FishSense - iOS Application

ANUSHRI ESWARAN, University of California San Diego, USA
ZIYUAN LIN, University of California San Diego, USA
TANISHQ RATHORE, University of California San Diego, USA
CHRISTOPHER CRUTCHFIELD, University of California San Diego, USA

Accurate fish measurement and species identification are vital for conservation and fisheries research, yet existing methods are often invasive, inconsistent, or reliant on expert input. While existing alternatives use AI-driven photo analysis, they can suffer from poor accuracy due to variable image conditions and lack of depth perception. FishSense addresses this challenge by combining LiDAR-based spatial scanning with machine learning classification in a user-friendly iOS application. Using mobile LiDAR, the app captures precise, fish length measurements, while a convolutional neural network identifies species from images. These results establish a more robust and ethical method for catch-and-release practices and data collection. By integrating spatial and visual data in a mobile, FishSense sets a new standard for scalable, non-invasive fish monitoring tools, enabling broader participation in ecological research and improving the quality of citizen science data. Crucially, FishSense operates entirely offline, running AI on the edge via optimized ONNX models on Apple Metal, ensuring fast, private, and reliable performance without internet dependency. Evaluations demonstrate that FishSense achieves a median length measurement error of approximately 7.5% on all devices, and a species classification accuracy of 70.69%.

CCS Concepts: • **Computing methodologies → Computer vision problems**; *Neural networks*; • **Applied computing →** *Environmental sciences*; • **Human-centered computing** → *Mobile computing*; • **Software and its engineering** → Real-time mobile systems; Concurrent programming languages.

Additional Key Words and Phrases: LiDAR, depth sensing, machine learning, convolutional neural networks, mobile computing, iOS app development, ecological monitoring, fish species identification, Rust programming

ACM Reference Format:
Anushri Eswaran, Ziyuan Lin, Tanishq Rathore, and Christopher Crutchfield. 2025. FishSense - iOS Application. *UCSD CSE 145/237D* 1, 1 (June 2025), 14 pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

Monitoring fish populations—knowing both who's in the water and how big they are—is fundamental to sustainable fisheries management, conservation research, and recreational angling. Traditionally, scientists and anglers alike have relied on manual tools such as calipers and measuring boards, combined with expert judgment for species identification. These methods are time-consuming, labor-intensive, and can pose physical risks to fish through excessive handling. In practice, issues such as measurement error, poor lighting, and observer fatigue can introduce significant variability into collected data, undermining its reliability for long-term monitoring or high-throughput surveys.

Authors' Contact Information: Anushri Eswaran, aeswaran@ucsd.edu, University of California San Diego, CA, USA; Ziyuan Lin, zil102@ucsd.edu, University of California San Diego, CA, USA; Tanishq Rathore, trathore@ucsd.edu, University of California San Diego, CA, USA; Christopher Crutchfield, ccrutchf@ucsd.edu, University of California San Diego, CA, USA.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
© 2025 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM 2476-1249/2025/6-ART
https://doi.org/XXXXXXX.XXXXXXX

Over the past decade, computer vision systems based solely on RGB imagery have begun to automate parts of this workflow. Deep neural networks can now detect fish outlines and infer body length from a single photo. While this represents a meaningful step forward, such systems are fundamentally limited by the absence of true depth information. Their size estimates can vary significantly due to changes in camera angle, subject distance, or background clutter. Deploying these models in real-world field conditions often requires careful calibration and controlled setups—constraints that are difficult to satisfy on rivers, lakes, or small boats.

The emergence of consumer-grade LiDAR sensors on modern iOS devices presents a game-changing opportunity. These sensors allow handheld devices to capture synchronized RGB images and precise depth maps in real time, enabling direct spatial measurement without the need for additional equipment. Motivated by this capability, we developed FishSense: a fully on-device iOS application that fuses LiDAR-based geometry with convolutional neural network (CNN) classification to enable rapid, non-invasive fish monitoring. By leveraging AI on the edge—running optimized ONNX models on Apple Metal—FishSense performs all inference locally, providing estimated length (in centimeters) and species predictions within seconds, without requiring cloud connectivity or heavy infrastructure.

This edge compute approach not only enhances speed and privacy but also democratizes the science, making advanced fish monitoring accessible to a broad range of users in diverse and dynamic field conditions. Our key contributions are as follows:

- **Integrated pipeline:** We utilize Apple's ARKit LiDAR framework for head-to-tail segmentation and length computation, paired with an ONNX-converted CNN model for species classification. The entire system is orchestrated within a Rust-backed engine for low-latency performance on-device.
- **Rigorous evaluation:** We curated and annotated a diverse dataset encompassing multiple species, lighting conditions, and aquatic environments. Using this dataset, we demonstrate a median length error of 7.5% and a species classification accuracy of 70.69%, outperforming prior mobile-only methods.
- **Field-ready usability:** FishSense requires no internet connectivity and leverages only built-in hardware, reducing handling time and stress on fish. This aligns with best practices for catch-and-release fishing and enables citizen scientists to contribute high-quality ecological data from virtually any location.

The remainder of this paper is structured as follows: Section 2 reviews related work. Section 3.1 outlines our system architecture and algorithms. Section 3.2 describes our dataset collection and annotation methodology. Sections 3.2.7 and 4.5 present experimental results. Section 6 discusses key technical challenges and our milestone work. Finally, Section 7 concludes with directions for future improvement.

2 Related Work

2.1 Image-Based Fish Length Estimation

Numerous non-invasive techniques have been developed to estimate fish length using image-based analysis. Monkman et al. [10] proposed a computer vision pipeline that uses region-based convolutional neural networks (R-CNNs) to estimate the length of European sea bass from single images, automating a traditionally manual process and reducing measurement error significantly. Álvarez-Ellacuría et al. [16] expanded this direction by applying instance segmentation (Mask R-CNN) to detect fish contours in commercial landings, enabling accurate length prediction with only a few centimeters of error. In aquaculture environments, Tonachella et al. [14] developed a vision-based pipeline that achieved length measurement accuracy within 3% for gilthead seabream by integrating CNN models with depth cues.

Commercial solutions like FishTechy [6] combine AI with a proprietary "Proof Ball" reference object to let anglers measure not only length but also girth and weight in seconds via their mobile app. Unlike FishSense, which relies solely on built-in LiDAR and on-device inference to achieve sub-centimeter length accuracy without extra hardware, FishTechy requires users to carry and properly position their calibrated Proof Ball for any capture.

FishTechy has recently added a LiDAR system to their app, but the app requires Wi-Fi or data to function—a limitation that FishSense overcomes.

In more controlled environments, Almansa et al. [2] demonstrated that 3D laser scanners could reliably estimate fish biomass and size in tanks, providing a contactless alternative to calipers or boards. A related system, FishSense by Tueller et al. [15], integrated an Intel RealSense RGB-D sensor underwater to autonomously detect and measure fish. The device reported a mean error of just 0.3 cm and proved the viability of real-time 3D length estimation for ecological monitoring.

Building upon these advances, Cao et al. [4] proposed a customizable dimension measurement method based on an improved YOLOv5-keypoint framework enhanced with multi-attention mechanisms. Their method introduces flexible landmark configurations for fish dimension extraction, achieving high precision and real-time speed suitable for large-scale aquaculture applications. The approach is implemented as an online platform, MrRuler, enabling users to utilize preset models or upload custom training datasets for diverse aquaculture species.

2.2 Fish Species Identification Using CNNs

Fish species recognition has also seen substantial improvement with deep learning models. Allken et al. [1] trained CNNs to identify multiple pelagic species from underwater trawl camera images, using synthetic data to mitigate dataset limitations. Their approach performed well even under varying oceanic lighting conditions. Similarly, Hussain et al. [7] developed a 32-layer CNN architecture and achieved high classification accuracy on benchmark fish datasets, outperforming earlier shallow models. Jareño et al. [9] introduced a two-step strategy fine-tuning pretrained CNNs followed by traditional classifiers which boosted accuracy across 19 seafood species in market images.

Importantly, lightweight architectures such as MobileNet-V1 have proven effective for real-time, on-device inference. Suharto et al. [12] showed that MobileNet could achieve over 90% accuracy in freshwater fish classification, making it well-suited for mobile deployments like FishSense where computational resources are limited.

2.3 Mobile LiDAR and Depth Sensing in Ecology

Recent advancements in smartphone LiDAR technology have opened up new opportunities in ecological measurement. Tatsumi et al. [13] introduced ForestScanner, an iPhone-based LiDAR application that allows users to measure tree diameters in the field. The app demonstrated strong agreement with traditional methods ($R^2 \approx 0.96$) while requiring only 25% of the field time. Stitt et al. [11] further explored smartphone-based depth sensing by measuring nest cavity dimensions with a laser rangefinder, achieving sub-centimeter accuracy; similarly, a recent study demonstrated that the Spike handheld LIDAR device, which pairs with smartphones and tablets, can accurately and noninvasively measure woodpecker cavity entrance dimensions from distances up to 30 m, with average errors of less than 1 cm.

These examples show that mobile devices can now serve as precise spatial measurement tools. Applied to aquatic contexts, this means LiDAR-equipped smartphones can support accurate fish length estimation, especially when traditional tools are impractical. FishSense's use of iPhone LiDAR to capture depth-enhanced images offers a promising new direction for citizen-led fish monitoring.

2.4 On-Device Inference for Field Applications

Running machine learning inference directly on mobile devices offers numerous advantages for ecological tools: low latency, offline functionality, and data privacy. However, this approach presents challenges, including limited compute power and energy constraints. Frameworks such as TensorFlow Lite and Core ML have made it possible to deploy compact models (e.g., MobileNet, EfficientNet-lite) by optimizing for edge performance through quantization and pruning [5].

FishSense is designed to operate offline, relying on an efficient CNN architectures and Apple's neural engines to process fish measurements and species predictions in real-time. This allows users in remote aquatic settings to access core app functions without internet access—critical for field researchers and conservationists.

2.5 Citizen Science and Image-Based Monitoring

Computer vision has played a central role in scaling up biodiversity monitoring through citizen science. iNaturalist's mobile and web platforms allow users to upload organism images for automatic identification via deep learning models trained on tens of thousands of taxa [8]. Its companion app, Seek, runs entirely on-device, giving users immediate, offline species suggestions as they explore.

Wildbook [3] takes this further by identifying individual animals using computer vision on crowdsourced images. Projects like Wildbook for Whale Sharks have cataloged thousands of individuals using tourist photos and spot pattern recognition, enabling large-scale mark-recapture studies.

These platforms illustrate the power of combining user engagement with AI tools for ecological data collection. FishSense builds on this model by enabling non-experts to contribute high-quality length and species data via mobile devices, ultimately supporting large-scale fish population studies.

3 Technical Work

3.1 System Overview

FishSense operates through a streamlined multi-stage pipeline, beginning with image capture. A user initiates the process on an iOS device equipped with LiDAR sensors, typically available in the "Pro" models. On capture, the application collects both an RGB image and a depth map. This input is then passed to our Rust backend, which handles preprocessing. By default, the pipeline triggered post-capture is for fish length measurement. It utilizes the RGB data, depth map, and a head-tail detection algorithm based on fish geometry to calculate and return the fish's length.

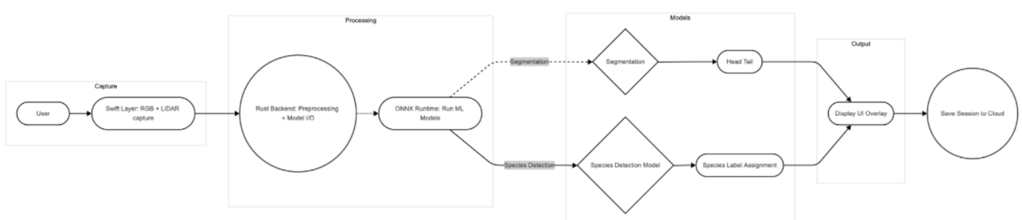


Fig. 1. System architecture of *FishSense*. The pipeline includes RGB image capture, depth sensing via LiDAR, confidence mapping, and CNN-based species classification.

If the user clicks the *Detect Species* button, the captured RGB image is preprocessed and fed into a model (running in ONNX Runtime) for species classification. This model identifies the species by comparing the image to known fish species it has been trained on. All captured data is stored locally on the device and can be pushed to the cloud at the user's discretion with a single tap.

The choice to build this as a mobile app is intentional: conventional fish length measurement methods are often invasive, time-consuming, and require specialized expertise. By leveraging the on-device processing capabilities of iOS, FishSense offers a smarter, safer, and more scalable solution. It enables real-time fish length detection and species classification directly on the device, minimizing handling and streamlining the monitoring process. FishSense is currently a mobile-only application, designed exclusively for iPhone/iPad Pro models due to the hardware requirements (LiDAR sensors) and chosen algorithms.

The application works fully offline, without requiring Wi-Fi or network connectivity (except for the optional cloud sync feature). Users can capture data in the field and later sync it to the cloud when connected to Wi-Fi. This mobile-first approach enhances efficiency, reduces reliance on trained personnel, and supports immediate, in-field data collection that can be saved and uploaded for further analysis.

3.2 Fish Length Measurement

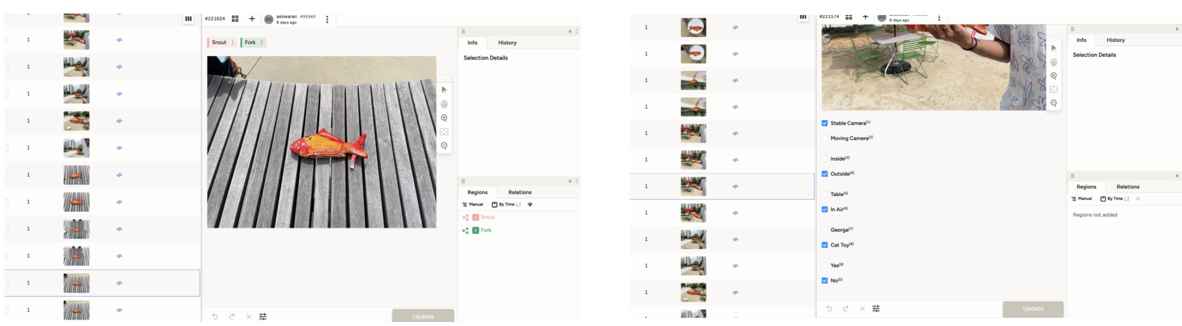
Fish length estimation begins with precise head-tail detection, using a geometry-based algorithm that analyzes fish contours from both RGB images and depth maps. Accurate localization of the head (snout) and tail (fork) is essential, as even minor deviations can lead to significant errors in the computed length.

3.2.1 Data Labeling Pipeline. To evaluate and refine our detection algorithm, we implemented a structured data pipeline. During image capture, relevant information—including the RGB image, depth map, and predicted length—is stored locally and can be synced to an S3 bucket. This enables seamless integration with a custom labeling tool for annotation.

Each image is manually annotated with the following attributes:

- Head (snout) and tail (fork) positions
- Environmental conditions (e.g., indoor vs. outdoor, static vs. moving capture)
- Subject identity (e.g., George or Barry—our cat toy test subjects)
- Pose details (e.g., whether the fish/toy was lying flat or angled)

We will be mainly using head and tail positions that has been annotated for our analysis.



(a) Label Studio interface for fish length annotation

(b) Label Studio interface for other analysis

Fig. 2. Screenshots of the Label Studio tool used for annotating fish images in our dataset.

3.2.2 Importance of the Labeling Workflow. This annotation pipeline plays a critical role in evaluating model performance and guiding improvements. By comparing predicted key points to labeled ground truth, we can identify systematic errors. These insights drive iterative enhancements to both the detection algorithm and the data capture process, ultimately improving overall model robustness.

3.2.3 Accuracy Analysis. Approximately 80% of the measurements fall within a reasonable 0–10% error range, indicating that the algorithm performs reliably in most cases. The violin plot shows a median error of around 7.5%, reinforcing the overall stability of the system.

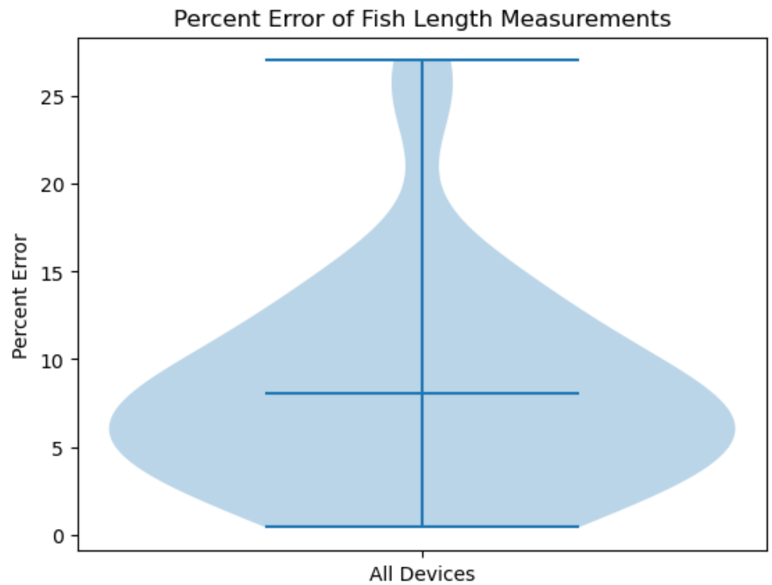


Fig. 3. Violin Plot showing percent error of fish length measurement of the all data collected collected.

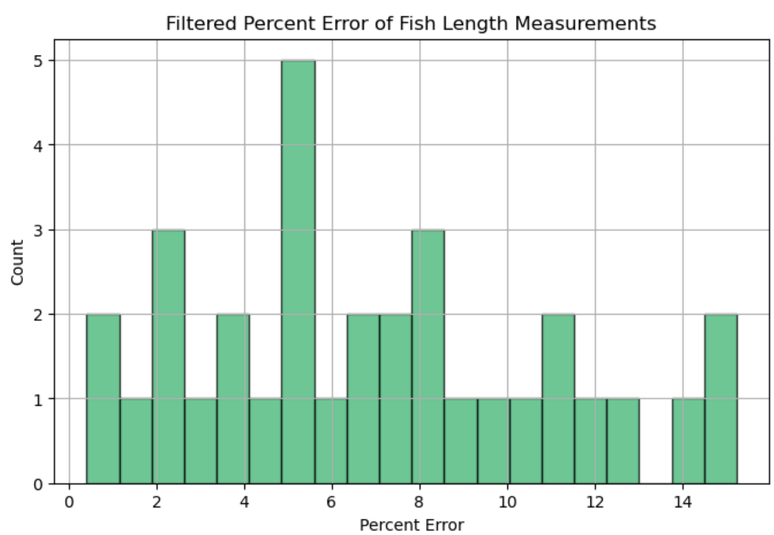


Fig. 4. Histogram plot if percent error in the fish length measurements.

The error distribution reveals two distinct clusters: a high-accuracy group within the 1–5% error range, and a moderate-accuracy group between 6–12%. This separation suggests variation in measurement conditions or device performance. Notably, cluster 1 highlights the algorithm's ability to deliver highly precise results under optimal conditions. Meanwhile, cluster 2 shows that even under less-than-ideal scenarios, the algorithm maintains acceptable accuracy.

These distinct patterns point to predictable behavior rather than random error, which raises an important question: can we identify or anticipate the conditions under which accuracy degrades? More specifically, is there a correlation between higher error rates and factors such as device type, image quality, or confidence map? Answering this could help flag unreliable measurements in real time and guide further optimization efforts.

3.2.4 Confidence Map. One of the key challenges lies in the quality and reliability of depth maps. Factors such as ambient lighting, reflective surfaces (like wet fish), and capture angle can introduce noise and affect accuracy. To address this, we analyze a confidence map that indicates the reliability of the depth data. This map assigns an `ARConfidenceLevel` value to each component in the depth map, reflecting the confidence in the scene's depth measurement.

The confidence levels are as follows:

- 0: Low confidence (less confident)
- 1: Moderate confidence (moderately confident)
- 2: High confidence (fairly confident)

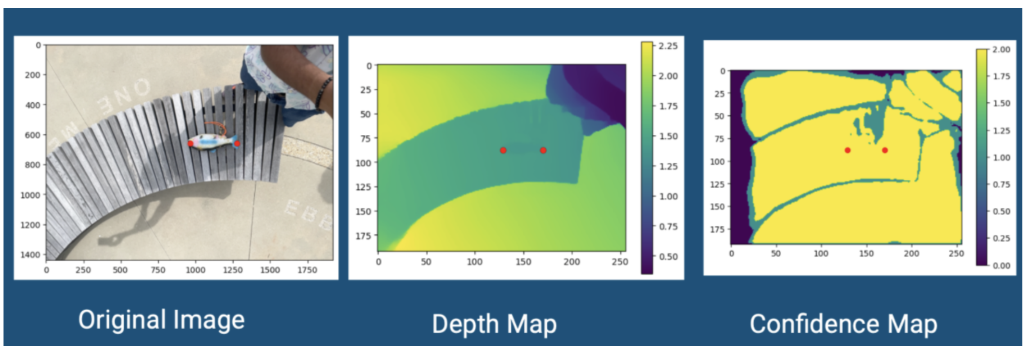


Fig. 5. Example image, depth map and its corresponding confidence map

The violin plot reveals a clear inverse relationship between confidence levels and measurement error. Low confidence measurements (level 0) show the widest error distribution, ranging from approximately −25% to +110%, with the majority of errors concentrated between 10–40%. This wide spread indicates significant uncertainty in measurements when depth data reliability is compromised.

Moderate confidence measurements (level 1) demonstrate substantially improved performance, with errors tightly clustered around 5–15% and a much narrower distribution range. High confidence measurements (level 2) show the best performance, with most errors concentrated in the 5–12% range and minimal outliers.

The confidence map serves as an effective predictor of measurement reliability. The extreme reduction in error variance as confidence increases from 0 to 2 validates the utility of confidence-based quality assessment. This relationship suggests that implementing confidence thresholds will significantly improve overall system accuracy by filtering or flagging low-confidence measurements for re-capturing.

3.2.5 Device (iPhone vs iPad). We compared measurement performance across iPhone [12 Pro and 15 Pro] and iPad devices to evaluate consistency. All devices use the same pipeline and rely on the built-in LiDAR sensor, but differences in camera alignment, processing speed, chip or physical handling could affect outcomes.

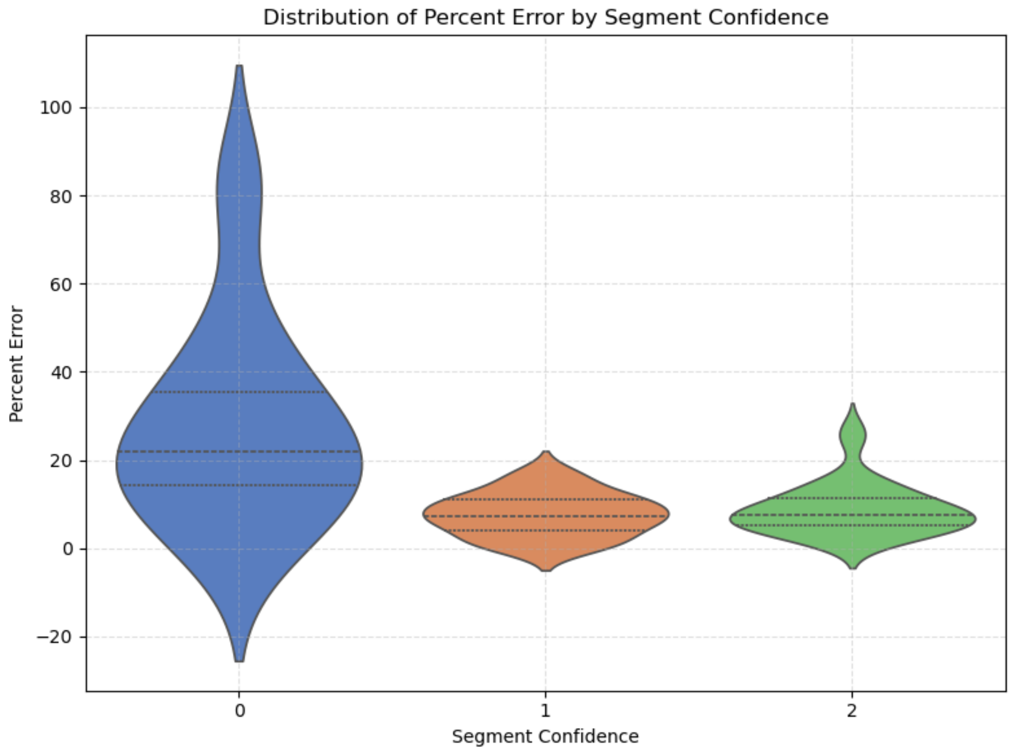


Fig. 6. Violin plot of percent error with respect to confidence score

We collected approximately 165 data points across three devices: iPhone 12 Pro, iPhone 15 Pro, and iPad Pro 11-inch (M4).

Table 1. Details of Devices Used for Data Collection

Device	OS Version	Screen Size	Processor	Data Points
iPhone 12 Pro	iOS 18.2	6.1 inch	A14 Bionic	54
iPhone 15 Pro	iOS 18.5	6.1 inch	A17 Pro	55
iPad Pro 11-inch (M4)	iPadOS 18.5	11 inch	M4	54

The iPad Pro (M4) showed the lowest percent error and smallest variance in fish length measurements, indicating superior consistency and accuracy. The iPhone 15 Pro outperformed the iPhone 12 Pro, with a lower median error and tighter error distribution. The iPhone 12 Pro exhibited the highest variability and largest percent errors, likely due to its older processor and camera system. These results are based on a small dataset, so while likely true for larger samples, some variability should be expected.

3.2.6 Head-Tail Detection Algorithm. Head and tail are initially distinguished by a heuristic comparing their position relative to the fish's convex hull. The tail point is refined by searching for the most concave point on the fish's polygon near the temporary tail.The head point is corrected using the principal eigenvector to find the hull

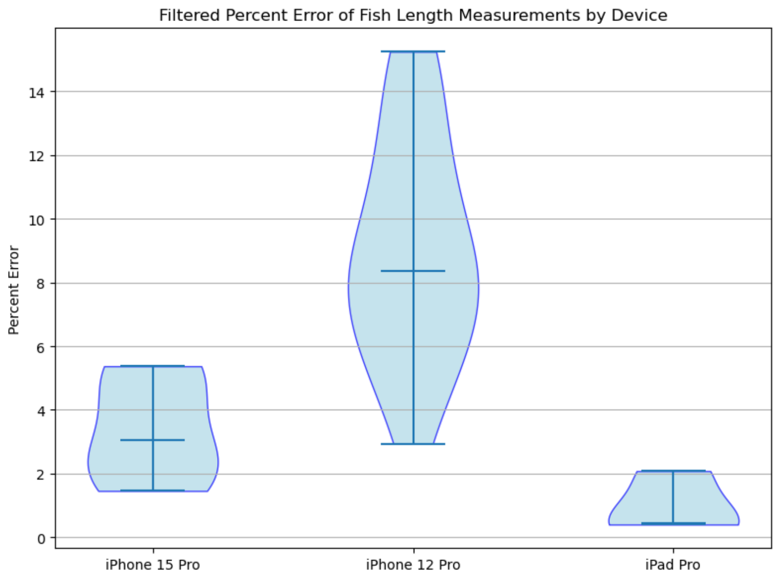


Fig. 7. Violinplot of percent error for each device

point furthest from a perpendicular line, aligned with the tail-to-head direction. This work was done by Shaurya Raswan.

3.2.7 Comparison with SOTA. Compared to FishTechy, our limited evaluation shows FishSense consistently outperforms in length accuracy, largely due to the integration of LiDAR-based depth sensing for precise geometric measurement.

Table 2. Fish Length Comparison: SOTA vs. FishSense

Sample	SOTA (cm)	FishSense (cm)	Original (cm)	SOTA Δ(cm)	FishSense Δ (cm)
1	31.24	31.3	33	-1.76	-1.7
2	35.52	32.0	33	+2.52	-1.0
3	29.44	31.0	33	-3.56	-2.0
4	0.00	32.2	33	-33.00	-0.8
5	27.67	25.4	26	+1.67	-0.6
6	31.96	25.6	26	+5.96	-0.4
7	28.73	24.9	26	+2.73	-1.1
8	31.67	26.2	26	+5.67	+0.2
9	29.22	26.4	26	+3.22	+0.4
10	31.7	27.1	26	+5.70	+1.1

4 Fish Species Classification

4.1 Model Conversion and Architecture

FishSense incorporates an offline species classification feature powered by a deep learning model adapted from the Fishial open-source classifier. The original Fishial model was trained for fish species recognition using large-scale image datasets, and we converted this model into the ONNX (Open Neural Network Exchange) format to ensure compatibility with mobile deployment on iOS. This conversion enables efficient, on-device inference using the ONNX Runtime without relying on cloud resources or an internet connection.

4.2 Preprocessing Pipeline

Prior to classification, the captured RGB image undergoes a preprocessing pipeline designed to improve model performance and maintain consistency with training conditions. The main preprocessing steps include cropping, scaling and normalization.

4.3 Classification Pipeline

Once preprocessing is complete, the image is passed into the ONNX classifier. The model performs feature extraction followed by a similarity-based matching process. Rather than returning hardcoded species labels, the classifier compares extracted features against known fish species in its training set and ranks the most similar matches. The highest-confidence prediction is then returned as the classified species.

This architecture allows the model to generalize better across fish poses and lighting conditions, leveraging feature similarity rather than overfitting to specific visual templates.

4.4 Integration into the Mobile App

The species classification runs entirely on-device, following a user-initiated tap on the "Detect Species" button. At this point, the RGB image is preprocessed and passed to the ONNX runtime environment embedded within the app. Predictions are rendered in real time and displayed to the user alongside the image.

All captured images and predictions are stored locally and can optionally be synced to the cloud for later validation and analysis.

4.5 Accuracy and Evaluation

While real-world classification accuracy is influenced by factors such as image quality, environmental conditions, and species similarity, initial results on a curated validation set are promising. We evaluated the model on a set of 60 images collected during field deployments in collaboration with the California Collaborative Fisheries Research Program and FishSense during the summer of 2024. On this dataset, the model achieved an accuracy of 70.69%. The following confusion matrix illustrates the performance across different species and highlights areas where misclassification occurred, particularly among visually similar fish types. Overall, the model performs well on species it has been trained on, making it a strong baseline for practical use. However, to expand its applicability to a broader, more global range of species, the model will require fine-tuning with additional species data or retraining on a more diverse dataset.

5 Session Management

FishSense is designed with robust session management to support both real-time and offline use cases in field environments. All captured data, including RGB images, depth maps, predicted lengths, species classifications, and associated metadata, is first stored locally on the device. This ensures that the application remains fully functional even in remote areas without internet access.

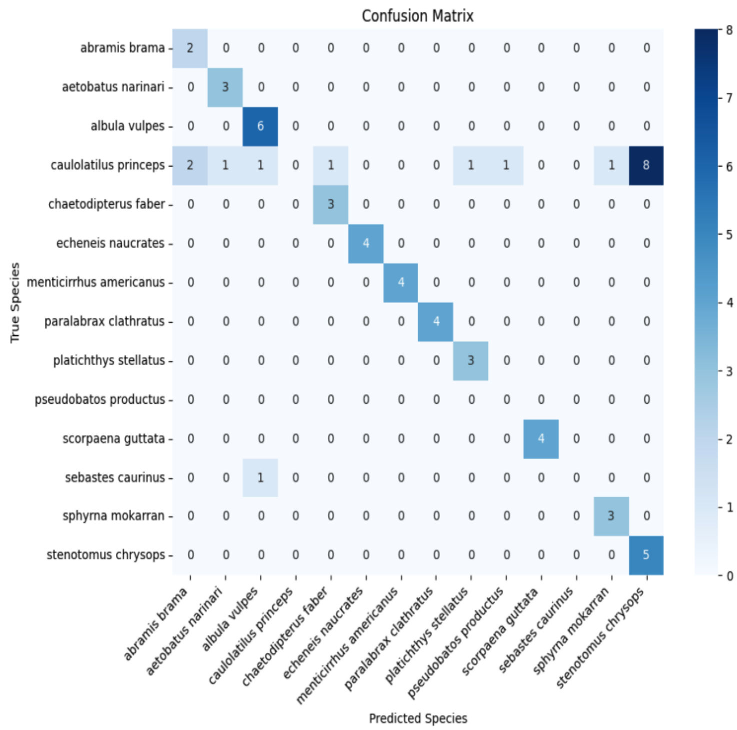


Fig. 8. Confusion Matrix of the Species Detecion model on curated CCFRP dataset

When a network connection becomes available, the user can click the "Upload Data" button and sync stored sessions to the cloud. This is accomplished via a `POST` request to an AWS Lambda function behind an API Gateway. The Lambda endpoint handles authentication, data validation, and formatting before pushing the session data into structured cloud storage. This architecture supports asynchronous data collection while maintaining data integrity and consistency.

The local-first approach guarantees uninterrupted workflow and reliability in the field, while the optional sync mechanism enables seamless backup and centralized access for further processing, analysis, or model improvement.

6 Milestones and Project Development

6.1 Original and Refined Milestones

Original Milestone Plan:

- **Week 2–Week 3:** Code reading and analysis, Swift and Rust learning.
- **Week 4:** Bug fixing, CCFRP dataset cleaning, fish head and tail algorithm understanding, and identification of dataset for training fish classification and health detection module.
- **Week 5:** Refine existing ML model, search/build models for fish classification and health detection.
- **Week 6:** Model fine-tuning, app development, and mid-quarter report.
- **Week 7:** Investigate methods to improve length detection and build a model to test classification results.
- **Week 8–Week 9:** Local and remote database implementation, and UI improvement.

- **Week 10:** Final presentation and report.

Refined Milestone Plan:

- **Week 2–Week 3:** Code reading and analysis, Swift and Rust learning.
- **Week 4:** Bug fixing, CCFRP dataset cleaning, fish head and tail algorithm understanding, and identification of dataset for training fish classification and health detection module.
- **Week 5:** LiDAR and depth map investigation and problem identification.
- **Week 6:** FishSense UI development and implementation of a new head and tail detection algorithm.
- **Week 7:** Confidence map implementation, classification module deployment, and depth map rebuilding.
- **Week 8–Week 9:** Local and remote database implementation, and UI improvement.
- **Week 10:** Final presentation and report.

6.2 Revisions and Rationale

During Week 5, we identified a critical issue with the depth map: it was incorrectly clipped at a 1-meter range, which significantly impaired the accuracy of length detection. Consequently, we shifted our focus to investigate and resolve this problem.

In Week 4, we had discovered an existing fish classification model (Fishial) that was suitable for integration into our application. This allowed us to postpone developing a custom classification model until Week 7.

By Week 6, with support from Shaurya Raswan, we successfully deployed a new head and tail detection algorithm, which prompted us to revise our project milestones.

In Week 7, we prioritized analyzing our algorithm, leveraging confidence map and rebuilding the depth map storage system. This decision was based on our evaluation that accurate length detection and reliable depth maps are more critical to the system's long-term success. Fish health detection was de-prioritized due both to these shifting priorities and the extensive data requirements needed for each species.

6.3 Problems Encountered and Solutions

- **Depth Map Clipping:** Some depth maps (from the CCFRP Summer 2024 dataset) appeared to max out at a value of 255, suggesting the LiDAR failed to detect objects beyond 1 meter. We investigated the code and discovered that the PNG transformation process clipped values beyond 1 meter. We resolved this by analyzing and updating the depth map generation pipeline to preserve the full depth information and creating our own dataset to do analysis on.
- **ONNX Model Conversion:** We faced difficulties converting our classification model to ONNX format due to unsupported operations and version mismatches. After identifying the incompatible layers and updating the model and export parameters accordingly, we were able to successfully convert it.
- **Remote Database Setup:** Initially, we were unsure which remote database to use or how to connect the local database to a remote one. After researching, we selected Amazon AWS due to its popularity and documentation. We implemented a solution using AWS Gateway to trigger a Lambda function that transmits local data to the remote database.
- **Classification Model Output Error:** Although the classification pipeline was correctly implemented, the model failed to predict species accurately. With Chris's help, we discovered that the issue stemmed from a mismatch between BGR and RGB color modes. Correcting the color format resolved the prediction errors. It's a good reminder that sometimes all you need is a fresh pair of eyes on code you've been staring at for hours.

6.4 Milestone Completion Summary

All of the milestones mentioned in the milestone report were completed. The milestone report reflected a re-prioritization of data analysis and storage over fish health detection due to time constraints, in alignment with the rationale provided in the initial project proposal. This shift allowed us to focus on more foundational components such as depth map accuracy and system performance analysis, which are critical for the future scalability of the project.

7 Conclusion

FishSense is an iOS application initially developed for automatic fish length detection. In this project, we extended its functionality to include fish species classification, data collection, and remote data sharing. A key technical advancement was the improvement of the length detection algorithm, accomplished with the help of an external contributor, which significantly enhanced detection accuracy.

Beyond the app, we evaluated the viability of the previously used CCFRP dataset and determined that it was not sufficient for our needs. As a result, we created a new dataset tailored to our updated models and objectives. We also investigated the performance of the depth map and confidence map, both of which are critical for reliable fish measurement and result verification. Additionally, we developed a robust local-to-remote database system, empowering everyday users, to participate in data collection and contribute to marine science. These contributions establish a strong foundation for future development and research.

Potential directions include using the confidence map to prompt real-time photo recapture and estimate the reliability of predictions, supporting Android and non-LiDAR devices via stereo vision, and incorporating video-based input to increase classification and measurement accuracy. Quantizing the classification model for faster inference is also a promising optimization.

Looking ahead, deploying these improved algorithms during this year's CCFRP summer data collection would be a valuable opportunity to evaluate system performance in the field. It would offer critical insights into where we stand and how well the current pipeline supports real-world scientific data gathering—bringing FishSense one step closer to becoming a practical tool for collaborative marine research.

Acknowledgments

We acknowledge the use of ChatGPT for assistance with formatting and developing this report. Special thanks to Christopher Crutchfield for his invaluable support and knowledge throughout the project. We also acknowledge Shaurya Raswan for contributions to the head-tail detection algorithm.

References

[1] Vaneeda Allken, Shale Rosen, Nils Olav Handegard, and Ketil Malde. 2021. A deep learning-based method to identify and count pelagic and mesopelagic fishes from trawl camera images. *ICES Journal of Marine Science* 78, 10 (11 2021), 3780–3792. doi:10.1093/icesjms/fsab227

[2] E. Almansa and Lourdes Reig. 2015. The laser scanner is a reliable method to estimate the biomass of a Senegalese sole (Solea senegalensis) population in a tank. *Aquacultural Engineering* 69 (2015), 78–83. doi:10.1016/j.aquaeng.2015.10.003

[3] Tanya Y. Berger-Wolf, Daniel I. Rubenstein, Charles V. Stewart, Jason A. Holmberg, Jason Parham, Sreejith Menon, Jonathan Crall, Jon Van Oast, Emre Kiciman, and Lucas Joppa. 2017. Wildbook: Crowdsourcing, computer vision, and data science for conservation. arXiv:1710.08880 [cs.CY] https://arxiv.org/abs/1710.08880

[4] Danying Cao, Cheng Guo, Mijuan Shi, Yuhang Liu, Yutong Fang, Hong Yang, Yingyin Cheng, Wanting Zhang, Yaping Wang, Yongming Li, and Xiao-Qin Xia. 2024. A Method for Custom Measurement of Fish Dimensions Using the Improved YOLOv5-Keypoint Framework with Multi-attention Mechanisms. *Water Biology and Security* 3, 4 (2024), 100293. doi:10.1016/j.watbs.2024.100293

[5] Robert David, Jared Duke, Advait Jain, Vijay Janapa Reddi, Nat Jeffries, Jian Li, Nick Kreeger, Ian Nappier, Meghna Natraj, Shlomi Regev, Rocky Rhodes, Tiezhen Wang, and Pete Warden. 2021. TensorFlow Lite Micro: Embedded Machine Learning on TinyML Systems. arXiv:2010.08678 [cs.LG] https://arxiv.org/abs/2010.08678

[6] Fishtechy. 2025. How It Works. https://fishtechy.com/how-it-works/ Accessed: 2025-06-14.

[7] Muhammad Ather Iqbal Hussain, Zhi-Jie Wang, Zain Ali, and Shazia Riaz. 2021. Automatic Fish Species Classification Using Deep Convolutional Neural Networks. *Wireless Personal Communications* 116 (01 2021). doi:10.1007/s11277-019-06634-1

[8] iNaturalist. 2021. iNaturalist. https://www.inaturalist.org. Accessed: 2024-06-13.

[9] D. Jareño, F. Muñoz-Muñoz, and F. Rodrigo. 2024. Automatic labeling of fish species using deep learning across different classification strategies. ResearchGate. https://www.researchgate.net/publication/378795097

[10] Graham G. Monkman, Kieran Hyder, Michel J. Kaiser, and Franck P. Vidal. 2019. Using machine vision to estimate fish length from images using regional convolutional neural networks. *Methods in Ecology and Evolution* 10, 12 (Dec. 2019), 2045–2056. doi:10.1111/2041-210X.13282 Publisher Copyright: © 2019 The Authors. Methods in Ecology and Evolution © 2019 British Ecological Society.

[11] Jessica Stitt, Leona Svancara, Lee Vierling, and Kerri Vierling. 2019. Smartphone LIDAR can measure tree cavity dimensions for wildlife studies. *Wildlife Society Bulletin* 43 (02 2019). doi:10.1002/wsb.949

[12] E. Suharto, Suhartono, A. P. Widodo, and E. A. Sarwoko. 2020. The use of MobileNet V1 for identifying various types of freshwater fish. *Journal of Physics: Conference Series* 1524 (2020), 012105. doi:10.1088/1742-6596/1524/1/012105 The 9th International Seminar on New Paradigm and Innovation of Natural Sciences and its Application, 22 October 2019, Central Java, Indonesia, Open Access.

[13] Shinichi Tatsumi, Keiji Yamaguchi, and Naoyuki Furuya. 2022. ForestScanner: A mobile application for measuring and mapping trees with LiDAR-equipped iPhone and iPad. *Methods in Ecology and Evolution* (2022). doi:10.1111/2041-210X.13900 Open Access, Handling Editor: Carlos Alberto Silva, First published: 08 June 2022.

[14] N. Tonachella, A. Martini, M. Martinoli, D. Pulcini, A. Romano, and F. Capoccioni. 2022. An affordable and easy-to-use tool for automatic fish length and weight estimation in mariculture. *Scientific Reports* 12, 1 (2022), 15642. doi:10.1038/s41598-022-19932-9

[15] J. Tueller, C. Bian, and R. Manduchi. 2021. FishSense: Depth-based fish detection and measurement with underwater RGB-D cameras. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. 3348–3356. https://escholarship.org/uc/item/1n09f1hf

[16] Amaya Álvarez Ellacuría, Miquel Palmer, Ignacio A Catalán, and Jose-Luis Lisani. 2019. Image-based, unsupervised estimation of fish size from commercial landings using deep learning. *ICES Journal of Marine Science* 77, 4 (11 2019), 1330–1339. doi:10.1093/icesjms/fsz216 arXiv:https://academic.oup.com/icesjms/article-pdf/77/4/1330/33513197/fsz216.pdf