SIMULATION-BASED DATA COLLECTION AND DEEP LEARNING FOR DISTRESS SHIP DETECTION USING DRONES FOR MARITIME SEARCH AND RESCUE

Jeonghyo Oh[1], Juhee Lee[1], Youngseo Je[1], Euiik Jeon[1] and Impyeong Lee*[2]
[1]Graduate Student, Department of Geoinformatics, University of Seoul
163 Seoulsiripdae-ro, Dongdaemun-gu, Seoul
Email: {ohrora98, soar21, je_0seo, euiik0323}@uos.ac.kr

[2]Professor, Department of Geoinformatics, University of Seoul,
163 Seoulsiripdae-ro, Dongdaemun-gu, Seoul
Email: iplee@uos.ac.kr

KEY WORDS: Ship detection, Simulation, Drones, Marine search and rescue

ABSTRACT: Drones and deep learning are increasingly being used to quickly detect distress at sea, but the lack of data on distress ships limits detection. In this study, we develop a simulator for marine ship accidents to construct a dataset and train an object detection model to detect distress ships. When developing the simulator, we focused on sinking and capsizing ships. The drone was then used to construct a dataset of ships in distress. We selected the YOLOv8 object detection model for its accuracy and real-time performance, and we trained it using the constructed dataset, evaluating its performance based on various indicators. It was possible to identify and detect ships and distress ships, and all were detected without any missing ships in the test data. As a result, a high accuracy of mAP 0.969 was achieved. The results of this study show that simulation-based datasets can be useful for distress ship detection. In the future, if various marine environments and ships are implemented in the simulation to obtain learning data, it is expected to help minimize human casualties by quickly detecting distress ships in wide oceans.

1. INTRODUCTION

The number of maritime accidents and casualties is increasing worldwide. The scope of distress accidents is increasing due to the expanding maritime activities, and the problem is becoming more serious as maritime traffic and cargo increase. According to KOSTAT (Statistics Korea), there were 3,156 maritime accidents in 2020, up from 2,307 in 2016, indicating that the number of maritime distress accidents is on the rise, as are the number of casualties.

In the wide ocean, it is necessary to quickly detect ships in distress. Satellites, helicopters, and drone are being used to detect ships in distress. Among them, research is increasing on the use of drones, which can detect small-sized objects, have low operating costs and high spatial resolution (Poudel et al., 2023). Drone can be used for quick rescue operations such as ship sinkings in the ocean, and they can also be widely used to search for missing persons (MinHyung et al., 2016). Additionally, applying deep learning algorithms to drones can improve accuracy and shorten the time it takes to locate victims within the accident radius (Do Trong et al., 2021). To create a drone-based ship detection model, a new deep learning model based on CNN utilizing ResNet and DenseNet was constructed, and an accuracy of 99% was achieved. Accordingly, research on deep learning models that consider the features of drone images is increasing (Woźniak et al., 2022). Various deep learning algorithms and application scenarios for object detection using drone have been studied, and the results show that drone are efficient in detecting marine objects (Li et al., 2022).

However, existing ship detection research is focused on monitoring and detecting drifting ships. Therefore, there are limitations in detecting sinking and capsized ships. In order to detect distress ships using deep learning based on drone, it is necessary to have enough training data on various situations and various types of ships taken from the angle and height of the drone. However, there is a limited amount of training data on sinking and capsized ships. Implementing a real-world distress ship and constructing a variety of training data requires a lot of effort, cost, and risk. These problems can be solved by using simulation. Simulation enables the realization of distress situations in various marine environments with little cost and effort. Therefore, it is possible to construct a large amount of training data efficiently.

In this paper, we propose a data collection method for various ships and various search and rescue situations in simulation, as shown in Figure 1. Using a simulation engine, we implement an environment for ship accidents that may happen in the ocean. A camera is set up considering the features of detecting objects based on drone. Data for shipwrecks was constructed in the simulator using the set camera. Considering the features of marine search and rescue, we selected the YOLO(You Only Look Once) version 8 model for accuracy and speed. After training the ship detection model, it was validated with test data in various environments to evaluate its accuracy.

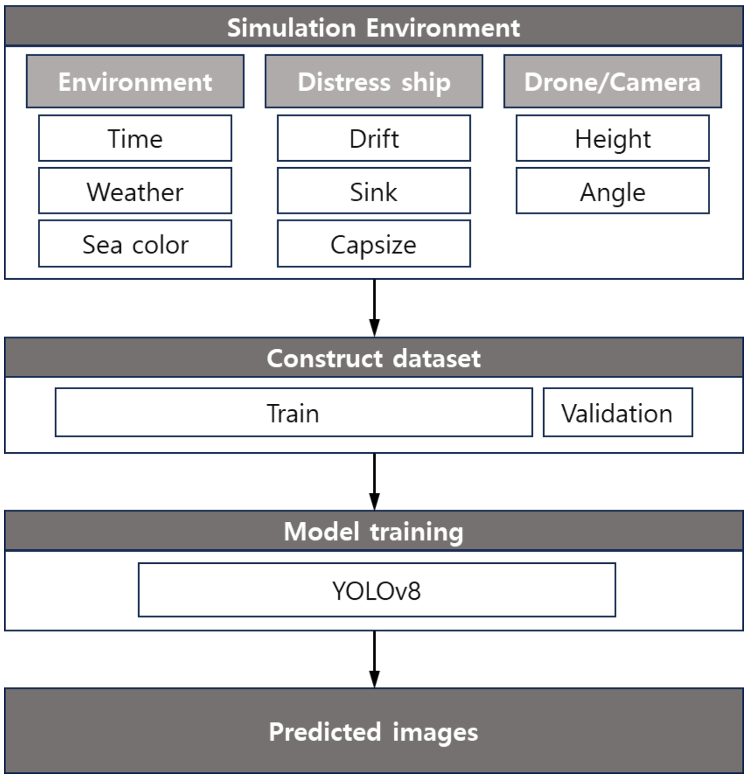


Figure 1. The process from creating a simulation environment to detecting a distress ships.

2. SIMULATION-BASED DATA COLLECTION

Unity was selected as the simulation engine for this study. Unity includes a graphics engine that can provide a realistic virtual environment. Unity is mainly used for game and simulator development, and it can create a realistic simulation environment through its built-in physics engine. Unity has an open marketplace for trading content and resources for simulation development called the Asset Store. The Asset Store is where many 3D objects, audio, and more can be downloaded and used. Unity is a real-time development platform that enables real-time 3D rendering, which is ideal for search and rescue (Noueihed et al., 2022).

Three factors were considered to create a variety of environments for filming distressed ships in the ocean. First, environmental factors such as sea conditions and weather conditions. Second, various types of ships, including sinking and capsizing. Lastly, set the camera to the drone's perspective. Here's how to build a marine environment simulation with Unity.

2.1 Simulation environment

The ocean environment consists of the ocean, time, and weather as shown in Figure 2. The ocean is implemented through resources from the Asset Store, and we used Materia to set the color of the ocean and the texture of the waves. In addition, scripts were written to control the height of the waves to achieve realistic ocean movement. In terms of time of day, the scene is divided into day and night. During the day, the sun can be controlled using a directional light. Directional lights are large, directional lights that come from a distance and are often used to implement the sun. The sunset effect is achieved by controlling the color of the sun.

Different weather factors can be applied to each time of day. Weather factors include various weather conditions that exist in the real world, such as clear, cloudy, clouds, rain, snow, and fog. Weather factors such as rain, snow, and fog can be implemented using Effects. In this study, we fixed the weather conditions of daytime, clear weather and calm waves to focus on the detection of distressed ships. This is a fixed background environment to accurately detect and analyze distressed ships. To implement clear weather during the daytime, skybox materials with clear skies and some clouds were downloaded and applied from the Asset Store. The directional light was controlled to ensure temporal variation in the clear sky, and the sea color was limited to three colors.

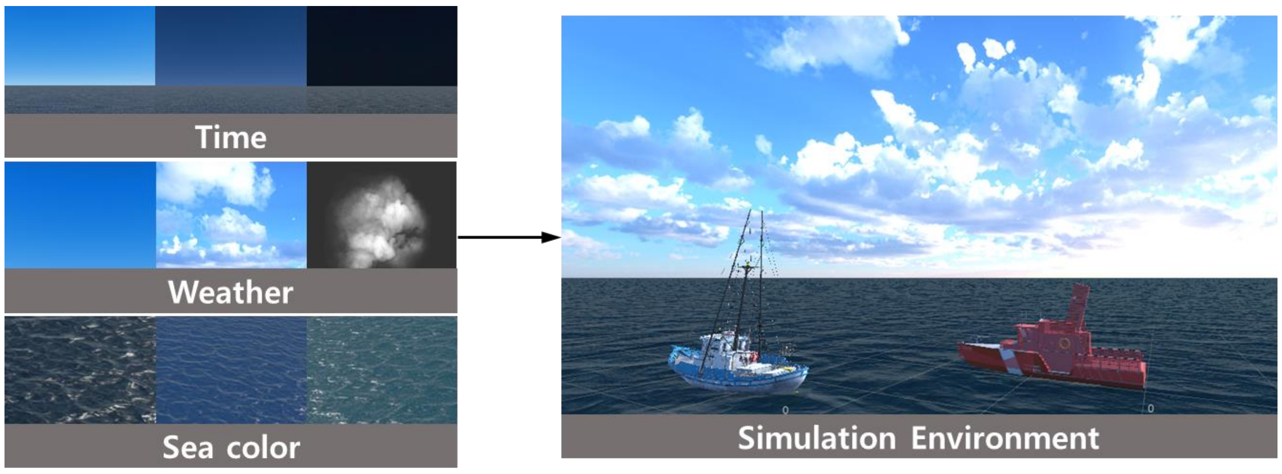


Figure 2. Simulation environment configuration.

2.2 Distress ship

The ships to be floated in the ocean were mainly fishing boats, which have the highest number of accidents, according to KOMSA(Korea Maritime Transportation Safety Authority) statistics. The 3D ships used in this study were downloaded from a website called "cgtrade" and the Unity Asset Store, including two fishing boats and one boat, as shown in Figure 3. In Unity, the scale of an object is a relative value and usually starts at 1.0. The scale was changed to adjust the size of the ship.

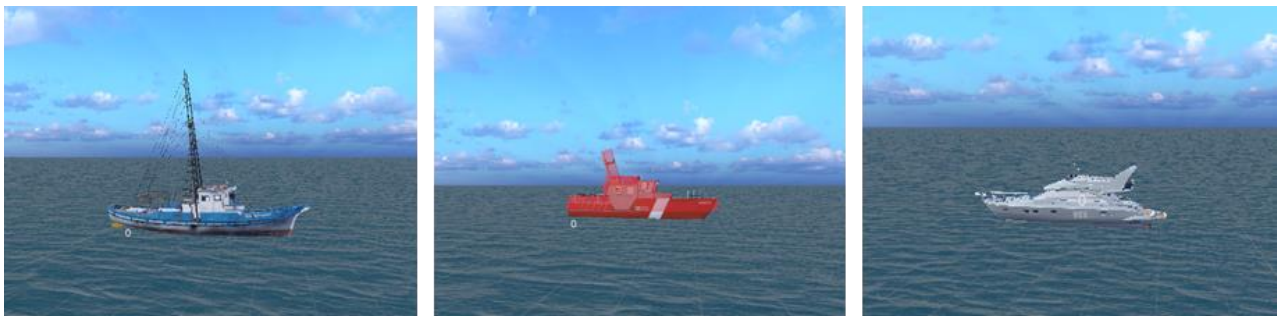


Figure 3. 3D ship model in simulation.

Next, we need to use Unity's physics engine to float a ship on the ocean, as shown in Figure 4, and then sink the floating ship and implement a distress ship. To implement the distress ship, we proceeded as follows. First, insert the ship into the virtual environment. When a ship is inserted into the virtual environment, it is fixed in the air. Next, gravity is applied to the fixed ship. Applying gravity causes the ship to sink to the bottom of the ocean. Next, in order to float the ship on the sea, buoyancy is implemented by applying collision between the sea and the ship. Buoyancy was used to allow the ship to move naturally according to the movement of the waves. In order to realize a sinking ship in various situations, the ship's center of gravity is moved to sink in various ways, such as forward, backward, left, or both. After sinking, a capsize, a state in which the ship was completely turned upside down, was implemented.

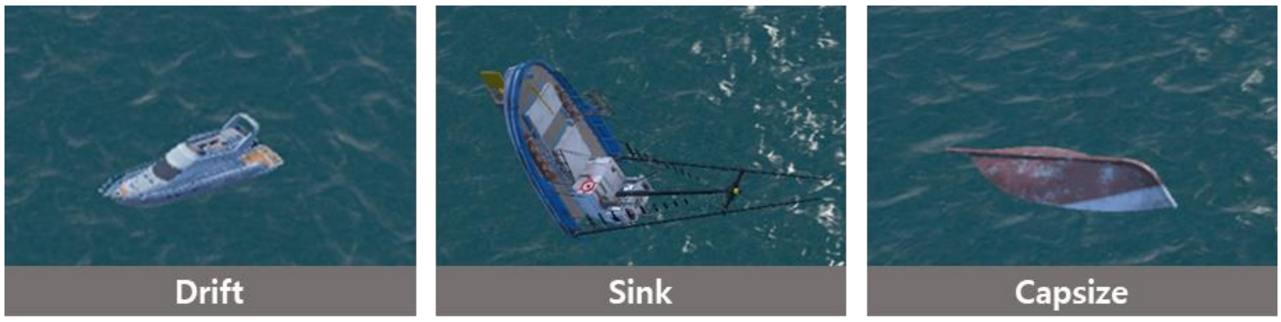


Figure 4. Ship conditions.

2.3 Drone camera

In this study, we implement the drone's camera considering that it is used to search for ships. The camera plays a role in determining what to render. In this study, the main camera is assumed to be the drone's camera since it is used to construct deep learning training data. Therefore, the settings for the height and angle of the main camera were set based on the drone and focused on detecting ships.
Set the camera's height and angle, FOV(Field of View), and render target. Considering that the camera is a drone, we set the height and angle to capture the ship stably and well, as shown in Figure 5. The height was set to 100m, 150m, and 200m, and the angle was set to -30°, -45°, and -75°.

FOV means the camera's field of view, and it should be set according to the drone's field of view. Rendering displays what is visible from the camera's point of view, so we set it to the drone's point of view to render the distress ship or distress scene.

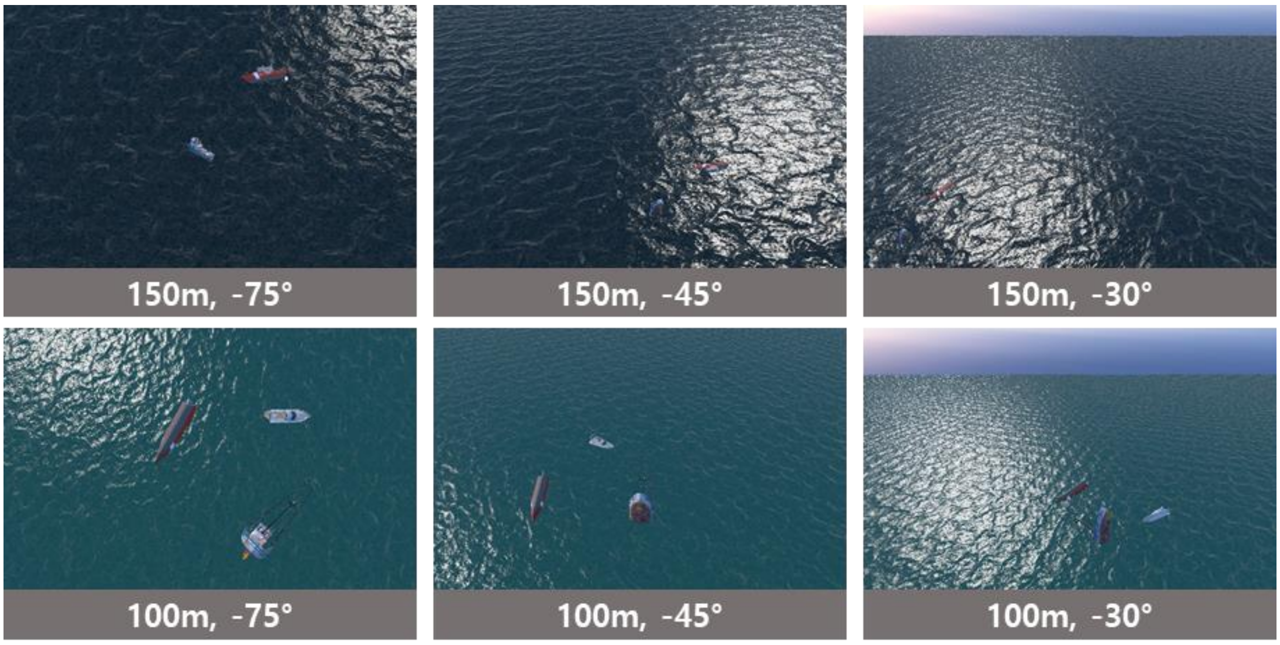
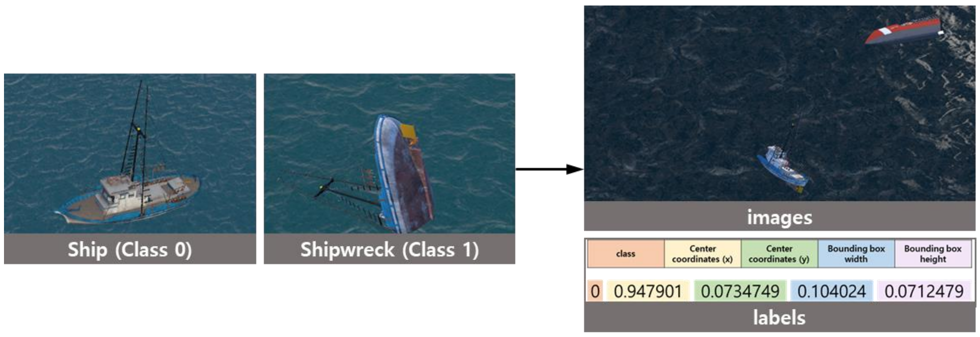


Figure 5. Images taken based on the height and angle of the drone.

2.4 Dataset generation via simulation

The dataset was created by collecting frame-by-frame data from a camera implemented by Unity. From the collected data, two classes of ships were defined: ship and shipwreck. The dataset consists of images and labels as shown in Figure 6. Images are virtual images extracted from the simulation, and labels are information about the ships in the image recorded in bounding boxes. A bounding box is a rectangular area that surrounds a ship and indicates the ship's location in coordinates. The bounding box information for a ship is saved as a text file and is recorded in the order of the ship's class, center coordinates (x, y), width, and height. It was extracted in the format of label data to train the object detection model YOLO.



class	Center coordinates (x)	Center coordinates (y)	Bounding box width	Bounding box height
0	0.947901	0.0734749	0.104024	0.0712479

Figure 6. Composition of the dataset.

3. DEEP LEARNING FOR SHIP DETECTION

We constructed a dataset of distress situations using the implemented simulator. The dataset was trained using YOLOv8, one of the object detection models. After the training was completed, the ship detection model was validated using the validation data.

3.1 Selection and Trainning of Object Detection Model

Object detection, a computer vision image recognition technology, was used to detect the distress ship. Object detection is a technology that performs classification and localization at the same time, identifying objects in the image and returning the coordinates of their location. Object detection models include SOLO, DINO and YOLO.

YOLO is ideal for detecting objects in real time and identifying multiple objects, so we selected the YOLO model considering the features of search and rescue. YOLO has been available in various versions, and we used YOLOv8 as one of the latest versions. YOLOv8 is different from the existing object detection model by applying the "Anchor Free Detection" technique. It is a method that directly predicts the center of an object without using multiple anchor boxes to predict the location of the object. This improves the calculation speed, and since only highly accurate Bounding Boxes are required, object detection is performed quickly and efficiently.

We trained YOLOv8, a deep learning object detection model, using the dataset built by the simulator. The dataset used for training is 4,405 photos, and the training data and validation data are divided into an 8:2 division. Both the training and validation datasets consist of virtual data created in the simulator. The environmental conditions were evenly distributed for data validation to minimize bias. The hyperparameters used for training were optimizer SGD, learning rate 0.01, momentum 0.937, weight decay 0.0005, epoch 100, and batch size 16. We checked the box loss, class loss, and differential loss while training. The Figure 7 shows that the loss decreases as the epoch increases. To validate the performance of the trained model, we used a simulated dataset. The performance was evaluated based on the performance metrics of precision, recall, and mAP50 (mean Average Precision).

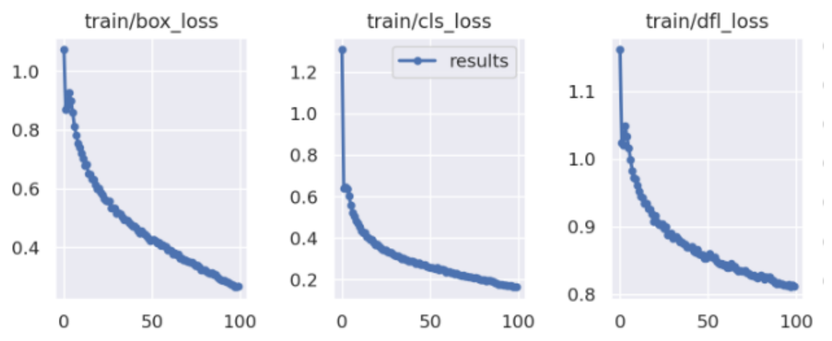


Figure 7. loss for epoch 100 on YOLOv8 model.

3.2 Performance Evaluation

Using the dataset generated through simulation, the trained models were applied to test data and the performance of the dataset was verified based on performance indicators.

Figure 8 shows the performance of the object detection model by changing the height and angle of the camera and applying various environmental factors such as illumination and sea color. As a result of the test with various environmental factors, object detection was performed effectively. In addition, ships and shipwreck classes were identified and detected, and there was not a single ship that was not detected in the entire test data.

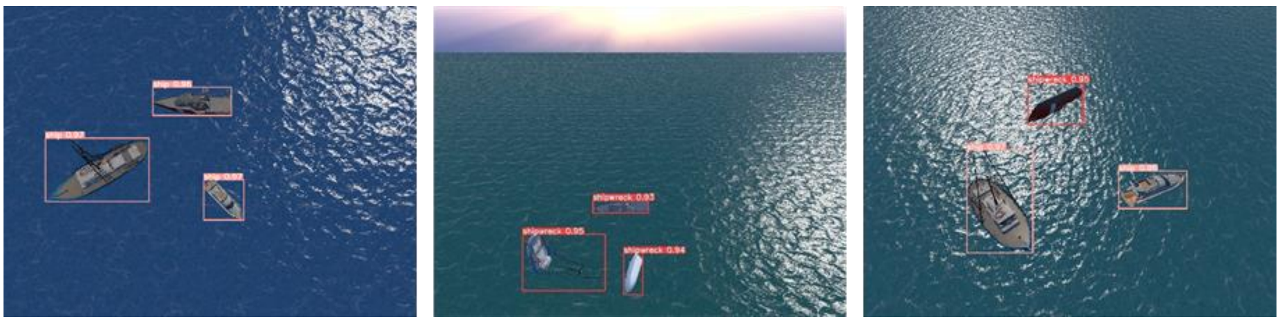


Figure 8. Detection result using YOLOv8.

However, there were also cases of false detections. As shown in Figure 9 (A), shipwreck is falsely detected as ship. This is considered a class imbalance. In (B), we can see that there are about twice as many shipwrecks as ships. Also, in Figure 10 (A), both classes are detected, and there are cases where three classes are detected for one ship. (B) shows the size of the bounding box output from the validation data. As such, we conclude that overdetection is caused by the limited scale of objects in the dataset that need to be detected.

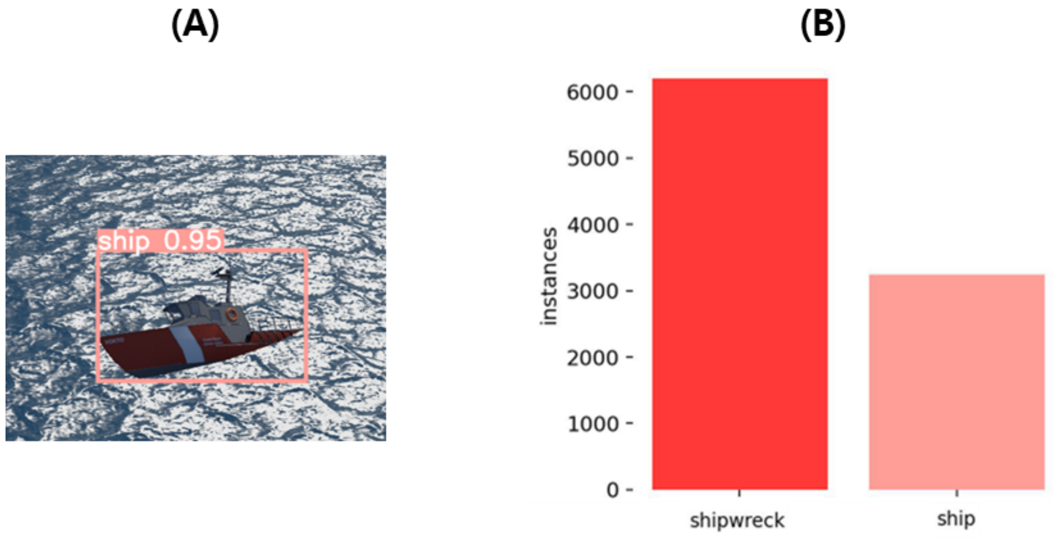


Figure 9. (A) Falsely detected ships, (B) Dataset validation results in an imbalance of classes.

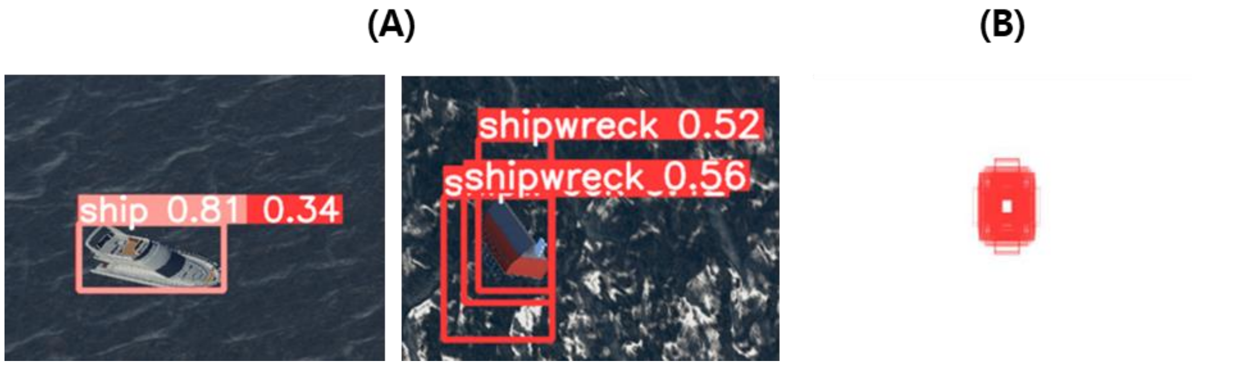


Figure 10. (A) Overdetected ships, (B) Size of the bounding box output from the validation data.

A deep learning model was trained on the dataset to detect ships and distress ships. To evaluate the trained model, the results are based on the performance metrics of precision, recall, and mAP50 for each class as shown in Table 1. For the classes of ship and shipwreck, mAPs of 0.992 and 0.947 were obtained. As a result, the mAP for all classes is also high at 0.969.

Table 1. Evaluate the constructed dataset based on YOLOv8.

Class	precision	recall	mAP50
all	0.906	0.931	0.969
shipwreck	0.933	0.866	0.947
ship	0.879	0.995	0.992

High accuracy was achieved by training the object detection model using virtual data. The high accuracy is likely due to the fact that both the training and validation data were built in the simulator. In order to apply it to the real world, it is necessary to train and verify deep learning models using data constructed by real drone.

4. CONCLUSIONS

In this study, a simulation environment was implemented focusing on the detection of sinking and capsizing of ships using drone. From the implemented simulation, a dataset was constructed considering the features of drone. We trained YOLOv8, an object detection model, using the constructed dataset, and validated the accuracy through the constructed data in the simulation. As a result, ships and distress ships were detected in various environments. In addition, all ships were detected with no missing ships when checked manually. In addition, a high mAP of 0.969 was obtained in terms of performance indicators. However, there were cases where the shipwreck was incorrectly detected as a ship, and there were cases where a single ship was detected multiple times. In the next study, we plan to consider ways to reduce the number of false and duplicate detections through data from various ships and various environments. Through this study, we expect to improve the rescue rate by quickly detecting sunken or capsized ships using drones during marine rescue.

ACKNOWLEDGEMENTS

This research was supported by Korea Institute of Marine Science & Technology Promotion(KIMST) funded by the Ministry of Oceans and Fisheries(KIMST-20220463).

REFERENCES

Do Trong, T., Hai, Q. T., Duc, N. T., & Thanh, H. T. 2021. A novelty approach to emulate field data captured by unmanned aerial vehicles for training deep learning algorithms used for search-and-rescue activities at sea. In: *2020 IEEE Eighth International Conference on Communications and Electronics (ICCE)*, pp. 288-293.

Li, Y., Yuan, H., Wang, Y., & Xiao, C. 2022. GGT-YOLO: a novel object detection algorithm for drone-based maritime cruising. *Drones*, 6(11), pp.335.

MinHyung, L., & KangMoon, L., 2016. Research on coast guard safety activities utilizing unmanned aerial vehicles. *Journal of the Korean Coast Guard Academy*, 6(3), pp.125-142.

Noueihed, H., Harb, H., & Tekli, J. 2022. Knowledge-based virtual outdoor weather event simulator using unity 3D. *The Journal of Supercomputing*, 78(8), pp.10620-10655.

Poudel, R., Lima, L., & Andrade, F. 2023. A Novel Framework to Evaluate and Train Object Detection Models for Real-Time Victims Search and Rescue at Sea with Autonomous Unmanned Aerial Systems Using High-Fidelity Dynamic Marine Simulation Environment. In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision,* pp. 239-247.

Woźniak, M., Wieczorek, M., & Siłka, J. 2022. Deep neural network with transfer learning in remote object detection from drone. In: *Proceedings of the 5th international ACM mobicom workshop on drone assisted wireless communications for 5G and beyond*, pp. 121-126.