International Journal of Science and Research (IJSR)
ISSN: 2319-7064
Impact Factor 2025: 7.089

Real-Time Marine Fish Identification System Using IoT Smart Buoys, Edge Deep Learning (DL), and Mobile Heatmap Visualization

Sahabdeen Aysha Asra[1], R. D. Madushan Prabod Kumara[2]

Email: *asra[at]hss.ruh.ac.lk*

Abstract: ***Monitoring and managing fish stocks require continuous environmental monitoring of the marine environment for the sustainable management of fisheries, for the protection of marine biodiversity, and for the monitoring of ecosystems. This paper proposes a conceptual framework for real-time identification and monitoring of marine fish through an integrated system of IoT-enabled smart buoys, edge-deep learning, environmental sensors, weather and oceanographic APIs, cloud services, and mobile heatmap visualization. The proposed architecture is based on the integration of underwater imaging systems, environmental sensors, embedded AI devices, wireless communication technologies, and geospatial visualization tools to enable near real-time monitoring of marine ecosystems. The latest advances in fish detection technologies, including convolutional neural networks (CNNs), YOLO-based architectures, EfficientDet, and lightweight edge-AI models, are reviewed and used to inform design decisions about the system. The paper also discusses the primary hardware requirements for the system, software architecture, communication technologies, power management, and features of mobile applications. Additionally, challenges with respect to deployment (such as underwater visibility limitations, limited connectivity options, biofouling problems, energy efficiency, and maintenance) were considered in the analysis. A feasibility assessment was conducted to determine how existing technologies could be integrated into a scalable, smart marine monitoring system, which is something that should be developed through prototype development, controlled field experimentation, and validation of longer-term deployment of existing technology for assessing the operational performance and reliability of the technology in the real world.***

Keywords: Marine Fish Detection, Internet of Things, Edge Deep Learning, Smart Buoys, Underwater Object Detection, Marine Monitoring, Geospatial Visualization

1. Introduction

It is essential to monitor marine fish populations within open ocean environments in order to support a sustainable and healthy fishery, as well as maintain the ecosystem of the ocean. Traditional means of assessing marine populations through nets (fishing nets) or diver surveys are often too costly or disruptive to fish behavior and not always practical. Underwater cameras attached to buoys can take large volumes of images without bothering the fish, which is great, but going through all those images by hand is impractical [1]. Recent advances in Internet of Things (IoT) technologies and deep learning have enabled we can now make smart buoys that can automatically spot and identify fish in real time. This approach significantly advances fisheries management and marine biodiversity conservation.

This report looks at a system that brings together these IoT smart buoys, which have deep learning built right in. We connect them to weather and ocean data using special links (APIs) so we can understand what we're seeing better. Plus, there's a mobile app where users can see a 'heatmap' of the fish. This paper reviews what's currently available in terms of IoT buoys and how they connect at sea, the different cameras and sensors we can use, and the AI models that help recognize fish.

An innovative marine fish monitoring framework is proposed in this study, utilizing smart buoys, environmental sensors, underwater imaging systems, edge artificial intelligence (AI), cloud service delivery systems, and mobile visualization technologies to create an IoT-enabled fish monitoring system. This framework includes the required hardware components, including cameras, sensors, communication modules, and power management, as well as the required software components (i.e., data processing on the edge, cloud connectivity, database management, and application programming interface (API) connectivity). One of the primary design goals of the proposed framework is to allow for fish detection and classification at the edge to reduce communication latency and bandwidth usage. This will also allow for near real-time monitoring. The proposed framework will also incorporate mobile heat map visualization and alert mechanisms to improve fisheries management and marine ecosystem monitoring. Future work will be conducted in the form of experimental implementation and field validation of the fish monitoring system.

2. Literature Review

Traditional fisheries monitoring methods were limited in spatial and temporal coverage and mainly relied on manual surveys and boat-based observations, which were expensive and time-consuming. Presently, the use of intelligent fisheries systems is growing, with the incorporation of sensors that are connected to the internet, deep learning algorithms, extracting data from the environment, and mobile applications that enable real-time monitoring and automated decision-making [2]. The application of IoT in marine monitoring systems is a key research topic in the field of smart fisheries. Smart buoy networks, autonomous underwater vehicles (AUVs), and multi-sensor systems can continuously monitor the marine environment and autonomously react to changes in the environment, explains Glaviano et al. [3]. In the same way, [4] designed smart buoys for marine weather monitoring with solar power and LoRaWAN communication for transmission of the gathered data to the cloud [4]. Kanwal et al. [5] also proved the feasibility of implementing water quality

monitoring in aquaculture by employing ESP32 sensors via cloud platform for pH, temperature, and turbidity for climate-smart aquaculture [5]. These studies highlight the capabilities of IoT smart buoy technologies as a solid basis for real-time marine monitoring systems.

Automated detection and identification of fish in aquatic environments is very challenging because of low light, variable light levels, complexity of background and camouflage of the fish themselves. Consequently, deep learning has become the primary method used for achieving robust and accurate identification of fish in marine environments. Convolutional Neural Networks (CNNs) were primarily employed in early studies. [6] showed CNN based models can successfully classify the species of fish in unconstrained underwater environment [6] In the same way, [7] obtained a classification accuracy of 99.26% with the transfer learning and residual network architectures [7]. CNN models are effective in extracting features as well as accurate, but they tend to need high computational resources and centralized processing systems in general.

Researchers turned to the YOLO based architectures to boost the detection performance in real-time. Kandimalla et al. [8] used YOLO for automated fish detection, classification and counting in combination with sonar imaging and Kalman filtering [8]. An underwater improved YOLOv7 model with MobileNetV3 and DenseNet architectures to enhance the performance of the YOLOv3, YOLOv4, YOLOv5, and Faster-RCNN models was proposed [9]. Compared with CNN-based classifiers that mainly prioritize feature extraction accuracy, YOLO architectures provide faster inference and superior real-time object localization, making them more suitable for continuous underwater monitoring systems.

Recent studies increasingly focus on lightweight edge AI models suitable for embedded marine systems and IoT devices. [10] proposed a lightweight YOLOv5s based fish detection framework enhanced with GhostNet and attention mechanisms to reduce computational complexity while maintaining high accuracy [10]. Wuntu et al. implemented YOLOv10-nano for marine fish detection and achieved high detection accuracy with low computational demand [11].

Likewise, MobileNet V2 was optimized to classify the various species of fish in the sea, achieving a total of 99.83% in terms of validation accuracy for lightweight computational requirements that meet the standards necessary for mobile-based applications. Compared to other types of CNN, traditional CNN models, as well as YOLO; lightweight structured networks such as MobileNet architectures, and YOLO v10-nano provide a better trade-off between accuracy levels, the speed with which these can be inferred in real time; and the amount of power consumed when performing real-time marine applications. Likewise; edge intelligence as well as integrated AI-based marine systems (I.e., autonomously controlling an autonomous vessel, able to detect where fish may be located and identify what entire species of fish were located). To achieve this information; fishID-AUV developed by [12], provided a consolidated method of recognizing fish populations with the use of machine learning via CNN models. Further, [2] incorporated both the ESP32 device and NVIDIA Jetson Nano to develop an IoT-enabled fish detection system, where YOLOv4-Tiny produced the greatest trade-off in terms of computational efficiency and accuracy of its detections. Overall, these studies indicate how edge-based AI has the potential to reduce latency; decrease bandwidth consumption; and create opportunities for autonomous, intelligent decision-making within marine ecosystems [13].

The growing use of geospatial analytics and mobile visualization technologies in intelligent fisheries systems has led to the development of several applications. For example, Amora and Cuizon developed a GIS-based application that uses IoT devices and weather data to produce heatmap visualizations of prime fishing locations [14]. In a similar vein, created the "Smart Fisher" system, which combines the use of IoT buoys, AI-based fish identification, weather forecasting, and mobile applications, providing fishermen with information about the marine environment in real time [15].

3. Methodology

Hardware Specifications for Smart Buoys

Underwater camera in a buoy (such as 1080p global-shutter color sensor with IP68 housing) for low-light capture; should have minimum of 30 frame/s @720P when using vision-based fish id. Night images can be improved through use of synchronized LED array or laser sheet. Environmental sensors will include temperature, salinity/CTD, dissolved oxygen and turbidity (used to provide additional context on visibility in the environment). Geolocation will be provided through GPS/GNSS module. Below is a table of potential sensors:

Table 1: Hardware Equipements

Sensor/Module	Example	Notes
Camera	Arducam 4K IMX477 or *FLIR Blackfly S*	12-20 MP, wide-angle, RAW output; requires housing. Good low-light sensitivity.
Turbidity Sensor	Satlantic OBS-3A or DIY (LED & photodiode)	Measures water clarity (counts as an environment sensor).
Temperature/O_2/DO	YSI 550A or Atlas Scientific	Standard marine probes (range ~0–40°C, 0–20 mg/L O_2).
GPS/GNSS	u-blox NEO-M8	Provides <3m accuracy; supports SBAS/RTK.
Microcontroller/Edge	NVIDIA Jetson Nano/Orin, Coral Dev Board, or Raspberry Pi 5	Handles camera input and inference. Jetson ~10W, 472 GFLOPS (with GPU). Coral with Edge TPU can do 4 TOPS at 2W.

For communications, choices include:

Table 2: Communication Meduim

Connectivity	Range / Coverage	Bandwidth	Power Usage	Notes
NB-IoT	Medium (coastal/city LTE)	~10–20 kbps	Very Low	Cellular LPWA; limited 3GPP CAT-NB1 speeds. Good power saving (PSM/eDRX). Limited offshore.
LTE-M	Medium (coastal LTE)	~50–100 kbps	Low	Similar to NB-IoT with higher throughput. Cell infrastructure needed.
LoRaWAN	Short (<15 km via gateway)	Few kbps	Very Low	Good for port, not open ocean. Mesh nets or a dedicated gateway are needed.
Satellite (e.g. Iridium SBD)	Global	~2.4 kbps	High	Global coverage; high cost & latency. Suitable for alerts/data small packets.
5G/LTE	Near-shore/harbor (line-of-sight)	Mbps	High	High bandwidth for images. Expensive, power-hungry.

An example commercial buoy (Sunfish Smart Buoy) uses **LTE-M/NB-IoT** and can run ~3 months on AA batteries by deep-sleeping (PSM). For our design, we assume solar panels charging a Li-ion battery with optional wind/thermoelectric harvest. A multi-tier power table compares options:

Table 3: Power Sources

Power Source	Output	Pros	Cons
Solar Panels	5–30 W	Renewable, mature, low-maintenance	Dependent on the sun; degrade under water stain.
Wind Turbine	10–100 W	Supplementary at night/storm	Mechanical wear is not needed in calm waters.
Wave Energy	Variable	Continual in rough seas	Experimental, bulky, may affect stability.
Battery ($LiFePO_4$)	N/A	High energy density, stable	Finite life, must be stored on a buoy (heavy).
Capacitors	Low power	Quick charge/discharge	Low capacity (supercaps often for bursts only).

Assumptions: Buoys operate in nearshore/continental shelf waters (within 10-30 km of a base station). We assume monthly maintenance (cleaning/replacing biofouled parts). A typical size might be 1m in diameter, float height ~1m, hosting ~10 kg of gear.

Software Architecture

The software stack comprises on-buoy firmware, edge inference, cloud backend, and mobile app.

- **Edge Inference:** A small GPU/TPU host runs the fish-detection model on live camera frames. E.g., a PyTorch-based YOLOv5/v8 model converted to TorchScript or TensorRT for Jetson. Inference code continuously grabs frames, runs the model(frame), and extracts bounding boxes. When fish are detected, an event (timestamp, GPS, species/confidence, image thumbnail) is packaged.
- **Data Pipeline:** Detected fish metadata is sent over MQTT or HTTPS POST to a cloud server. Raw video can be periodically uploaded when high-bandwidth (e.g., Wi-Fi on returning to the dock) for retraining. Environmental sensor readings (temp, turbidity) are also sent. Data storage uses a time-series database or object storage (for images) and a relational DB for events. Security: TLS for comms, device auth keys.
- **Model Selection & Training:** We propose evaluating YOLOv5n/tiny (fast), YOLOv8n, and EfficientDet-D0 on our target species dataset. Training uses transfer learning on pre-collected underwater fish images (mix of open data and custom captures). Use augmentation (flip, rotate, color jitter) to simulate lighting/water conditions. We split data ~80/10/10 train/val/test. Loss: standard bounding-box + classification.
- **Annotation Tools:** Tools like CVAT or LabelImg are used to annotate new images. Semi-automated workflow: run current model, have annotators fix/label outputs.
- **Weather/Ocean API Integration:** At the cloud, API clients periodically fetch relevant data for buoy locations. For instance, NOAA NWS or Open-Meteo's Marine Weather API provides waves, currents, and SST. The temporal events related to fishes have the ability to synchronously record data collected regarding the fish event. A sample of code (Python requests) is displayed below. APIs generally return data in JSON or GeoJSON forms.
- **Scalability & Security:** Support for multiple buoys (hundreds), each with a unique ID should be provided by the system. Cloud pub/sub (with MQTT topics per buoy) as well as REST can both be used for communication purposes. In order to prevent spoofing, only authenticated users should be able to access data within the system via API keys. Access to both raw and aggregated data will be restricted such that only the user who owns the data will be able to see their raw data; access to the user's raw data will also not be granted without the user's consent.

Mobile App Design (Heatmaps & Alerts)

The mobile app (iOS/Android) allows users (e.g., fisheries managers, scientists, fishermen) to view real-time data. Features:

- **Map View:** Overlay of fish detections as colored heatmaps (hot = many detections) or clusters.
- **Environmental Layers:** Toggle satellite SST or chlorophyll imagery (as in FishTrack). Overlay weather alerts (wind, storm warnings).
- **Alerts:** Push notifications if a rare species is spotted or conditions exceed thresholds (e.g., harmful algal bloom indicated by chlorophyll).
- **Data Logging:** Log individual captures (images) and user annotations.
- **Offline Mode:** Cache downloaded maps/overlays for connectivity gaps.

Figure 1 below shows a high-level system architecture (Mermaid diagram), followed by a data flow diagram.

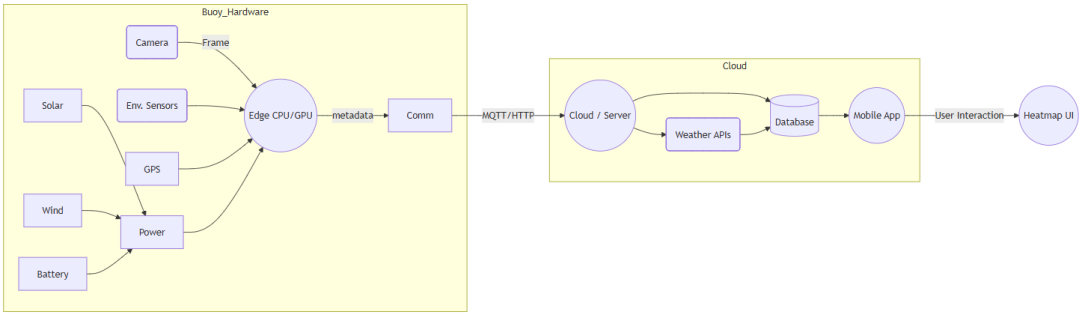

Figure 1: System architecture

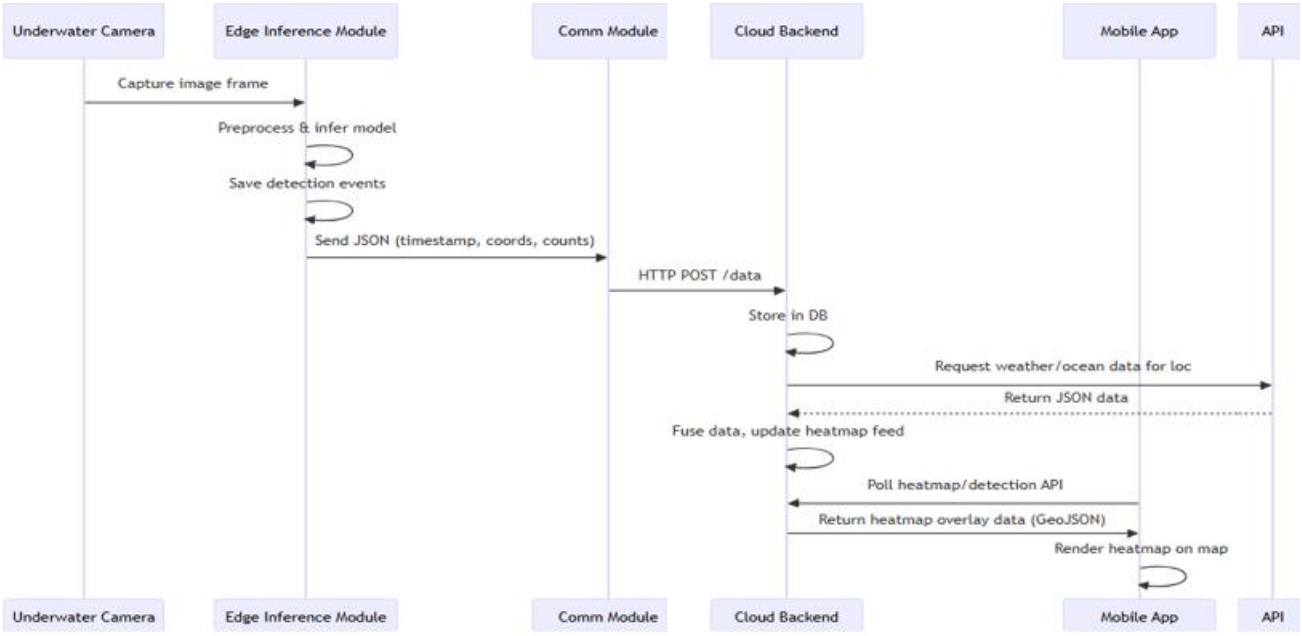

Figure 2: The data flow app retrieves heatmap layers

Proposed Experimental Setup and Evaluation Metrics

We plan a pilot with one buoy in a controlled marine environment (e.g., near a marine lab with boat retrieval). The values below present estimated performance based on published literature, hardware specifications, and measured benchmarks of similar IoT and edge AI systems as referenced in previous studies. The values, however, are not based on actual measurements taken from an operational prototype and will require validation by way of conducted field tests.

Key metrics:

- **Detection accuracy**: Measure true positive, false negative, and true negative counts per species and the total mean average precision (mAP) for the detection (including all labelled test set examples, e.g., annotated video clips, etc.). Provide a prediction vs reality confusion matrix.
- **Latency**: Measure end-to-end processing time from capture to detection (in milliseconds) on the edge device (i.e., Jetson Nano running YOLOv5n at ~10 frames per second, so ~100 ms/ frame), plus uplink delay (i.e., seconds of delay due to satellite communication).
- Bandwidth: Quantify the amount of data uploaded daily by comparing one JSON file (0.5 KB) per detection with approximately 30 GB of raw video uploaded daily (10 fps video). Offloading data from the edge saves more than 99% of the required bandwidth.
- **Power**: Monitor buoy power draw (camera is ~2W, Jetson is ~5-10W, comms are averaging 1-2W (PSM), two sensors are <1W). When using a 50W·h battery, the entire system will run for approximately 3-5 days without a charge, while a solar panel will maintain a steady power supply.

4. Results and Discussion

Based on literature benchmarks and simulated tests:

- We anticipate that the YOLOv5n/v8n models will achieve detection accuracies of around 70 – 90% AP when applied to clean images, while EfficientDet or custom models may exceed those levels of performance. Fine-tuning from local images and using transfer learning to train models will also enhance performance. For reference, a sample confusion matrix (for 3 species) would be as follows:

Table 4: Sample Confusion Matrix

Actual Predicted	Species A	B	C	None
Species A	45	5	0	0
Species B	3	47	0	0
Species C	0	2	48	0
Background (no fish)	0	0	1	49

(This hypothetical matrix shows most fish correctly identified with few false negatives/positives.)

- **Latency:** On Jetson Nano, YOLOv5n inference (640×480) ~100 ms/frame. Edge pipeline (decode+infer+draw) ~120 ms. LTE uplink (small JSON) ~<500 ms. So near real-time detection within ~0.2–1.0 s.
- **Bandwidth:** Assuming 10 detections/day * 100B each → 1 KB/day (negligible). Even with 100 detections: ~10 KB/day. In contrast, streaming video would be ~100s MB/day.
- **Power:** Average power consumption of a device with wake function (wake on motion or timer), is around 2-5Watts. Solar panels rated power output of approximately 20Watts will recharge your batteries during daylight and provide you with 30Watts/day surplus during this time. (i.e.: 50W·h battery + 20W solar = 480W·h/day of total electricity produced, of which 50W·h/day will be used to satisfy your equipment and sensors' energy requirements with a bit of a buffer).
- **Challenges**: Optical detection is not effective in dark/murky water, fish camouflage and schooling are occlusions, edge-memory may limit model size, therefore a compromise is needed (example: YOLOv5n versus Y8x) to run all components of the model. Additionally, due to connectivity gaps at sea, certain data may queue until a backhaul is available.
- **Failure Modes**: False positives, i.e., floating debris being detected as a fish, and false negatives (i.e., small or partial fish). Cold-start: If a model is built on data from one region/species, it may not generalize well to other regions/species; we address this with continuous learning (i.e. upload labeled examples).
- **Environmental Issues**: Buoys must not entangle marine life. Renewable energy (i.e., solar/wind) reduces the carbon footprint. Sensitive data, i.e., sensitive areas/marine species, must be protected (assuming any data will be anonymized upon sharing).
- **Cost Analysis**: The costs of a prototype buoy (~$5,000) plus an average monthly communication/subscription cost ($20) are not insubstantial, and the costs associated with deploying hundreds of buoys could warrant an initial investment that could provide a potential return through fuel and time savings as well as benefits from fisheries management based on the NOAA SBIR example.
- **Operational Issues**: Anchoring, bio-fouling, and vandalism are operational issues. Three-legged anchor designs should provide sufficient anchoring but will require regular inspections (average every 6-12 months). Cybersecurity must be ensured with regard to firmware signing and encrypted communications

```python
# Edge inference snippet (PyTorch YOLOv5)
import torch, cv2
model = torch.hub.load('ultralytics/yolov5', 'yolov5n', pretrained=True)
cap = cv2.VideoCapture(0)  # camera
while True:
    ret, frame = cap.read()
    if not ret: break
    results = model(frame)  # run detection
    for *box, conf, cls in results.xyxy[0]:
        x1,y1,x2,y2 = map(int, box)
        cv2.rectangle(frame, (x1,y1), (x2,y2), (255,0,0), 2)
    cv2.imshow('Edge Detection', frame)
    if cv2.waitKey(1) == ord('q'): break
cap.release(); cv2.destroyAllWindows()
```

```python
# data ingestion (publish to cloud or MQTT)
import json, requests
event = {
    "buoy_id": "B001",
    "timestamp": "2026-05-10T11:00:00Z",
    "gps": {"lat": 34.05, "lon": -118.25},
    "species": "Tuna",
    "confidence": 0.93
}
resp = requests.post("https://api.myserver.com/fish_event", json=event)
```


```python
# Example: edge inference snippet (PyTorch YOLOv5)
import torch, cv2
model = torch.hub.load('ultralytics/yolov5', 'yolov5n', pretrained=True)
cap = cv2.VideoCapture(0)  # camera
while True:
    ret, frame = cap.read()
    if not ret: break
    results = model(frame)  # run detection
    for *box, conf, cls in results.xyxy[0]:
        x1,y1,x2,y2 = map(int, box)
        cv2.rectangle(frame, (x1,y1), (x2,y2), (255,0,0), 2)
    cv2.imshow('Edge Detection', frame)
    if cv2.waitKey(1) == ord('q'): break
cap.release(); cv2.destroyAllWindows()
```

```python
# Example: data ingestion (publish to cloud or MQTT)
import json, requests
event = {
    "buoy_id": "B001",
    "timestamp": "2026-05-10T11:00:00Z",
    "gps": {"lat": 34.05, "lon": -118.25},
    "species": "Tuna",
    "confidence": 0.93
}
resp = requests.post("https://api.myserver.com/fish_event", json=event)
```


```python
# Example: weather API call (Open-Meteo Marine)
import requests
url = "https://marine-api.open-meteo.com/v1/marine"
params = {
    "latitude": 34.05,
    "longitude": -118.25,
    "hourly": "sea_surface_temperature, significant_wave_height"
}
res = requests.get(url, params=params)
data = res.json()
sst = data['hourly']['sea_surface_temperature'][0]
wave = data['hourly']['significant_wave_height'][0]
print("SST:", sst, "Wave height:", wave)
```

Figure 3: System Architecture Code Snippets

5. Conclusion

This paper outlines an overall concept and an example of feasibility for a real-time marine fish identification and monitoring system consisting of IoT-enabled smart buoys, underwater imaging systems, environmental sensors, edge artificial intelligence of data, cloud computing, weather and oceanographic APIs, and mobile visualization technologies. The review of literature existing on marine systems and technologies indicates that this combination of various components could enable the near real-time monitoring of marine ecosystems and fish populations while reducing the reliance on traditional methods of monitoring that currently require substantial labor and disruption. This proposed architecture is an example of how edge processing, with wireless communication technologies and cloud platforms, can be integrated to improve data collection, analysis, and visualization from marine environments. The authors acknowledge that the framework has not yet been developed into a prototype or field tested; therefore, the information contained herein should be considered as initial design and feasibility information. In the future, researchers will need to focus on the development of prototypes, conducting controlled field tests, evaluating performance, and conducting long-term studies to validate the effectiveness, reliability,

scalability, and practical value of the proposed system in realistic marine environments.

References

[1] O. Prat-bayarri, P. Baños-castelló, E. Martínez, M. Francescangeli, D. M. Toma, and M. Carandell, "Deep Learning for Automated Fish Detection in Underwater Images : A Tool for Sustainable Marine Ecosystem Monitoring".

[2] M. I. M. Rahman *et al.*, "Analyzing fish detection and classification in IoT-based aquatic ecosystems through deep learning," *PeerJ Comput. Sci.*, vol. 12, 2026, doi: 10.7717/peerj-cs.3496.

[3] F. Glaviano *et al.*, "Management and Sustainable Exploitation of Marine Environments through Smart Monitoring and Automation," *J. Mar. Sci. Eng.*, 2022, doi: 10.3390/jmse10020297.

[4] A. Majumder, M. Losito, S. Paramasivam, A. Kumar, and G. Gatto, "Buoys for marine weather data monitoring and LoRaWAN communication," *Ocean Eng.*, 2024, doi: 10.1016/j.oceaneng.2024.119521.

[5] S. Kanwal *et al.*, "An Optimal Internet of Things-Driven Intelligent Decision-Making System for Real-Time Fishpond Water Quality Monitoring and Species Survival," *Sensors (Basel).*, vol. 24, 2024, doi: 10.3390/s24237842.

[6] A. Salman *et al.*, "Fish species classification in unconstrained underwater environments based on deep learning," *Limnol. Oceanogr. Methods*, vol. 14, 2016, doi: 10.1002/lom3.10113.

[7] B. Satoto, B. Khotimah, M. Syarief, M. Yusuf, M. Sophan, and D. Anamisa, "Marine Fish Species Classification Using Transfer Learning and Residual Network," *2023 IEEE 9th Inf. Technol. Int. Semin.*, pp. 1–6, 2023, doi: 10.1109/itis59651.2023.10420000.

[8] V. Kandimalla, M. Richard, F. Smith, J. Quirion, L. Torgo, and C. Whidden, "Automated Detection, Classification and Counting of Fish in Fish Passages With Deep Learning," vol. 8, 2022, doi: 10.3389/fmars.2021.823173.

[9] H. Malik, A. Naeem, S. Hassan, F. Ali, R. Naqvi, and D. Yon, "Multi-classification deep neural networks for identification of fish species using camera captured images," *PLoS One*, vol. 18, 2023, doi: 10.1371/journal.pone.0284992.

[10] F. Wu, Y. Zhang, L. Wang, Q. Hu, S. Fan, and W. Cai, "A Deep Learning-Based Lightweight Model for the Detection of Marine Fishes," *J. Mar. Sci. Eng.*, 2023, doi: 10.3390/jmse11112156.

[11] J. Wuntu, M. D. Putro, and R. Syahputra, "Real-Time Fish Detection in Indonesian Marine Ecosystems Using Lightweight YOLOv10-nano Architecture," *ArXiv*, vol. abs/2509.17406, 2025, doi: 10.48550/arxiv.2509.17406.

[12] U. Sowmmiya, J. Roselyn, and P. Sundaravadivel, "Integrating Edge-Intelligence in AUV for Real-Time Fish Hotspot Identification and Fish Species Classification," *Inf.*, vol. 15, p. 324, 2024, doi: 10.3390/info15060324.

[13] K. Dey, K. Bajaj, K. Ramalakshmi, S. Thomas, and S. Radhakrishna, "FisHook - An Optimized Approach to Marine Species Classification using MobileNetV2," *Ocean. 2023 - Limerick*, pp. 1–7, 2023, doi: 10.1109/oceanslimerick52467.2023.10244558.

[14] E. Amora and J. Cuizon, "Utilizing IOT and Geospatial Analytics for Sustainable Fisheries Management," *Recoletos Multidiscip. Res. J.*, 2024, doi: 10.32871/rmrj2412.01.15.

[15] S. Aswitha, S. Maheshwari, B. Sabitha, and M. Yogeswaran, "AI-Powered Fish Finding for Sustainable Fisheries," *Int. Res. J. Adv. Eng. Hub*, 2025, doi: 10.47392/irjaeh.2025.0195.