IJCCS (Indonesian Journal of Computing and Cybernetics Systems)
Vol.19, No.1, January 2025, pp. 105~116
ISSN (print): 1978-1520, ISSN (online): 2460-7258
DOI: https://doi.org/10.22146/ijccs.101698

Obstacles Detection in Underwater Environment Using ROV Based on Convolutional Neural Network

Purwidi Asri*[1], Yuning Widiarti[2], Endang Pudji Purwanti[3], Endah Wismawati[4], M. Firman Tsany Arifin[5]
[1,2,5]Department of Marine Electrical Engineering, Politeknik Perkapalan Negeri Surabaya;
[3]Department of Shipbuilding Engineering, Politeknik Perkapalan Negeri Surabaya;
[4]Department of Marine Mechanical Engineering, Politeknik Perkapalan Negeri Surabaya
Surabaya, Indonesia
e-mail: ***[1]purwidi@ppns.ac.id**, [2]yuning.widiarti@ppns.ac.id, [3]endangpudjip@ppns.ac.id, [4]endahw@ppns.ac.id**,** [5]firmantsaan@gmail.com

Abstrak

Deteksi hambatan di lingkungan bawah laut merupakan tantangan besar bagi navigasi Remotely Operated Vehicles (ROV) karena kondisi yang kompleks dan keterbatasan visibilitas. Metode konvensional sering kali gagal mengenali hambatan bawah laut dengan akurat, sehingga diperlukan teknologi canggih untuk meningkatkan kemampuan deteksi. Penelitian ini bertujuan untuk mengembangkan sistem deteksi hambatan yang efisien dan akurat di lingkungan bawah laut dengan menggunakan ROV yang terintegrasi dengan Convolutional Neural Network (CNN). Tujuan utamanya adalah untuk meningkatkan kemampuan ROV dalam mendeteksi dan menghindari hambatan, sehingga mendukung misi-misi bawah laut seperti inspeksi kapal, eksplorasi laut, dan pemantauan lingkungan. Sistem yang diusulkan mengintegrasikan kamera bawah air dengan ROV, menggunakan pemrosesan gambar berbasis CNN untuk mendeteksi hambatan. Model CNN dilatih dengan dataset gambar bawah air yang beragam untuk memastikan keandalan di berbagai lingkungan. CNN mengekstrak fitur-fitur penting dari gambar melalui beberapa lapisan konvolusi, di mana setiap lapisan konvolusi menggunakan filter untuk mendeteksi pola seperti tepi, sudut, atau tekstur dari gambar masukan. Pada tahap akhir, fitur-fitur yang telah diproses diteruskan melalui lapisan fully-connected yang berfungsi sebagai pengklasifikasi. Hasil pengujian menunjukkan kinerja sistem yang kuat, dengan akurasi validasi mencapai 99,25% dan akurasi klasifikasi real-time sebesar 85% dalam mengidentifikasi berbagai hambatan bawah laut, seperti botol, tiang kayu, rantai, dan baling-baling, melampaui pendekatan tradisional dalam hal presisi dan respons waktu nyata. Kesimpulannya, penggunaan CNN untuk deteksi hambatan secara signifikan meningkatkan keandalan operasional ROV, menjadikannya solusi yang menjanjikan untuk aplikasi kelautan seperti inspeksi dan eksplorasi bawah laut, di mana akurasi dan efisiensi sangat penting.

Kata kunci*—CNN, halangan, deteksi, klasifikasi, ROV*

Abstract

The detection of obstacles in underwater environments poses a significant challenge for the navigation of Remotely Operated Vehicles (ROVs) due to complex surroundings and limited visibility. Conventional methods often fail to recognize underwater obstacles accurately, necessitating advanced technologies to improve detection capabilities. This study aims to develop an efficient and accurate obstacle detection system for underwater environments using an ROV integrated with a Convolutional Neural Network (CNN). The primary objective is to

Received November 20th,2024; Revised January 8th, 2025; Accepted January 28th, 2025

enhance the ROV's ability to detect and avoid obstacles, supporting underwater missions such as ship inspection, marine exploration, and environmental monitoring. The proposed system integrates an underwater camera with the ROV, using CNN-based image processing to detect obstacles. The CNN model was trained on a diverse dataset of underwater images to ensure reliability in different environments. CNN extracts key features from images through multiple convolutional layers. Each convolutional layer uses filters to detect patterns such as edges, corners, or textures from the input images. In the final stage, the processed features are passed through fully connected layers that act as classifiers. The result demonstrated strong system performance, achieving a validation accuracy of 99.25% and a real-time classification accuracy of 85% in identifying various underwater obstacles, bottles, wooden poles, chains, and propellers, surpassing traditional approaches in precision and real-time response capabilities. In conclusion, using CNN for obstacle detection significantly enhances the ROV's operational reliability, making it a promising solution for marine applications such as underwater inspections and exploration, where accuracy and efficiency are crucial.

Keywords—*CNN, obstacle, detection, classification, ROV, underwater*

1. INTRODUCTION

The advancement of research, development, and utilization of marine resources will continue to grow steadily. The emergence of remotely operated vehicles (ROVs) or unmanned underwater vehicles (UUVs) brings benefits and allows discoveries to be made undersea. The ROV is usually used to assist work in underwater environments where safety considerations are a major factor. Apart from that, the ROV can also reach difficult and remote underwater areas. Therefore, improving the ROV autonomy capabilities, including perceived autonomy, planning autonomy, and capacity for independent behavior, is an important trend for developing ROV or unmanned vehicles. The air, sediment, and underwater environment have different characteristics. In addition, the attenuation, reflection, and signal propagation parameters vary [1,2], demanding hardware settings must be able to adapt to the environment in which the obstacle detection system is expected to operate. Unmanned vehicles cannot learn new behaviors or acquire new knowledge simply by repeating appropriate actions programmed in design [3,4]. However, as artificial intelligence develops, the ability to learn independently from unmanned vehicles is no longer a fantasy.

Obstacles in the water pose a significant challenge for an ROV when operating, as undetected obstacles can damage its body and components. To prevent collisions, the ROV must detect these underwater obstructions. Image processing is highly effective in underwater exploration, as light scattering in aquatic environments can interfere with the camera's ability to capture clear images. The advanced image processing techniques are crucial for identifying objects and ensuring the ROV's safe navigation.

For enhancing underwater images, traditional image processing techniques involve color correction algorithms, contrast enhancement algorithms, and the white balance method [5]. The grey world hypothesis [6] and gray edge hypothesis [7] are common methods used to enhance image quality, while histogram equalization [8] and limited contrast histogram equalization [9] are frequently employed to increase contrast. However, these techniques often yield unsatisfactory results when applied to underwater images due to the complexity of the marine environment. Factors such as light scattering and absorption by water, as well as the presence of suspended particles, significantly interfere with image quality in underwater settings.

The detection of objects underwater primarily involves the use of digital cameras. Image processing techniques enhance image quality and reduce noise, often using contour segmentation to identify objects. Various methods have been proposed for target detection. For example, Chen Chang and colleagues [10] introduced a new image-denoising filter based on the

standard median filter to detect and replace initial pixel values with a newer median. Additionally, Prabhakar and colleagues [11] proposed a denoising method that utilizes homomorphic filters for correcting uneven lighting and anisotropic filters for noise removal in underwater images. The techniques mentioned utilize wavelet decomposition, statistical approaches, laser technology, or color polarization theories. While these methods demonstrate reasonability and effectiveness, a shared drawback is their extensive processing time, making real-time detection challenging.

Convolutional Neural Network (CNN) [12,13,14] has significantly advanced in recent years, emerging as a standout element in the rapidly evolving field of deep neural networks. Computer vision technology enables artificial intelligence to perceive and comprehend visual information. The enhancement of computer hardware capabilities and the development of extensive image annotation datasets have propelled deep learning-based computer vision algorithms to excel in traditional tasks such as image classification, object detection, and image segmentation. In 2014, Girshick and colleagues introduced R-CNN [15], marking the first application of convolutional neural network (CNN) to object detection. This innovation boosted detection accuracy by nearly 30% over traditional algorithms, generating significant interest. Both academic research and practical applications show that CNN-based object detection algorithms offer superior accuracy [16,17] and offer significant advantages in object recognition compared to traditional methods such as Support Vector Machines (SVM) or K-nearest neighbors (K-NN). It performs automatic feature extraction, eliminating the need for manual feature engineering, and allowing the model to learn patterns from simple to complex features [18] Moreover, CNNs leverage spatial hierarchies through convolutional and pooling layers, making them more robust to translation, and rotation, and image scaling variations.[19] The convolutional method also enables parameter sharing, significantly reducing the number of parameters compared to the fully connected neural network, resulting in more computational efficiency.[20] Numerous studies have demonstrated that CNN consistently outperforms traditional methods in object classification, especially on large datasets such as ImageNet, achieving far higher accuracy than non-deep learning approaches. [21]

This research applies the CNN method to ROV for underwater object detection. For the decision-making process, the object detection method is enhanced with underwater object classification, with direct testing conducted in a pool at various distances. Additionally, this study designs an ROV with excellent maneuverability and equipped with a balance sensor that can maintain a specific position despite wave movements. Several stages of work were conducted during this research. The first step is determining the objects the camera detects and identifies on the ROV. Once the objects are defined, images are captured by the camera and categorized into dataset classes: plastic bottle, chain, pole, propeller, and no obstacle, corresponding to their respective objects. The next stage involves processing the images through convolution to create new inputs, followed by max-pooling to speed up computation and reduce noise, and then training the neural network to generate weights for classifying the output. When an obstacle is detected, the output data is sent through Arduino's serial communication to control the motor's movement and avoid the obstacle.

2. METHODS

In this research, we apply the CNN method to ROV for underwater object detection. For the decision-making process, the object detection method is enhanced with underwater object classification, with direct testing conducted in a pool at various distances. Additionally, this study designs an ROV with excellent maneuverability and equipped with a balance sensor that can maintain a specific position despite wave movements. Several stages of work were conducted during this research. The first step is determining the objects the camera detects and identifies on the ROV. Once the objects are defined, images are captured by the camera and categorized into dataset classes: plastic bottle, chain, pole, propeller, and no obstacle,

corresponding to their respective objects. The next stage is processing, where the images undergo convolution to create new inputs. The max-pooling process accelerates computation and reduces noise in the captured image inputs. The subsequent step is training the neural network to generate weights used for classifying the output. When an obstacle output is detected, the output data is parsed through Arduino's serial communication to direct the motor's movement to avoid the obstacle.

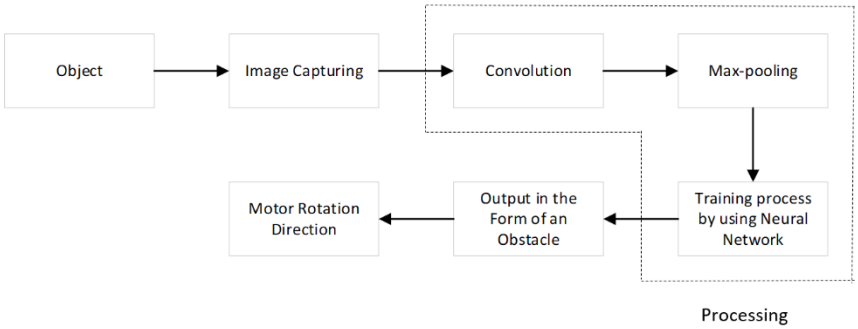


Figure 1 The Stages of Research Work

The next stage is the training process with a Neural Network to produce weights that will be used for output classification. If an obstacle is detected, the output data will be parsed through the Arduino serial interface to determine the motor's movement direction to avoid the obstacle. The work process can be shown in Figure 1.

2.1 Hardware Setup

In the initial stage of the ROV manufacturing process, a system requirements analysis is conducted to determine the necessary equipment and components for achieving the expected system performance. The elements required for designing an ROV in this research include a low-light HD USB camera, BLDC motor, ESC motor driver, gyro sensor (GY-955), LiPo battery, and PC. The flowchart in Figure 2 illustrates the ROV's steps to identify objects. Initially, the ROV dives underwater by activating the underwater camera. It then moves until the camera detects an object or obstacle in front of it. Once an object is detected, the ROV operator reduces the ROV speed to facilitate object or obstacle identification. Upon identification, the name of the object matching the dataset's similarity will be displayed, and the output data will be transmitted via Arduino serial to control the motor movement direction to avoid obstacles.
In the hardware design, the laptop is directly connected to a camera, where the images captured by the camera correspond to predefined classes. These images are used to create a dataset for the training process in the CNN to detect obstacles. The CNN process is implemented using Anaconda software and the Python programming language. The CNN-based detection system is also designed to ensure that the captured images are easily identifiable by the system, considering the water turbidity, which affects image quality.

Another very crucial process is the mechanical design process. Mechanical design is carried out based on a design process using CAD software. One component that plays an important role in an ROV is the propeller. During installation, these propellers are installed at each corner of the ROV using the omni-four wheels robot concept. Apart from the propeller, there are camera components whose quality must be considered. The ROV utilizes a Sony Exmor IMX322 sensor for capturing high-quality underwater images, which is essential for object classification using CNN. The sensor's excellent low-light performance and high

resolution ensure clear, detailed images in challenging underwater environments. This enables the CNN to effectively process and classify objects with greater accuracy.

Another crucial stage in the development of an ROV is the electrical design. This involves several essential components, including six BLDC underwater thruster motors, six Flycolor ESCs, a LiPo battery, a GY-955 sensor, and an Arduino Uno microcontroller. All these electrical components will be wired and arranged in an organized manner within a compartment, ensuring they are neat and can be securely housed inside the ROV's component tube. As a voltage source for the motorbike, a Lithium Polymer battery has the following specifications: Voltage used: 11.1 VDC – 12.6 VDC and current: 2200 mAh. Meanwhile, the voltage source used for the camera is USB power which has the following specifications: Voltage used: 5V/1A. Meanwhile, the propeller driving motor uses a Brushless Motor. BLDC motors are preferred for ROV thrusters due to their efficiency, precise control, and long lifespan. They offer compact power, smooth operation, and high torque at low speeds, making them ideal for maneuvering in underwater environments. Their durability and minimal maintenance needs further enhance their practicality for extended missions in harsh conditions. Each ROV component including the underwater camera and BLDC motors for the six thrusters has been tested separately before being integrated into the ROV, with test results showing that all components functioned well.

The electrical design can be illustrated in Figure 3. Table 1 and Table 2 show the specifications of the ROV and camera used to take pictures respectively.

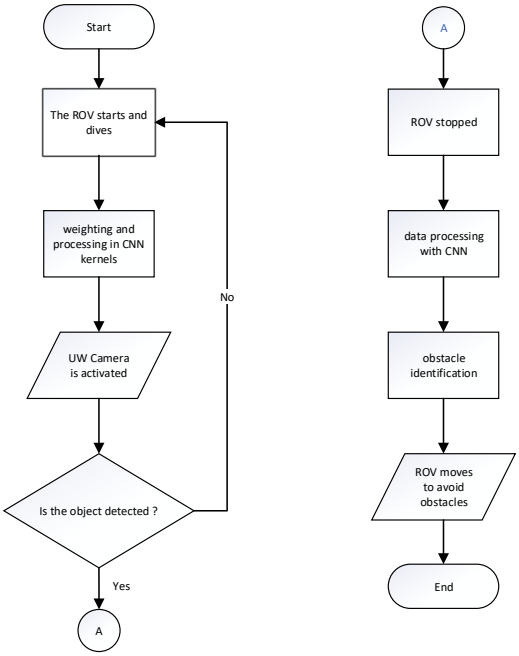


Figure 2 The Flowchart of Detection System

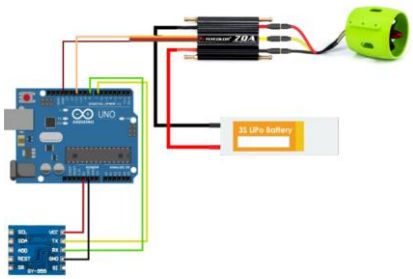


Figure 3 The ROV Electrical Design

Table 1 ROV Spesification

Parameter	Value
Dimension	450 mm x 340 mm x 250 mm
Weight in the air (with ballast and battery)	11-12 kg
Weight in the air (without ballast and battery)	9-10 kg
Buoyancy foam	R-3318 urethane foam rated to 244 m
Construction	HDPE frame, aluminum flanges/end cap, & acrylic or aluminum tubes
Thruster configuration	6 thrusters: 4 vectored and 2 vertical

Table 2 Camera Spesification

Parameter	Value
Type	Sony Exmor IMX322 / IMX323
Camera PCB Dimensions	32mm x 32mm
Mass (total)	17g
Mass (without cable)	13.5g
Working Temperature	-20 - 75°C
Total pixels*	2000(H) x 1121(V) ~2.24 MP
Recording pixels	1920(H) x 1080(V) ~2.07 MP
Minimum Illumination	0.01 lux
Sensitivity	5.0V/lux-sec@550nm
Compression format	H.264 / MJPEG / YUV2（YUYV）
Channels	single-channel, 44.1 kHz
Format	Linear-PCM (L16)
Supply Voltage	5 V
USB Version	2.0
Distortion	1%

2.2 CNN Architecture

After going through the retrieval and adjustment stages with the dataset, the classification of the dataset taken will be processed further using the CNN method. In this research, the CNN architecture has output in the form of 5 categories which are classified as follows: 1. No obstacles; 2. Chain; 3. Plastic Bottles; 4. Wooden Poles; 5. Propeller. The architecture of the CNN method can be seen in Figure 4, which has a pre-processing process, namely dataset formation. Then the next process is processing which consists of image convolution, maxpooling, NN training, and softmax. The last process is classification which has 1 process, namely determining output. This architecture is used as the basis for creating CNN programs, the following is an overview of the CNN architecture. In this architecture, there is an input image that has a resolution of 300 x 300. This input image with the same pixel size that has been determined will enter the first convolution process.

The initial convolution operation utilizes a 5 x 5 x 32 kernel or filter, meaning the filter has a spatial size of 5 x 5 pixels and applies 32 different filters. After processing via the Python program, this operation produces feature maps as the new input image. These feature maps have a reduced resolution compared to the original image but exhibit increased depth, resulting in a resolution of 296 x 296 x 32. The output from this first convolution is then passed to the subsequent stage, the max-pooling process. Max-pooling selects the largest value within specific pixel regions—in this case, 2 x 2 pixels. For each 2 x 2 block of pixels in the feature maps, the maximum value is chosen, reducing the image resolution to 148 x 148 x 32. Upon completion of the max-pooling process, the resulting image proceeds to the second convolution, as this research employs two convolution layers to accelerate the computation process. The resulting image size after going through the second convolution process using a kernel or filter measuring 3 x 3 x 64 has a resolution of 146 x 146 x 64, then a second max-pooling process is

carried out after getting the image from the first convolution process, in the second max-pooling stage This pixel used is 2 x 2, the result is that the resolution of the previous image will change to 73 x 73 x 64.

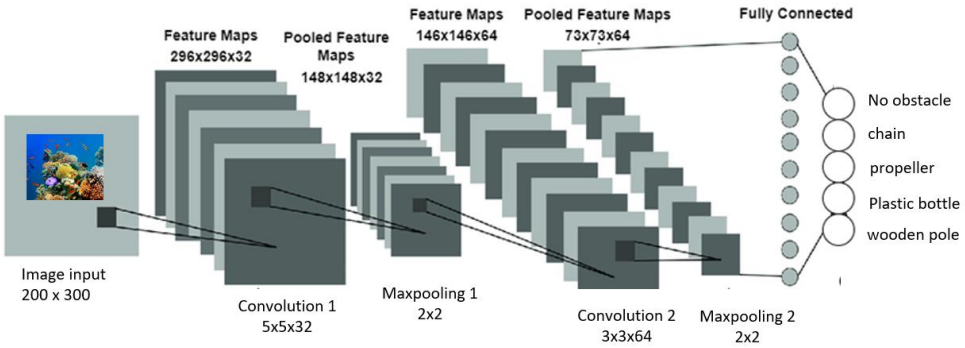


Figure 4 CNN Architecture

After that, we enter the next stage, namely the flattening stage. This stage is a process where the input image that has been obtained will be converted into 1 dimension for training with a neural network (NN). The result of the flattening process is 341,056 x 1. This value will be the input vector for the neural network process. Before entering the training stage, an Optimizer analysis process is needed. This analysis functions to determine the optimizer that will be used which will later be tested with several optimizers. This analysis process takes quite a long time because it requires trial and error. An iterative training process is carried out to determine the appropriate optimizer to obtain the best accuracy with the architecture that has been created. In this study, a comparison was made of the use of optimizers, including Adadelta, Adam, and Adamax.

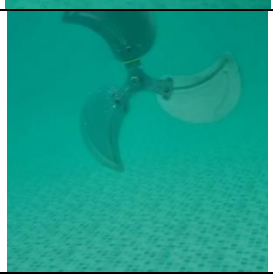
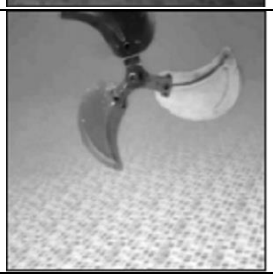
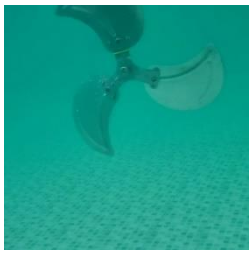
3. RESULTS AND DISCUSSION

3.1 Convolution and Max-Pooling Process

The test starts by determining the optimal distance for obstacle detection, with the results dependent on the resolution of the camera used. A 2.24-megapixel diving camera was used, and the obstacle image data was taken during the day in a swimming pool with sufficient lighting. The distance variations are set at four levels: 2 meters, 1.5 meters, 1 meter, and 0.5 meters. The obstacle distance test is conducted by placing the obstacle at predetermined distances using a distance meter to ensure accurate measurements between the obstacle and the ROV. At a distance of 1.5 meters, the obstacle data appears optimal for training, as all object classes are visible. Similarly, at 1 meter, the images remain optimal for training as the obstacle details are clear, and the proximity is not too close. However, at a distance of 0.5 meters, the images are unsuitable for training as the obstacles are too close. Therefore, data collection at a distance of 1.5 m is considered the optimal distance in this test.

This research's initial stage involves image processing, specifically the convolution and max-pooling processes. The dataset used consists of 50 to 60 images per category, making a total of 270 images. A 5x5 filter with 32 kernels was applied in the first convolution layer. Then, a 2x2 max pooling process followed, and a 3x3 filter with 64 kernels was applied in the second convolution layer. This was followed by another 2x2 max pooling process. The training was conducted using a Neural Network with one hidden layer and 256 hidden neurons. The image that has been processed can be seen in Table 3.

Table 3 Dataset Processing

No	Original Image	Convolution I	Max-pooling I
1			
2			
No	Original Image	Convolution II	Max-pooling II
1			
2			

3.2 Data Training Process

The training process used the Spyder application version 3.0.0 with the Keras library. Keras is a Python library commonly used for building machine learning or artificial intelligence models, with a particular focus on neural network prototyping, ease of use, modularity, and extensibility in Python. During the training process, the `Dense` layer provided by Keras was used for variable declaration. The number of hidden layers in this process was set to 256. The activation function used in this training was the ReLU function, chosen to introduce non-linearity into the model. Additionally, the flattening function was applied to convert the output of the convolutional process, which is a matrix, into a vector format.

The training was conducted over 40 epochs with a total duration of 205 minutes. The optimizer used was Adadelta, achieving an accuracy of 99.25%. The training accuracy with the Adadelta optimizer showed a fluctuating increase up to the 15th epoch, followed by a sharp decline around the 25th epoch, where accuracy dropped to approximately 50%-60%. After the 25th epoch, the accuracy gradually improved, reaching 99%. Adadelta is ideal for tasks with dynamic or fluctuating gradients, offering memory efficiency, automatic learning rate

adaptation, and stable training without the need for manual tuning, making it more suitable than Adam or Adamax in this study.

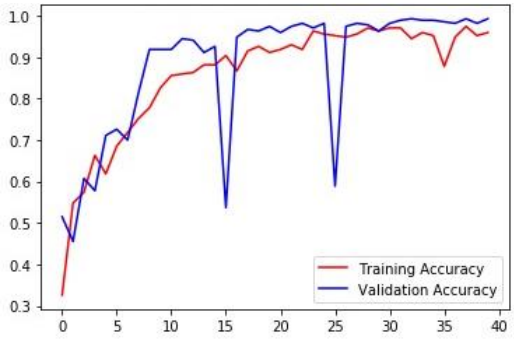


Figure 5 Adadelta Optimizer Accuracy Training Chart

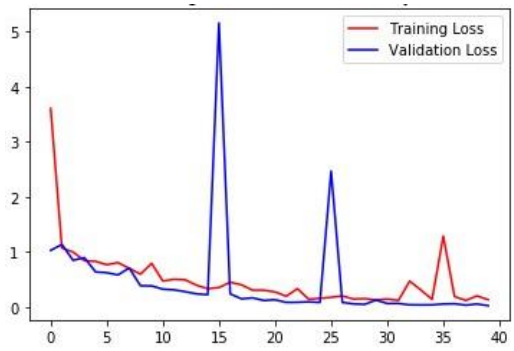


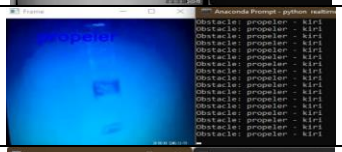
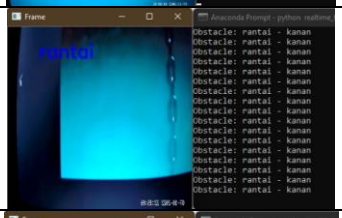
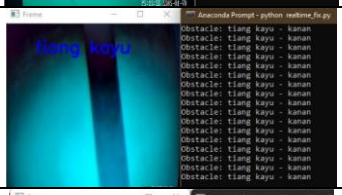
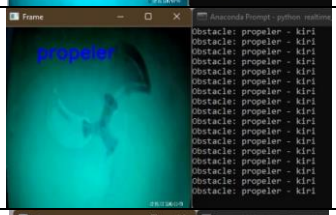
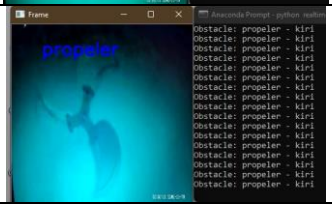
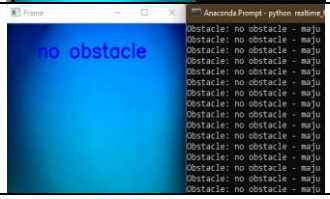
Figure 6 Adadelta Training Loss Optimizer Chart

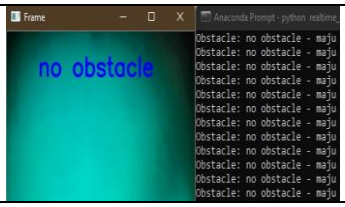
As shown in Figure 5, the accuracy continued to increase steadily, indicating that the Adadelta optimizer is well-suited for the CNN architecture used in this research. As shown in Figure 6, the training loss with the Adadelta optimizer follows a similar trend to the accuracy graph, particularly at epochs 15 and 25. A noticeable difference between the validation loss and training loss is observed at these epochs, with a significant divergence occurring at epochs 15 and 25. The results obtained from the training process using the Adadelta optimizer are as follows: training accuracy of 0.9593, training loss of 0.1283, validation accuracy of 0.9925, and validation loss of 0.0196.

3.3 Real-time Testing

Real-time testing is carried out to determine the true accuracy of the training results carried out. This test is carried out by directing the camera at predetermined obstacles. The weights from the training results that have been carried out are used for real-time testing as shown in Table 4. From the test results in a real environment, the system correctly predicted 51 out of 60 tests, achieving an accuracy of 85%. Several detection errors occurred when testing in underwater conditions, where the system misclassified images due to similarities between certain classes, particularly in the presence of noise in underwater data. This issue was most noticeable between the propeller and bottle classes, as the images from these two classes were nearly indistinguishable. These results suggest that the CNN can make reasonably accurate predictions, even when the test conditions differ from those during dataset collection. However, it is not entirely suitable for controlling motor outputs, as the CNN can change its predictions dynamically, causing serial data sent from Python to Arduino to fluctuate, which in turn affects motor movement. This issue can be mitigated by expanding the training dataset to reduce the occurrence of erratic detection changes and improve CNN detection accuracy.

Table 4 Real-time Test Result

No	Target	*Output*	Test Result
1	Bottle	Bottle	
2	Bottle	Propeller	
3	Chain	Chain	
4	Chain	Chain	
5	Wooden pole	Wooden pole	
6	Wooden pole	Wooden pole	
7	Propeller	Propeller	
8	Propeller	Propeller	
9	*No Obstacle*	*No Obstacle*	

10	*No Obstacle*	*No Obstacle*		

4. CONCLUSIONS

The CNN method with five class categories (bottle, propeller, chain, wooden pole, and no obstacle) demonstrated strong performance, achieving a validation accuracy of 99.25% and a real-time classification accuracy of 85%. Using 5x5 and 3x3 convolutional filters combined with successive max-pooling effectively produced a robust model. Adadelta was selected as the best optimizer based on training results, yielding an accuracy of 0.9593 and low loss values. However, detection errors occurred due to environmental similarities, such as water turbidity, and the limited dataset size for each class. Moreover, although CNN performed well under varying test conditions compared to the dataset collection environment, its detection instability could affect data transmission to the motor system. This instability can be mitigated by increasing the dataset size, which would enhance detection accuracy and reduce unpredictable changes in output. For further implementation, it is recommended to use a more diverse dataset with images from various types of water conditions, such as water with different levels of turbidity, varying lighting, and more complex dissolved particles. In addition, testing in more diverse environments such as the deep sea or coastal areas with many visual disturbances is needed to improve the generalization of the CNN model.

REFERENCES

[1] Y. Widiarti, W. Wirawan, and S. Suwadi, "Joint time-reversal precoding and spatial diversity technique for acoustic communication in shallow water environment," *International Journal of Intelligent Engineering and Systems*, vol. 13, no.1, pp.237-247, 2020, doi: 10.22266/ijies2020.0229.22

[2] Y. Widiarti, E. Setiawan, H.A. Prasetiyo, B. Budianto, I. Sutrisno, A. Adianto, and M.B. Rahmat, "Corrosion Detection on Ship Hull Using ROV Based on Convolutional Neural Network," *International Journal of Marine Engineering Innovation and Research*, vol. *9*, no.1, pp. 218-229, 2024. [Online]. Available: https://iptek.its.ac.id/index.php/ijmeir/article/view/17235 [Accessed: 1-Nov-2024]

[3] R. Kot, "Review of Collision Avoidance and Path Planning Algorithms Used in Autonomous Underwater Vehicles," *Electronics*, vol.11, no. 15, 2022 [Online]. Available: https://doi.org/10.3390/electronics11152301 [Accessed: 3-Nov-2024]

[4] M.A. Kamel, X. Yu, Y. Zhang, "Formation control and coordination of multiple unmanned ground vehicles in normal and faulty situations: A review," *Annu. Rev.* Control, vol. 49, pp.128–144, 2020 [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S1367578820300031 [Accessed: 3-Nov-2024]

[5] E. Y. Lam, "Combining gray world and retinex theory for automatic white balance in digital photography," Proceedings of the Ninth International Symposium on Consumer Electronics, 2005, doi**:** 10.1109/ISCE.2005.1502356

[6] G. Buchsbaum, "A spatial processor model for object colour perception," *Journal of the Franklin Institute*, vol. 310, no. 1, pp. 1–26, 1980 [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/0016003280900587 [Accessed: 4-Nov-2024]

[7] J. Van De Weijer, T. Gevers, and A. Gijsenij, "Edge-based color constancy," *IEEE Transactions on Image Processing*, vol. 16, no. 9, pp. 2207–2214, 2010 [Online]. Available: https://staff.science.uva.nl/th.gevers/pub/GeversTIP07.pdf [Accessed: 4-Nov-2024]

[8] R. Hummel, "Image enhancement by histogram transformation," *Computer Graphics and Image Processing*, vol. 6, no. 2, pp. 184–195, 1977 [Online]. Available:https://www.sciencedirect.com/science/article/abs/pii/S0146664X77800117 [Accessed: 29-Okt-2024]

[9] A. R. Awwalin, E. Setiawati, and C. Anam, "Implementasi Metode Contrast Limited Adaptive Histogram Equalization Dan Laplacian Of Gaussian Filter Untuk Peningkatan Kontras Citra CT," BERKALA FISIKA, vol. 24, no. 1, pp. 35-43, Jul. 2021. [Online]. Available: https://ejournal.undip.ac.id/index.php/berkala_fisika/article/view/39825 [Accessed: 25-Okt-2024]

[10] C.C. Chang, J.Y. Hsiao, and C.P. Hsieh, "An Adaptive Median Filter for Image Denoising," Second International Symposium on Intelligent Information Technology Application, Shanghai, China, 2008, pp. 346-350, doi: 10.1109/IITA.2008.259.

[11] C.J. Prabhakar, P.U.P. Kumar, "Underwater image denoising using adaptive wavelet subband thresholding," Proceedings of the 2010 International Conference on Signal and Image Processing, Chennai, India, 15–17 December 2010; pp. 322–327, doi: 10.1109/ICSIP.2010.5697491

[12] Z. Li, F. Liu, W. Yang, S. Peng, and J. Zhou, "A Survey of Convolutional Neural Networks: Analysis, Applications, and Prospects," in IEEE Transactions on Neural Networks and Learning Systems, vol. 33, no. 12, pp. 6999-7019, Dec. 2022, doi: 10.1109/TNNLS.2021.3084827.

[13] Q. Xie, Y. Wang, J. Ding, and J. Niu, "Light Convolutional Neural Network for Digital Predistortion of Radio Frequency Power Amplifiers," in IEEE Communications Letters, vol. 28, no. 10, pp. 2377-2381, Oct. 2024, doi: 10.1109/LCOMM.2024.3443104.

[14] Z. Gao, Z. Lu, J. Wang, S. Ying, and J. Shi, "A Convolutional Neural Network and Graph Convolutional Network Based Framework for Classification of Breast Histopathological Images," in IEEE Journal of Biomedical and Health Informatics, vol. 26, no. 7, pp. 3163-3173, July 2022, doi: 10.1109/JBHI.2022.3153671.

[15] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation," 2014 IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 2014, pp. 580-587, doi: 10.1109/CVPR.2014.81.

[16] Z. Wang, K. Xu, S. Wu, L. Liu, L. Liu, and D. Wang, "Sparse-YOLO: Hardware/Software Co-Design of an FPGA Accelerator for YOLOv2," in IEEE Access, vol. 8, pp. 116569-116585, 2020, doi: 10.1109/ACCESS.2020.3004198.

[17] Q. Xie, D. Zhou, R. Tang, and H. Feng, "A Deep CNN-Based Detection Method for Multi-Scale Fine-Grained Objects in Remote Sensing Images," in IEEE Access, vol. 12, pp. 15622-15630, 2024, doi: 10.1109/ACCESS.2024.3356716.

[18] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," in Proceedings of the IEEE, vol. 86, no. 11, pp. 2278-2324, Nov. 1998, doi: 10.1109/5.726791.

[19] A. Krizhevsky, I. Sutskever, I., and G.E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Communications of the ACM*, vol. 60, no.6, pp. 84-90.

[20] I. Goodfellow, Y. Bengio, and A. Courville, "*Deep learning*," MIT Press, 2016.

[21] K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 2016, pp. 770-778, doi: 10.1109/CVPR.2016.90.