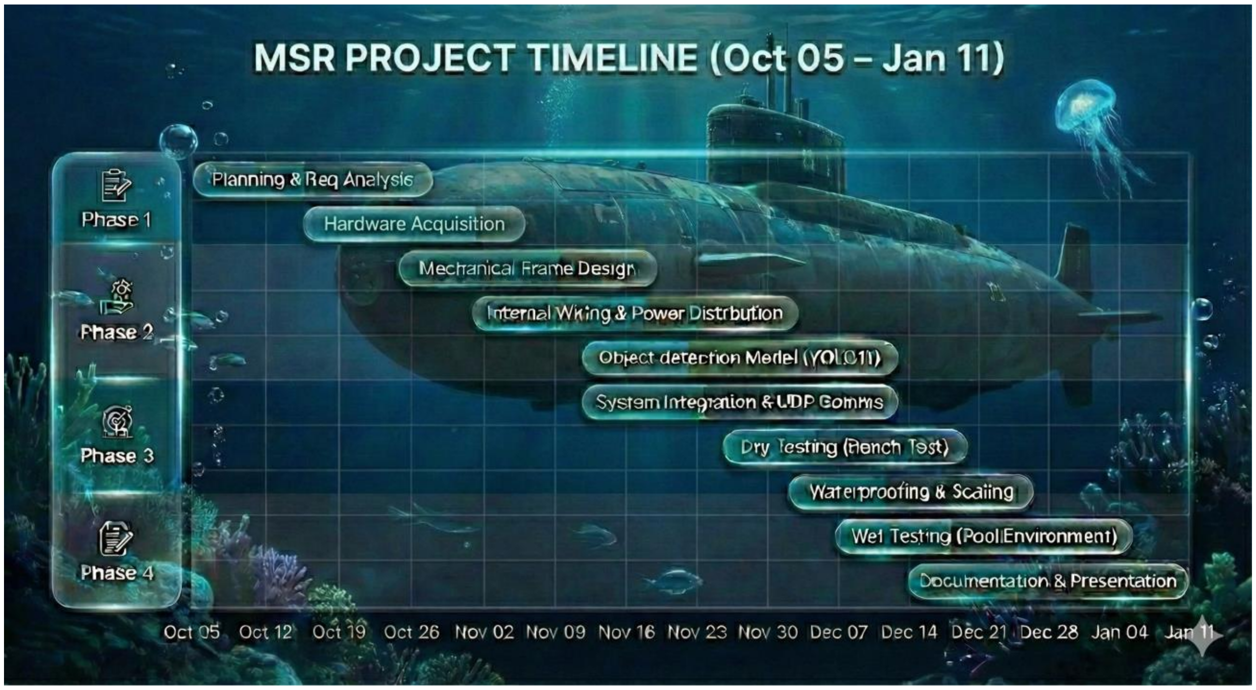
1.7 Project Limitations

- **Depth and endurance limitations** due to battery constraints and the physical length of the Ethernet tether.
- **The model's performance depends on the quality and comprehensiveness of the trained dataset.**
- **Environmental conditions** (such as strong currents and water visibility) may affect navigation stability and camera performance.

1.8 Project Expected Output

- Functional submarine prototype.
- AI model capable of classifying coral health conditions.
- Dataset of Aqaba reef conditions.
- Visualization dashboard for environmental monitoring.

1.9 Project schedule



CHAPTER 2: Related Existing Systems

2.1 Introduction

The field of underwater robotics has evolved significantly, offering various solutions for marine exploration and environmental monitoring. These systems typically range from high-end industrial vehicles to low-cost recreational units; each designed for specific operational depths and tasks. This chapter provides a comparative review of existing underwater systems, analyzing their capabilities, costs, and limitations. By examining these precedents, the chapter establishes the current technological landscape and highlights the specific gaps that the proposed project aims to address.

2.2 Comparison of Underwater Drones

Type	Power Consumption	Speed (kt)	Depth	Cost $	KeyPros	Key Cons	Image
Aquanaut (Nauticus Robotics)	High (30–300 kWh battery)	3–5	3000 m	Very High	Hybrid autonomy, no large vessel needed	Complex, very high cost	
Bluefin HAUV	Moderate (battery, 3–6 h missions)	2–3	60 m	Very High	Portable, precise hovering	Shallow depth, niche use	
Boaty McBoatface	Low to Moderate (ultra-efficient lithium)	1.0	6000 m	Very High	World-class endurance, deep diving	Slow, no manipulation	
IIPOSAQQ Drone (8 Thrusters)	Moderate (1–2 h battery)	1–2	100 m	$1.5k–$5k	Easy deploy, stable video	Shallow, low payload	
CP5 DIY Drone	Low (small battery, short demos)	0.5–1	Shallow (pool)	$200–$1000	Good for beginners, flexible	Not commercial, limited perf.	
BlueROV2	Moderate (battery-powered, tethered control)	2–3	100–300 m	$4k–$10k	Affordable, open-source, customizable	Limited depth	
DIY PVC ROV	Low (small battery pack)	0.5–1	5–30 m	$150–$800	Cheap, customizable, learning	Low depth/reliability	

2.3 Overall Problems of Existing Systems

Despite the wide variety of available underwater robotic systems, several common limitations exist when considering coral reef monitoring applications:

1. **High Cost:**
   Many advanced systems are prohibitively expensive, making them unsuitable for continuous environmental monitoring by local conservation authorities.

2. **Overqualification:**
   Deep-sea vehicles provide capabilities far beyond what is required for shallow reef environments, resulting in unnecessary complexity and cost.

3. **Data Latency:**
   Wireless AUVs cannot provide the high-bandwidth, real-time video feedback required for immediate AI analysis on the surface.

2.4 Overall Solution Approach

MSR utilizes a tethered ROV architecture that balances professional performance with affordability. By transmitting video to a surface laptop via Ethernet, the system overcomes onboard processing limits, enabling advanced real-time AI analysis without the high cost of industrial vehicles.

The proposed approach emphasizes:

- **Cost-Effectiveness:** Uses off-the-shelf components (Arduino, Jetson Nano) to create an accessible design.

- **Real-Time Intelligence:** Leverages surface computing power for immediate YOLO-based coral detection.

- **Extended Endurance:** Employs a high-capacity 4S battery to effectively navigate local currents.

- **Reliable Control:** Ensures operational safety and precision through tethered, manual command.

- **Minimal Impact:** Prioritizes non-intrusive visual monitoring to protect the ecosystem.

CHAPTER 3: Requirements Engineering and Analysis

3.1 Introduction

Requirements engineering defines the functional and non-functional needs of a system to ensure it meets its intended purpose. For the underwater systems, clearly defined requirements are essential to guarantee reliable operation in complex marine environments. This chapter outlines the key requirements for the MSR.

3.2 Stakeholders

The primary stakeholders include marine researchers, environmental conservation authorities, system operators, and local coastal management bodies. These stakeholders rely on accurate environmental data, reliable system operation, and minimal ecological disturbance.

3.2.1 Functional Requirements

- **Remote Navigation:** The system must be controllable via a USB joystick connected to a laptop.
- **Video Transmission:** Real-time video must be sent from the CSI v1.3 camera via Cat 6 Ethernet to the surface.
- **Object Detection:** The laptop must run a YOLO v11 model to localize and classify coral as healthy, bleached, stressed, or dead.
- **Data Visualization:** Detection results must be displayed on a Streamlit app, and data logged to InfluxDB for Grafana visualization.
- **Motor Control:** The Arduino must control three brushless motors (2 side, 1 vertical) based on commands relayed from the Jetson Nano.

3.2.2 Non-Functional Requirements

- **Power Stability:** The system must handle burst currents up to 100A using the 4S battery.
- **Latency:** Control lag and video delay must be minimal to ensure safe navigation.
- **Durability:** Wiring must be fully integrated and waterproofed for marine environments.

3.2.3 Hardware Requirements

- **Processing:** Jetson Nano (4GB), Laptop.
- **Vision:** CSI v1.3 Camera.
- **Control:** Arduino, USB Joystick.
- **Power:** 4S 18Ah Battery, 36V-to-5V Buck Converters.
- **Propulsion:** 3x Brushless Motors + ESCs.
- **Connectivity:** Cat 6 Ethernet Cable.

3.2.4 Software Requirements

- VS Code, Arduino IDE.
- Python for system integration and data processing
- YOLO11 object detection model for coral detection and health-state analysis
- Streamlit, Grafana, InfluxDB.

3.3 Constraints

- Tether length restricts maximum depth and distance.
- Variability in water visibility and current conditions.

CHAPTER 4: Architecture and Design

4.1 Overview

MSR is designed as a tethered ROV. The architecture facilitates a "split-brain" approach: the mechanical control and video capture occur underwater, while the heavy AI processing and user interface reside on the surface laptop.

4.2 Software Architecture

The software architecture follows a layered and distributed design to improve system reliability, scalability, and real-time performance.

4.2.1 Logical View

The logical view illustrates the functional components and their interactions. The system is divided into the following logical layers:

- **Sensing Layer:** CSI Camera captures video; Jetson Nano encodes and transmits it via Ethernet.
- **Processing Layer (Surface):** Laptop receives video, runs YOLO v11 inference, and processes Joystick inputs.
- **Control Layer**: Python scripts send motor commands -> Ethernet -> Jetson Nano -> USB/Serial -> Arduino -> Motors.
- **Presentation Layer:** Streamlit web app for video/AI overlay; Grafana for metrics.

4.2.2 Process View

The process view describes the runtime behavior of the system and concurrent task execution. The main processes include:

- **Video Stream:** Camera -> Jetson -> Ethernet -> Streamlit (Laptop).
- **AI Inference:** Streamlit Frame -> YOLO v11 -> Detection of corals.

  Command Loop: Joystick Input -> Python Script -> Jetson Relay -> Arduino -> Motor Actuation.

4.2.3 Physical View

The physical view maps software components to hardware units:

- **Surface Station:**
  - **Laptop:** The primary computing unit, executing the YOLO model and processing control commands.
  - **USB Joystick:** Provides manual input for navigation.

- **Tether:**
  - **Cat 6 Ethernet Cable:** Serves as the high-speed data link for video streaming and command transmission.

- **Underwater Vehicle:**
  - **Jetson Nano (4GB):** Responsible for encoding video and relaying data to the surface; it does not perform AI processing.
  - **Arduino Board:** Manages motor signals and hardware interfacing.
  - **Power System:**
    - **Main Battery:** 4S (14.8V) 18Ah Li-ion battery with size of (8 cm x 14 cm; Weight: 1.18 kg) which include a protection circuit and its capable of 100A instantaneous and continuous power output to support high-torque motor maneuvers.
    - **Monitoring:** Features a built-in battery percentage display on the casing for easy visual verification of charge levels before deployment.
    - **Charging System:** A 2A / 16.8V charger designed for safe, slow charging. It includes a built-in protection circuit to prevent overcharging and ensure battery longevity.
    - **Voltage Regulation:** Includes a 36V-to-5V voltage reducer (buck converter) to step down the battery voltage for the Jetson Nano and Arduino logic circuits.

  - **Propulsion:** Three brushless motors (two side thrusters for steering/movement, one vertical thruster for depth).

Communication between the Arduino and Jetson Nano is achieved through a serial or USB interface.

4.2.4 Component Details

- **Propulsion and Motor Control Unit**

The vehicle utilizes three brushless thrusters (two horizontal, one vertical) to enable 3-DOF movement (Forward/Reverse, Yaw, Heave). The motors are driven by integrated Electronic Speed Controllers (ESCs) that interpret PWM signals from the Arduino.

❖ **Motor Specifications:**

- **Voltage Range:** 12-24V (Supports 3S-6S LiPo).
- **Optimal Operating Voltage:** 16V (4S LiPo configuration, matching the project battery).

- **Max Power & Current:** 390W peak power with a maximum current draw of 17A per motor.
- **Physical Dimensions:** 75mm x 75mm; Weight: 178g.
- **Cable Length:** 330mm.

❖ **Electronic Speed Controller (ESC) Features:**

- **Control Logic:** Two-way control allowing forward (PWM: 1500µs–2000µs) and reverse (PWM: 1500µs–1000µs) thrust.
- **Environment Note:** These specific prototype motors are rated for freshwater environments. (Note: For long-term deployment in the saline conditions of Aqaba, upgraded seawater-rated bearings would be required).

- **Vision System**
  CSI v1.3 Camera Captures underwater images and video streams used for coral detection and visual inspection.
- **AI Processing Unit**
  Runs a YOLO11 object detection model to detect coral regions, localize them using bounding boxes, and identify coral health conditions based on predefined classes.
- **Data Management Unit**
  InfluxDB stores historical data, linked to a Grafana dashboard.

- **Power Consumption Analysis**
  To ensure the vehicle meets the operational requirements for reef monitoring, a detailed power budget was calculated for three distinct operational profiles. The calculations assume a constant base load of **0.8A** for the electronics (Jetson Nano, Arduino, and Sensors) and varying loads for the three ApisQueen U1 thrusters.

Table 4.1: Power Consumption and Estimated Endurance

Operational Profile	Throttle %	Current per Motor	Total Motor Load (3x)	Electronics Load	Total System Load	Est. Endurance (18Ah)
Max Speed	100%	17.0 A	51.0 A	0.8 A	51.8 A	~20 Minutes
Medium Speed	25%	4.25 A	12.75 A	0.8 A	13.55 A	~1 hr 20 min
Minimum Speed	10%	1.70 A	5.10 A	0.8 A	5.90 A	~3 hours

4.3 Software Design

The software design follows a modular approach, allowing independent development and maintenance of system components.

Arduino (C): Handles low-level motor control by converting received commands into PWM signals for the thrusters.

Jetson Nano (Python): Functions strictly as a communication bridge. It streams video to the surface and relays control data to the Arduino without performing onboard computing.

Processing & AI: Runs Python scripts to capture joystick inputs and execute the YOLO v11 model for real-time coral classification.

Streamlit: A web app displays the live feed and detection results.

InfluxDB: Stores time-series data of detection events and system metrics.

Grafana: Connects to the database to provide a visual dashboard for monitoring trends and post-mission analysis.

CHAPTER 5: Implementation and Testing

5.1 Description of Implementation

MSR implementation focuses on seamless integration between the surface laptop and the submerged hardware. The electrical system is powered by a high-capacity 4S 18Ah battery. To ensure reliability, two separate buck converters step down the voltage to 5V for the Jetson Nano and Arduino, isolating the logic power from the high-current motor draw.

Control signals generated by the USB joystick are processed by Python scripts on the laptop and sent via the Cat 6 Ethernet cable to the Jetson Nano. The Jetson relays these signals to the Arduino, which controls the three brushless motors. Simultaneously, the video feed travels up the Ethernet cable to the Streamlit application, where the YOLO v11 model performs real-time detection of coral conditions.

Testing is conducted in stages, beginning with dry bench testing, followed by controlled water tests to verify propulsion, stability, and AI inference performance.

5.2 Programming Languages and Technologies

The system implementation uses the following programming languages and technologies:

- **C:** for embedded programming on the Arduino platform.
- **Python:** for high-level system control, data processing, and AI integration.
- **Linux:** as the operating system for the Jetson Nano.
- **YOLO11:** for real-time coral detection.
- **OpenCV:** for image capture and preprocessing.
- **Arduino IDE:** for firmware development.
- **Serial/USB:** communication for data exchange between Arduino and Jetson Nano.
- **InfluxDB & Grafana:** For time-series data storage and visualization.
- **Streamlit:** For the operator's front-end interface.

These technologies were selected to ensure real-time performance, ease of development, and compatibility with embedded AI platforms.

5.3 Implementation Details

Electrical Integration:

The system features fully integrated wiring, managing the high current (100A burst) capabilities of the battery safely.

Communication and Control Protocols: To ensure low-latency performance, the system utilizes specific User Datagram Protocol (UDP) channels over the Ethernet tether:

- **Video Transmission:** The video feed is captured by the Jetson Nano and streamed to the surface laptop on **UDP Port 5002**. Secure Shell (SSH) is used to remotely initiate and manage this stream.
- **Motor Control:** Navigation commands are generated using an **Xbox Controller** connected to the surface laptop. These inputs are processed and sent to the Jetson Nano on **Port 5005** to trigger motor actuation.

Embedded Control Implementation

The Arduino firmware interfaces with Electronic Speed Controllers (ESCs) to manage the three brushless motors. It translates navigation commands relayed by the Jetson Nano (received from the surface laptop) into PWM signals, enabling precise differential steering and depth control.

Vision and AI Implementation

The Jetson Nano streams live footage from the CSI camera directly to the surface laptop via Ethernet. The laptop executes the YOLO v11 model, processing frames in real-time to detect coral, assign health labels, and overlay bounding boxes on the video feed.

YOLOv11 Model Development

The AI model was trained using the YOLOv11 Object Detection (Fast) architecture. This specific model variant was chosen to minimize latency while maintaining high detection accuracy.

Dataset Acquisition and Splitting The model was trained on a custom dataset sourced from Roboflow Universe (Coral-vwaqq). The dataset consists of 936 total images of coral environments, annotated for health conditions. To ensure rigorous evaluation, the dataset was split as follows:

- **Training Set:** 864 images (92%)
- **Validation Set:** 54 images (6%)
- **Testing Set:** 18 images (2%)

Preprocessing and Augmentation To maximize model robustness in the underwater environment—where camera angles and orientation vary significantly—extensive preprocessing and augmentation techniques were applied during the training pipeline:

- **Preprocessing:**
  - **Auto-Orient:** Applied to strip EXIF orientation data.
  - **Resize:** Images were resized to fit the YOLOv11 input specifications.
- **Augmentation:**
  - **Flip:** Horizontal and Vertical.
  - **90° Rotate:** Clockwise, Counter-Clockwise, and Upside Down.
  - **Crop:** Random zoom between 0% (Minimum) and 20% (Maximum).
  - **Rotation:** Random variation between -15° and +15°.
  - **Shear:** ±10° Horizontal and ±10° Vertical.

Data Handling and Visualization

Detection results are processed entirely on the surface station. A Streamlit application displays the live video interface, while InfluxDB logs classification data. A Grafana dashboard connects to this database to visualize real-time metrics and historical trends.

5.4 Testing Approach

Testing is performed in multiple stages:

- **Bench Testing:** Verification of wiring integrity, buck converter voltage stability, and motor response to joystick inputs.
- **Software Testing:** Validating the YOLO v11 model accuracy on the custom dataset and ensuring low-latency video streaming via Streamlit.
- **Pool Testing:** Assessing watertight integrity, buoyancy, and the ability of the 3-motor configuration to navigate against resistance. The testing confirmed the system's readiness for monitoring tasks.

These tests ensure that the system functions as intended before field deployment.

CHAPTER 6: Conclusion and Results

6.1 Conclusion

MSR successfully demonstrated a cost-effective, tethered ROV architecture for monitoring Aqaba's coral reefs. By utilizing a "split-processing" design, the system leverages a surface laptop for heavy AI computation while keeping the underwater unit compact. The integration of a Jetson Nano for communication and an Arduino for propulsion proved effective, providing local authorities with a scalable, non-intrusive tool for continuous ecosystem observation.

6.2 Results

The implementation yielded the following key outcomes regarding system stability and AI performance:

6.2.1 System Performance

- **Stable Video Transmission:** Achieved low-latency streaming via the Cat 6 Ethernet tether, ensuring effective manual control.
- **System Endurance:** The 4S 18Ah power system demonstrated flexible operational endurance. Under maximum stress (high-speed transit), the system provides a minimum runtime of ~20 minutes, while standard inspection and hovering modes extend the operational window to between 1.5 and 3 hours.
- **Data Visualization:** InfluxDB and Grafana successfully logged detection events and displayed live trends for immediate post-mission analysis.

6.2.2 AI Model Metrics

The YOLOv11 model successfully detected and classified coral health (Healthy, Stressed, Bleached, Dead) in real-time. The performance metrics on the test dataset were as follows:

- **mAP@50:** 77.5%
- **Precision:** 81.1%
- **Recall:** 72.8%

These results confirm that the model is highly capable of identifying coral conditions with a strong balance between false positives (Precision) and missed detections (Recall).

6.3 Future Work

To further enhance the capabilities of the MSR, future development will focus on:

1. **Autonomous Navigation:** Implementing visual SLAM (Simultaneous Localization and Mapping) to allow the vehicle to follow transect lines automatically without constant manual input.
2. **Wireless Surface Buoy:** replacing the direct laptop tether with a floating buoy that transmits data via Wi-Fi to a shore station, increasing the operational range.
3. **Expanded Sensor Suite**: Integrating additional environmental sensors (such as pH, nitrate, and turbidity probes) to correlate quantitative water chemistry data directly with the visual health status of the corals.

4. **Multi-Model AI Deployment:** Developing and deploying additional deep learning models to solve other environmental problems, such as automated waste detection to identify pollution sources, invasive species tracking, and marine biodiversity classification.

6.4 References

[1] A. Raphael, Z. Dubinsky, N. S. Netanyahu, and D. Iluz, "Deep Neural Network Analysis for Environmental Study of Coral Reefs in the Gulf of Eilat (Aqaba)," *Big Data and Cognitive Computing*, vol. 4, no. 3, p. 19, 2020. [Online]. Available: https://doi.org/10.3390/bdcc4030019

[2] G. Jocher, A. Chaurasia, and J. Qiu, "Ultralytics YOLO," Version 11.0, [Software], 2024. [Online]. Available: https://github.com/ultralytics/ultralytics

[3] M. Mahmoodi, et al., "YOLO-Coral: A Real-Time Object Detection Model for Coral Reef Health Assessment," *Ecological Informatics*, vol. 75, p. 102047, 2023. [Online]. Available: https://doi.org/10.1016/j.ecoinf.2023.102047

[4] N. J. M. Alves, et al., "ArduinoSub: A Low-Cost ROV Kit for Ocean Engineering Education," in *Proceedings of the IEEE Global Engineering Education Conference (EDUCON)*, Porto, Portugal, 2020, pp. 1022–1029. [Online]. Available: https://ieeexplore.ieee.org/document/9125345

[5] P. Noprianto, "Real-time Monitoring Development Board based on InfluxDB and Grafana for IoT Applications," *International Journal of Computer Applications*, vol. 180, no. 14, pp. 12–18, 2021. [Online]. Available: https://doi.org/10.5120/ijca2021921073

[6] Streamlit Inc., "Streamlit: The fastest way to build and share data apps," 2024. [Online]. Available: https://streamlit.io

[7] T. Beermann, "Implementation of Distributed Computing Monitoring Dashboards using InfluxDB and Grafana," *EPJ Web of Conferences*, vol. 245, p. 03031, 2020. [Online]. Available: https://doi.org/10.1051/epjconf/202024503031

[8] G. Conte, D. Scaradozzi, and D. Mannochi, "Development of a Low Cost, Modular AUV for Shallow Water Applications," in *Proceedings of the 24th Mediterranean Conference on Control and Automation (MED)*, 2016. [Online]. Available: https://ieeexplore.ieee.org/document/7535930

[9] R. A. Wahab, et al., "Coral Reef Health Monitoring Using Deep Learning: A Review," *Journal of Marine Science and Engineering*, vol. 10, no. 11, p. 1765, 2022. [Online]. Available: https://doi.org/10.3390/jmse10111765

[10] NVIDIA Corporation, “NVIDIA Jetson Nano Developer Kit User Guide,” 2023. [Online]. Available: https://developer.nvidia.com/embedded/jetson-nano-developer-kit

[11] O. Hoegh-Guldberg, et al., “Coral Reefs Under Rapid Climate Change and Ocean Acidification,” *Science*, vol. 318, no. 5857, pp. 1737–1742, 2007. [Online]. Available: https://doi.org/10.1126/science.1152509

[12] Transnational Red Sea Center, “The Red Sea: A Unique Refuge for Coral Reefs,” *TRSC Scientific Reports*, 2023. [Online]. Available: https://trsc.org/science

[13] Roboflow Universe, "Coral-vwaqq Dataset," available at: https://universe.roboflow.com/for-work-bu6kl/coral-vwaqq/dataset/2