2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

Image-Level Domain Alignment for Real-time Underwater Crack Detection Using YOLO with an ROV

Pachelle Carelle Negue Kala[1,2], Christophe Viel[3], Lucia Bergantin[1,*]

***Abstract*— Underwater concrete infrastructure plays a crucial role in energy and water systems. However, it requires regular inspections to ensure structural integrity. Remotely Operated Vehicles (ROVs) offer a safer and more cost-effective alternative to diver-based inspections. The data collected during inspections often require extensive post-mission processing, either manually or through computationally intensive algorithms. This limitation makes real-time damage detection during inspections impossible. In this study, we present a real-time image-level domain alignment pipeline suitable for deployment on resource-constrained hardware. It combines image enhancement with crack detection using a YOLO11n-seg model fine-tuned on a publicly available aerial concrete crack dataset. The model was quantized and deployed on a Jetson Nano, which was connected to an ROV for real-time inference. To reduce the domain gap between the raw underwater images captured by the ROV and the aerial training data, a Contrast Limited Adaptive Histogram Equalization (CLAHE)-based strategy was applied. Field tests were conducted on a submerged concrete embankment in a turbid lake environment. A validation dataset was developed to evaluate performance offline and is publicly available.**

I. Introduction

Underwater concrete infrastructure (such as dams, offshore wind turbine foundations, and subsea pipelines) plays a crucial role in energy and water systems. Due to their importance, these structures require regular inspection to guarantee structural safety. However, such inspections are challenging due to limited access, harsh environments and the need to use non-destructive evaluation methods. In certain areas, diving is either not allowed for safety reasons or too risky to attempt, and sensor installation may be too costly. Thus, underwater robots, particularly Remotely Operated Vehicles (ROVs), are increasingly being used as a safer and more cost-effective alternative to diver-based inspections [1].

ROVs, a subset of Unmanned Underwater Vehicles (UUVs), are tethered systems remotely controlled from the surface. Equipped with cameras and various sensors, they are used for a wide range of tasks such as underwater inspection, data acquisition, and maintenance in deep or dangerous environments. Operating ROVs near submerged structures remains challenging due to currents, poor visibility, and other environmental constraints. Research efforts increasingly aim to enhance the intuitiveness, reliability, and safety of ROV control systems for remote operation [2], [3].

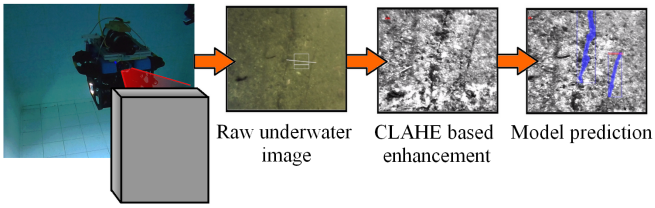

Fig. 1. During deployment, raw underwater images captured by the ROV's monocular camera are first enhanced using the CLAHE-based method, then passed to the deep learning model for real-time inference onboard the Jetson Nano.

The most common ROV-based approach for underwater concrete infrastructure inspection currently relies on human operators remotely controlling the vehicle to collect data. The data are subsequently processed after collection is complete, either through manual review or by computationally intensive algorithms. As a result, the process is either susceptible to human error or resource-intensive. In either case, it is time-consuming and may not be suitable for emergency situations where human operators need to act quickly. Detecting damage in real-time would alert the operator to conduct a more detailed inspection of a specific area, which is more difficult post-mission. These observations highlight the need for automated, real-time methods capable of detecting concrete damage online.

Inspection strategies for non-submerged concrete infrastructure using robots have been widely investigated. Recent examples include the use of autonomous flying robots [4], [5], [6] and climbing robots [7] (see [8] for a comprehensive review). Machine learning (ML) and Deep learning (DL)-based methods have also been widely explored for concrete infrastructure inspection [9]. The majority of these strategies focus on post-mission data processing (e.g., [10], [11]). A real-time DL-based approach using YOLOv3 has been tested onboard an autonomous flying robot for online damage detection [12]. A recent study has also explored quantization techniques for model deployment on flying robots with limited onboard computational resources [13].

The specific challenges of underwater environments (such as poor visibility, limited lighting, and debris) make the direct application of methods developed for non-submerged infrastructure to underwater concrete infrastructure ineffective [14]. Previous studies have proposed DL techniques for underwater inspection, specifically for crack identification [15], [16], [17], [18]. However, very few studies have tested

[1] IMT Atlantique, Lab-STICC, RAMBO team, UMR CNRS 6285, 29238, Brest, France
[2] Nantes Université, École Centrale Nantes, CNRS, LS2N, ARMEN team, UMR 6004, Nantes, France
[3] CNRS, Lab-STICC, ROBEX team, ENSTA IP Paris, F-29806, Brest, France
[*] Correspondence: lucia.bergantin at imt-atlantique.fr

these strategies in real-time onboard underwater robots under a wide range of visibility and water turbidity conditions [19].

One of the main challenges in developing DL-based methods for concrete infrastructure inspection is the lack of dedicated datasets. For non-submerged structures, a substantial body of work focuses on optical crack detection datasets (e.g., [20], [21], [22]). Publicly available datasets for other types of damage, such as spalling or crumbling [23], or for other types of data remain limited. In the case of DL-based strategies for underwater inspection, most studies rely on task-specific optical datasets previously collected, which are typically not made publicly available. Custom datasets are then used to fine-tune pre-trained models (e.g.,[17], [24]). However, data collection and annotation are time-consuming processes. Additionally, depending on visibility conditions (water color, turbidity, differences in brightness, etc.), optical underwater data annotation can be less reliable than aerial data. This shows the need for methods that can effectively leverage readily available datasets collected in-air without depending on additional large-scale data collection and annotation.

Image enhancement for underwater images is a widely studied field of research (see [25] for a comprehensive review). Image enhancement techniques can be divided into three categories: contrast enhancement, color correction, and hybrid methods. Previous research has shown the effectiveness of contrast enhancement techniques in increasing the contrast of underwater crack images [26], [27]. Image enhancement has also been exploited for fine-tuning based on publicly available optical aerial crack datasets. For example, previous studies have relied on multi-layer adversarial learning [28] or on transfer learning from optical to sonar images [17].

In this study, we present a real-time image-level domain alignment strategy suitable for deployment on resource-constrained hardware. Our approach avoids fine-tuning on task-specific custom datasets and enables online crack detection, with the latter potentially reducing the operator's cognitive load by highlighting areas for further review. This strategy is based on an image enhancement and crack detection pipeline (see Figure 1). We fine-tuned a pre-trained YOLO11n-seg model on a publicly available aerial concrete crack dataset [22]. The resulting model was then quantized and deployed on a Jetson Nano connected to an ROV for real-time inference. To visually reduce the domain gap between the raw underwater images captured by the ROV and the aerial dataset used to fine-tune the DL crack detection model, our method employs a simple, non-learning-based enhancement technique based on Contrast Limited Adaptive Histogram Equalization (CLAHE) [29]. During field deployment on a submerged concrete embankment in a lake environment with high turbidity, image enhancement was performed on a surface-level computer connected via cable to the ROV, while the Jetson Nano handled model inference. During tests, the ROV was manually controlled by a human operator, while the DL model performed automatic detection of cracks on the concrete surface. A validation dataset of underwater images, captured by the ROV's monocular camera during prior tests in the same turbid lake environment used for deployment, was developed for offline evaluation and is publicly available[1]. The pipeline performance was evaluated using standard detection metrics: precision, recall, and Intersection over Union (IoU).

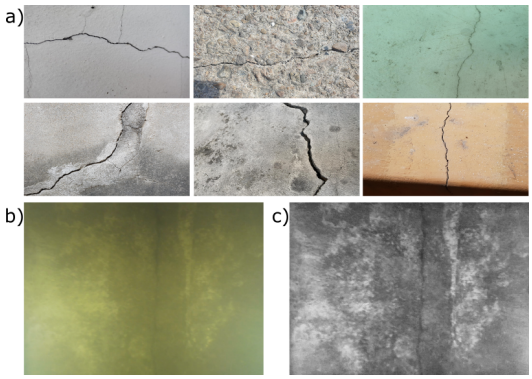

Fig. 2. a) Examples of images taken from the dataset used for fine-tuning [22]. b) Example of frame used to tune the CLAHE-based enhancement strategy. Only 4 frames were used for tuning. c) Frame shown in b) after processing.

The main contributions of this work are:

- A real-time image enhancement and crack detection pipeline based on CLAHE for underwater domain alignment on resource-constrained robotic platforms, with no training on annotated underwater datasets,
- Deployment of this pipeline on a Jetson Nano for onboard, real-time crack detection on submerged concrete surfaces,
- Development of a validation dataset consisting of underwater images captured by a monocular camera mounted on the ROV and used only for evaluation purposes,
- Field validation of the presented strategy on a concrete embankment in a turbid lake environment.

The outline of the paper is as follows: Section II gives details on the DL model and fine-tuning choices, Section III presents the CLAHE-based image enhancement strategy, Section IV gives details on the collection and annotation of the validation dataset, Section V describes the material and experimental setup, Section VI shows the results obtained, and finally Section VII presents conclusions and discusses perspectives.

II. Fine-tuning with a pre-trained YOLO

A. Pre-trained model choice

To perform crack detection, we selected the YOLO11n-seg pre-trained model developped by Ultralytics[2] based on a combination of accuracy, efficiency, and compatibility with our deployment and dataset constraints. Using a YOLO model allowed us to leverage a network pre-trained on the large COCO dataset, enabling us to benefit from its robustness without the need for extensive additional training.

[1] https://github.com/RAMBO-Lab-sticc/crack-detection
[2] https://docs.ultralytics.com/tasks/segment/\#models

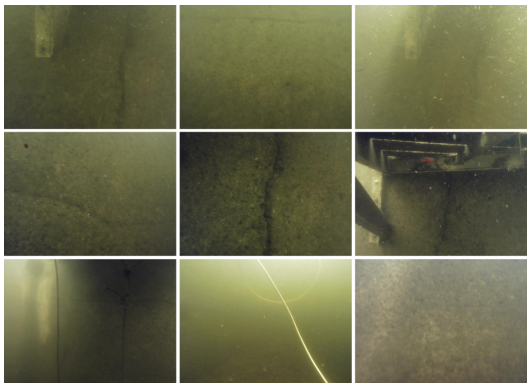

Fig. 3. Examples of images in the validation dataset.

Although YOLOv8 remains widely adopted in the robotics community, early benchmarks indicate that YOLO11 offers improved inference speed and precision, particularly in its lightweight configurations[3]. To accommodate the limited computational resources available on our Jetson Nano, we chose the smallest version, YOLO11n. We selected the instance segmentation version to align with the majority of publicly available aerial crack detection datasets. This choice also enables future extensions of our work toward quantifying crack features (e.g., length, width, and depth) and automatically identifying the most high-risk cases. The comparative performance of the different YOLO11-seg variants for instance segmentation are presented at this link[4].

B. Fine-tuning

Fine-tuning on YOLO11n-seg was performed using pre-trained weights originally trained on the COCO dataset. The dataset used for fine-tuning was taken from Roboflow[5] and presents 1551 images of cracks on non-submerged concrete surfaces, split in 1239 images for training and 312 images for validation [22][6] (see examples in Figure 2.a). Annotations include bounding boxes information and pixel-wise segmentation masks for cracks.

The model was trained using the server described in Section V-A, for 100 epochs with a batch size of 16 and an input resolution of 640x640. We used the SGD optimizer with a momentum of 0.8515 and a weight decay of 0.0005. The initial learning rate was 0.00777, and the final learning rate was 0.00705. Warmup was applied for 2.52 epochs with a warmup momentum of 0.7246. The loss components were weighted as follows: box loss = 7.6411, classification loss = 0.5715, and DFL loss = 1.4230. Data augmentation techniques included hue adjustment, saturation adjustment, brightness adjustment, translations, scale variations, left-right flipping, and mosaic augmentation.

[3]https://docs.ultralytics.com/compare/yolov8-vs-yolo11/

[4]https://docs.ultralytics.com/tasks/segment/\#models

[5]https://roboflow.com/

[6]https://universe.roboflow.com/university-bswxt/crack-bphdr/dataset/1

C. Quantization

After fine-tuning, the model was first exported to ONNX format on the server, then converted to a TensorRT engine using the trtexec tool on the Jetson Nano device. During this process, we applied quantization to reduce the weights from 32-bit to 16-bit floats, reducing the model size.

III. CLAHE-BASED IMAGE ENHANCEMENT

To avoid the need for further training on a custom underwater dataset, we opted for a domain alignment strategy: we employ a CLAHE-based image enhancement technique to reduce the domain gap between the target (underwater) and source (air) visual domains. We chose a CLAHE-based strategy for its lightweight and straightforward implementation, well suited for deployment on our platform. Learning-based approaches were avoided to prevent a two-model pipeline that would increase computational load.

During deployment, images captured by the monocular camera mounted on the ROV were cropped from their original resolution of 1280×720 to 1000×720 and enhanced using CLAHE[7]. To suppress noise and smooth the image, a Gaussian blur was then applied. Finally, the blurred image was combined with the CLAHE-enhanced image using weighted addition to sharpen features. These image enhancement steps were executed on a surface-level computer connected to the ROV via tether. The enhanced images were then transmitted to the Jetson Nano device and processed by the YOLO11n-seg model for inference (see Section V-C for more details).

The CLAHE parameters were tuned using a set of 4 frames (see Figure 2.b and c). These frames were extracted from a previously recorded video captured by the ROV in the same turbid lake environment used for deployment. The parameters adjusted during tuning were the contrast limiting threshold and the grid size (final values: 2 and 8x8 pixels, respectively). While our method avoids retraining the detection model with task-specific data, CLAHE parameters are tuned using a small, unlabeled, and task-agnostic dataset representative of the target environment. This approach eliminates the need for large-scale data collection and annotation, while still enabling effective adaptation to challenging real-world conditions. Unlike learning-based image enhancement methods, our approach is computationally lightweight. Offloading image enhancement to the surface-level computer reduces the computational burden on the Jetson device, preserving its resources for inference.

IV. VALIDATION DATASET

Due to the lack of publicly available underwater benchmarks for crack detection, we created a small custom dataset for offline validation. This dataset consists of 175 underwater images collected during previous field tests in the same turbid lake environment used for final deployment (see Section VI-C). The dataset presents not only examples of cracks on submerged concrete surfaces, but also very few examples

[7]https://www.geeksforgeeks.org/python/clahe-histogram-eqalization-opencv/

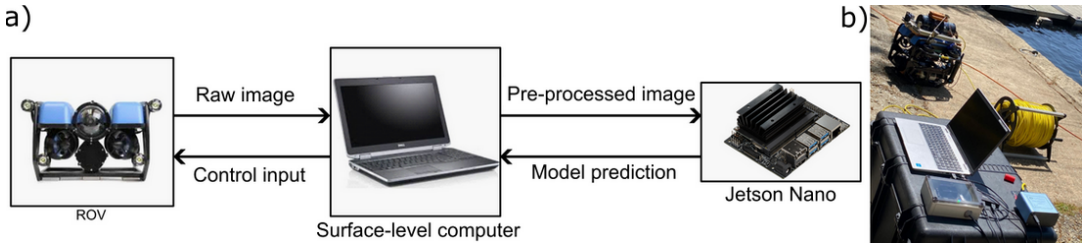

Fig. 4. a) Schema showing the connections among ROV, surface-level computer and Jetson Nano. b) Figure showing the experimental setup.

of ropes and tethers (three and one examples, respectively), which we empirically observed tend to confuse our fine-tuned model. Images were annotated on Roboflow, following the YOLO format, with both bounding boxes information and segmentation masks defined using polygon coordinates. Note that the manual annotations of segmentation masks are often unreliable due to low visibility and challenging image conditions. This limitation makes the use of this validation dataset unreliable for the evaluation of segmentation metrics. Since object detection is less sensitive to pixel-level errors, we can still use this dataset for object detection evaluation.

Although limited in size (and thus unsuitable for model training), this dataset enabled us to perform offline evaluation of the model's object detection performance in our target domain. The dataset is publicly available at this link[1] (see Figure 3 for examples).

V. Material

A. Training server

The server used to fine-tune the DL model was equipped with four NVIDIA RTX A5000 GPUs, each featuring 24 GB of GDDR6 memory. These GPUs delivered up to 20 TFLOPS in FP32 and 40 TFLOPS in FP16 precision, operating at a maximum frequency of 1845 MHz. The system ran on Ubuntu 20.04 and leveraged CUDA 12.9 for GPU-accelerated computations.

B. Experimental material

1) ROV: A BlueROV2[8] was used for tests in the lake. BlueROV2 are equipped by standard with a frontal monocular camera, two lights facing forward, a barometer, and an IMU. The monocular camera used for tests is the Low-Light HD USB Camera[9] (standard cameras on BlueROV2), developed for underwater vision with excellent low-light performance, good color handling, and onboard H.264 video compression. An image of size 1280×720 was chosen for the transmission. The ROV was manually controlled during the experiment, with control system in depth and heading to keep them constant when the operator did not control them.

2) Surface-level computer: The surface-level computer used during the tests was equipped with an 11$^{\text{th}}$ Generation Intel® Core™ i5-11500H processor. It featured a configurable TDP-up base frequency of 2.90 GHz, 6 cores, 12 threads, 12 MB cache, and a maximum turbo frequency of up to 4.60 GHz. The system ran Ubuntu 22.04. This computer was responsible for communicating with both the ROV and the Jetson Nano device, handling robot control commands, enhancing images before sending them to the Jetson for inference, and displaying results on the screen (raw underwater image and model prediction output). A GPU was not used to run image enhancement onboard the computer during deployment.

3) Jetson Nano: In this study, we used the NVIDIA Jetson Nano, part of the Jetson Nano Developer Kit by Waveshare[10]. This platform was chosen for its low power consumption (adjustable between 5 and 10 W) and compact size, with a view to future integration into the robot itself without going through the computer (solution not adopted in this study due to space limitations onboard the current ROV). It features a Maxwell GPU with 128 CUDA® cores clocked up to 921 MHz, a quad-core ARM Cortex-A57 CPU running at 1.43 GHz, and 4 GB of LPDDR4 RAM. Storage is provided via 16 GB eMMC, a microSD card, or an optional USB drive. The Jetson Nano supports high-resolution video processing, including H.264/H.265 encoding up to 4K at 30 FPS and decoding up to 4K at 60 FPS, and interfaces with cameras through 2 MIPI CSI-2 connectors. Additional connectivity includes Gigabit Ethernet, HDMI and DisplayPort outputs, M.2 Key E for WiFi modules, USB 3.0 ports, and various GPIO interfaces. The system operated under Ubuntu 20.04 and served as the deployment platform for the DL model, enabling real-time crack detection.

C. Experimental setup

A ROS2 architecture was used on the surface-level computer to run our code. The ROV communicated with the computer via the ROV's tether. Control input messages were sent using the MAVROS2 package, which communicated with the Pixhawk on the BlueROV2 via the MAVLink protocol encapsulated in UDP. Raw images from the ROV's camera were sent to the computer using a UDP video stream. The image cropping and CLAHE filtering were performed on the computer. The images were then transferred to the Jetson Nano using the SSH protocol. The results obtained by the Jetson Nano were then communicated to the computer in a similar manner. As the current Jetson Nano device runs with Ubuntu 20.04, installing ROS2 was not feasible. As a result,

[8] https://bluerobotics.com/store/rov/bluerov2/

[9] https://bluerobotics.com/store/sensors-cameras/cameras/cam-usb-low-light-r1/

[10] https://www.waveshare.com/wiki/Jetson_Nano_Developer_Kit

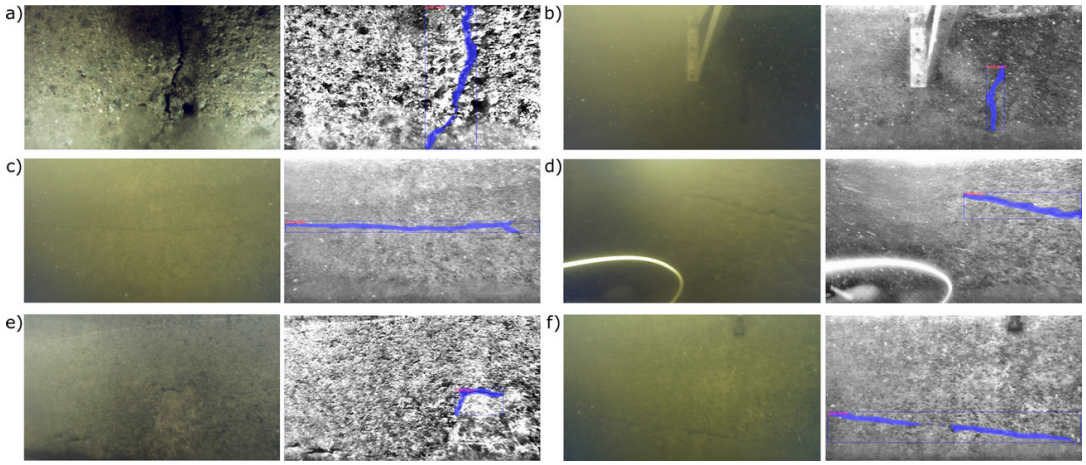

Fig. 5. Examples of results obtained during real-time deployment on a submerged embankment in the Guérledan Lake, in Brittany, France. For each example, on the left the raw underwater image captured by the monocular camera mounted on the ROV and on the right the pipeline output.

communication with the surface-level computer could not be implemented as efficiently as desired. Note that images were not buffered: the Jetson processed only the most recent image received, and all images captured during the inference and transmission computer-Jetson period were discarded. Figure 4 illustrates the connection between the ROV, the surface-level computer and the Jetson Nano device.

VI. Results

A. Validation results with aerial dataset

To validate our model's fine-tuning, we tested it on the validation portion of the aerial dataset used for training. The model demonstrated reliable bounding box predictions, achieving a precision of 0.848, a recall of 0.719, and an Average Precision at IoU 0.5 (AP50) of 0.794. It also achieved strong segmentation performance, with a mask precision of 0.789, a mask recall of 0.661, and a mask AP50 of 0.697. Inference during validation on the server required 3.8 ms per sample.

B. Validation results with underwater dataset

To evaluate our model's performance underwater, we tested it on the custom validation dataset described in Section IV. Such a dataset is representative of the deployment environment (see Section VI-C). However, the manually annotated segmentation masks were often unreliable due to low visibility and challenging image conditions (see Figure 3). Therefore, we decided to focus only on object detection metrics, which provide a more robust and interpretable evaluation of model performance in this setting.

First, validation was performed on the raw underwater images without any image enhancement. Under these conditions, the model performed poorly, achieving a precision of 0.815, recall of 0.326, and a AP50 of 0.342. When the image-level domain alignment strategy was applied, performance improved substantially, with a precision of 0.872, recall of 0.716, and a AP50 of 0.770. This demonstrates that our CLAHE-based image enhancement strategy significantly enhances the model's detection capabilities in underwater images, improving recall and AP50 performance (see Table I). However, the model's inference speed during validation on the server was slightly affected: with no image enhancement it required 4.5ms per image, while when this was applied inference increased slightly to 5.3ms per image.

Dataset	Condition	Precision	Recall	AP50
Aerial	No enhancement	0.848	0.719	0.794
Underwater	No enhancement	0.815	0.326	0.342
	With enhancement	0.872	0.716	0.770

TABLE I

Validation results for object detection on aerial and underwater (with and without image enhancement) datasets.

C. Tests in lake

We tested the presented image-level domain alignment pipeline with an ROV on a submerged concrete embankment in the Guerlédan Lake, Brittany, France (see Section V-C). The lake environment featured high turbidity and challenging lighting conditions. Currents and environmental conditions made manual control of the ROV difficult, resulting in additional motion blur in the captured images.

These tests qualitatively demonstrated that our strategy is effective in detecting cracks on submerged concrete surfaces in real time (see Figure 5 and supplemental videos[11]). The average inference time on the Jetson Nano during deployment was 63ms using the model quantized to 16-bit floats, which fits with the real-time requirements of the application at hand. Communication between the surface-level computer and the Jetson Nano device appears to be the slowest part of our pipeline, explaining the latency visible in the video. Since inference was performed during deployment, we were unable to quantitatively validate the model's performance. However, we observed that the presence of ropes, tethers and linear-looking objects occasionally confused the model,

[11] https://youtu.be/cyEjR-S8iJU

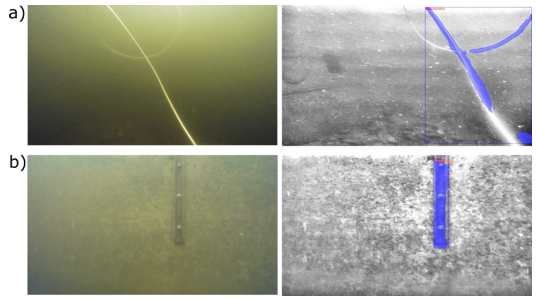

Fig. 6. Examples of false positives during deployment. For each example, on the left the raw underwater image captured by the monocular camera mounted on the ROV and on the right the pipeline output.

resulting in false positives (see Figure 6). Overall, the model successfully highlighted the presence of cracks to the human operator in real-time, allowing them to focus on the indicated areas for further analysis.

VII. Conclusions

In this study, we present a real-time image-level domain alignment strategy suitable for deployment on resource-constrained hardware with an ROV for detecting cracks on underwater concrete surfaces. Our aim is to develop a solution to assist human operators controlling the ROV in identifying cracks during inspections, without the need to wait for resource-intensive and time-consuming post-mission data processing. Our approach does not rely on large-scale data collection and annotation for further training of a DL model, but leverages publicly available aerial datasets thanks to a CLAHE-based image enhancement strategy.

While our method eliminates the need to retrain the crack detection model on annotated and task-specific data, it still requires manual tuning of CLAHE parameters using a small, unlabeled dataset representative of the deployment environment. This limitation could be mitigated by integrating non-learning-based adaptive contrast enhancement techniques adapted for fast processing of underwater images, such as [30]. In practical scenarios such as routine inspection of submerged concrete structures, this requirement is not a significant drawback. Inspections are generally done in known environments for which small amount of previously collected unlabeled data is already available. Nonetheless, adaptive contrast enhancement strategies could still be beneficial in environments with high variability in visual conditions. Such strategies could also enable adaptation of our pipeline to unknown environments (currently unsuitable) without pre-collected data. Here, we avoided learning-based approaches to limit computational demands, though future work may leverage more powerful hardware to explore them.

In this study, the Jetson Nano was not mounted onboard the ROV; instead, connections were made through a surface-level computer. This was done for practical reasons, due to space limitations onboard the ROV. Future implementations of our strategy will include mounting the Jetson Nano device on the ROV in a dedicated sealed enclosure and installing ROS2 directly on the processing unit (which was not possible here due to its current software setup). Doing so, the raw underwater images captured by the ROV could be sent directly to the Jetson Nano, without passing through the surface-level computer. These changes should help reduce latency and improve overall performance. Future work might exploit more powerful hardware to reduce inference time.

We developed a small underwater validation dataset consisting of images captured by the monocular camera mounted on the ROV during previous missions. This dataset was used only for evaluation purposes and is made publicly available[1]. However, the manual annotations of segmentation masks are often unreliable due to low visibility and challenging image conditions. This is why we consider this dataset unreliable for segmentation evaluation and report precision, recall, and AP50 results for object detection only. Future work should focus on developing a larger and more reliable underwater dataset to be used as a benchmark, by leveraging image enhancement techniques combined with synthetic data generation. Inter-annotator agreement and consensus-based labeling strategies should also be explored.

Our strategy enables real-time crack detection, highlighting areas that the human operator remotely controlling the ROV should prioritize for further analysis. While this does not entirely replace post-mission data processing, it may help in reducing the cognitive load of the operator by facilitating the identification of areas that warrant expert review. However, in our current implementation, the presence of ropes, tethers and other similar objects might lead to false positives. This limitation could be addressed by further training the DL model with aerial datasets or synthetic data that include additional relevant classes. This would allow the model to better distinguish between cracks and other visually similar objects. Our current validation dataset is too small for accurate statistical analysis, with only 3 examples of tethers and one of ropes. Future work should add more examples of these elements to the aforementioned benchmark dataset to improve reliability assessment. It will also focus on extending this approach to detect other common types of underwater concrete damage, such as spalling or crumbling. Quantitative characteristics of cracks (such as length, width, and depth) should also be automatically estimated, to provide more detailed real-time feedback to human operators during inspections.

Future work will also include further testing of this strategy in other challenging underwater environments, and the integration of adaptive contrast enhancement techniques to avoid tuning CLAHE parameters beforehand. Inference time during deployment should be reduced by further exploring quantization from 16-bit to 8-bit floating precision. Pruning techniques to reduce the overall model size should be explored. A deployment with field experts to compare performance against human detection (both during and post-mission) and to assess the impact on the operator's cognitive load would be interesting. Further improvements could include automatically locating concrete damage with respect to the initial position of the ROV, and automating the ROV itself to avoid the need for remote operation.

ACKNOWLEDGMENT

We thank Ing. V. Maugliani, Ing. M. Mauro (Ministero delle Infrastrutture e dei Trasporti Direzione Generale per le Dighe e Infrastrutture Idriche, Ufficio Tecnico per le Dighe di Milano) and Ing. G. Bergantin for their help. We also thank Ing. A. Terret (Ufficio Tecnico Consorzio di Bonifica di Piacenza) and Ing. F. Sainati (Edison) for the data provided. This work was supported by ISblue project, Interdisciplinary graduate school for the blue planet (ANR-17-EURE-0015) and co-funded by a grant from the French government under the program "Investissements d'Avenir" embedded in France 2030.

REFERENCES

[1] F D. Ledezma, A. Amer, F. Abdellatif, A. Outa, H. Trigui, S. Patel, and R. Binyahib. A market survey of offshore underwater robotic inspection technologies for the oil and gas industry. In *SPE Kingdom of Saudi Arabia Annual Technical Symposium and Exhibition*, pages SPE–177989. SPE, 2015.

[2] H. Teigland, V. Hassani, and M. T. Møller. Operator focused automation of rov operations. In *2020 IEEE/OES Autonomous Underwater Vehicles Symposium (AUV)*, pages 1–7, 2020.

[3] P. Xia, H. You, and J. Du. Visual-haptic feedback for rov subsea navigation control. *Automation in Construction*, 154:104987, 2023.

[4] Y. Nishimura, S. Takahashi, H. Mochiyama, and T. Yamaguchi. Automated hammering inspection system with multi-copter type mobile robot for concrete structures. *IEEE Robotics and Automation Letters*, 7(4):9993–10000, 2022.

[5] P. Pfändler, K. Bodie, G. Crotta, M. Pantic, R. Siegwart, and U. Angst. Non-destructive corrosion inspection of reinforced concrete structures using an autonomous flying robot. *Automation in Construction*, 158:105241, 2024.

[6] C. N. Tsaimou, K. T. Chelioti, and V. K. Tsoukala. Introducing unmanned aerial vehicles to multi-criteria performance assessment of inspection techniques for port concrete infrastructure. *Maritime Technology and Research*, 6(4):269216–269216, 2024.

[7] L. Yang, B. Li, J. Feng, G. Yang, Y. Chang, B. Jiang, and J. Xiao. Automated wall-climbing robot for concrete construction inspection. *Journal of Field Robotics*, 40(1):110–129, 2023.

[8] A. J. Lee, W. Song, B. Yu, D. Choi, C. Tirtawardhana, and H. Myung. Survey of robotics technologies for civil infrastructure inspection. *Journal of Infrastructure Intelligence and Resilience*, 2(1):100018, 2023.

[9] H. Zhuang, Y. Cheng, M. Zhou, and Z. Yang. Deep learning for surface crack detection in civil engineering: A comprehensive review. *Measurement*, page 116908, 2025.

[10] B. Chen, H. Zhang, G. Wang, J. Huo, Y. Li, and L. Li. Automatic concrete infrastructure crack semantic segmentation using deep learning. *Automation in Construction*, 152:104950, 2023.

[11] T. V. Tran, H. Nguyen-Xuan, and X. Zhuang. Investigation of crack segmentation and fast evaluation of crack propagation, based on deep learning. *Frontiers of Structural and Civil Engineering*, 18(4):516–535, 2024.

[12] P. Kumar, S. Batchu, S. Raju Kota, et al. Real-time concrete damage detection using deep learning for high rise structures. *IEEE Access*, 9:112312–112331, 2021.

[13] X. Yang, E. del Rey Castillo, Y.g Zou, and L. Wotherspoon. Uav-deployed deep learning network for real-time multi-class damage detection using model quantization techniques. *Automation in Construction*, 159:105254, 2024.

[14] D. Chen, B. Huang, and F. Kang. A review of detection technologies for underwater cracks on concrete dam surfaces. *Applied Sciences*, 13(6):3564, 2023.

[15] X. Li, L. Xu, M. Wei, L. Zhang, and C. Zhang. An underwater crack detection method based on improved yolov8. *Ocean Engineering*, 313:119508, 2024.

[16] Z.g Lv, S. Dong, Z. Xia, J. He, and J. Zhang. Enhanced real-time detection transformer (rt-detr) for robotic inspection of underwater bridge pier cracks. *Automation in Construction*, 170:105921, 2025.

[17] L.g Zheng, H. Tan, C. Ma, X.g Ding, and Y. Sun. A real-time crack detection approach for underwater concrete structures using sonar and deep learning. *Ocean Engineering*, 322:120582, 2025.

[18] W. Cao and J. Li. A novel image multitasking enhancement model for underwater crack detection. *Structural Health Monitoring*, 24(4):1969–1990, 2025.

[19] S. Teng, A. Liu, X. Ye, J. Wang, J. Fu, Z. Wu, B. Chen, C. Liu, H. Zhou, Y. Zeng, et al. Review of intelligent detection and health assessment of underwater structures. *Engineering Structures*, 308:117958, 2024.

[20] Ç. F. Özgenel. Concrete crack segmentation dataset, 2019. Version 1.

[21] Y. Shi, L. Cui, Z. Qi, F. Meng, and Z. Chen. Automatic road crack detection using random structured forests. *IEEE Transactions on Intelligent Transportation Systems*, 17(12):3434–3445, 2016.

[22] University. Crack dataset. `https://universe.roboflow.com/university-bswxt/crack-bphdr`, dec 2022. Accessed: 2025-08-29.

[23] L. Yang, B. Li, W. Li, Z. Liu, G. Yang, and J. Xiao. Deep concrete inspection using unmanned aerial vehicle towards cssc database. In *Proceedings of the IEEE/RSJ international conference on intelligent robots and systems*, pages 24–28, 2017.

[24] H. Liu, J.e Yuan, Q. Ren, M. Li, Z. Qi, and X. Deng. Remotely operated vehicle (rov) underwater vision-based micro-crack inspection for concrete dams using a customizable cnn framework. *Automation in Construction*, 173:106102, 2025.

[25] L. S. Saoud, M. Elmezain, A. Sultan, M. Heshmat, L. Seneviratne, and I. Hussain. Seeing through the haze: A comprehensive review of underwater image enhancement techniques. *IEEE Access*, 2024.

[26] G. Xin, X. Fan, P. Shi, C. Luo, J. Ni, and Y. Cao. A fine extraction algorithm for image-based surface cracks in underwater dams. *Measurement Science and Technology*, 34(3):035402, 2022.

[27] H. Zhang, J. Li, F. Kang, and J. Zhang. Monitoring depth and width of cracks in underwater concrete structures using embedded smart aggregates. *Measurement*, 204:112078, 2022.

[28] X.n Fan, P. Cao, P. Shi, X. Chen, X. Zhou, and Q. Gong. An underwater dam crack image segmentation method based on multi-level adversarial transfer learning. *Neurocomputing*, 505:19–29, 2022.

[29] K. Zuiderveld. Contrast limited adaptive histogram equalization. In *Graphics gems IV*, pages 474–485. 1994.

[30] W. Zhang, P. Zhuang, H.-H. Sun, G. Li, S. Kwong, and C. Li. Underwater image enhancement via minimal color loss and locally adaptive contrast enhancement. *IEEE Transactions on Image Processing*, 31:3997–4010, 2022.