applied sciences

Article

Improving CNN Fish Detection and Classification with Tracking

Boubker Zouin [1,2,3], Jihad Zahir [2], Florian Baletaud [3,4], Laurent Vigliola [3] and Sébastien Villon [3,*]

1 Faculty of Sciences Semlalia, Cadi Ayyad University, Marrakesh 40000, Morocco; zouinboubker1@gmail.com
2 LISI Laboratory, Cadi Ayyad University, Marrakesh 40000, Morocco; j.zahir@uca.ac.ma
3 ENTROPIE (Écologie Marine Tropicale des Océans Pacifique et Indien), IRD (Institut de Recherche Pour le Développement), UR (Université de la Réunion), UNC (Université de la Nouvelle-Calédonie), CNRS (Centre National de la Recherche Scientifique), IFREMER (Institut Français de Recherche pour l'exploitation de la mer), Centre IRD de Nouméa, 98000 Noumea, New-Caledonia, France; florianbaletaud@hotmail.com (F.B.); laurent.vigliola@ird.fr (L.V.)
4 Soproner, Groupe GINGER, 98000 Noumea, New Caledonia, France
* Correspondence: villon@cerfacs.fr

Abstract: The regular and consistent monitoring of marine ecosystems and fish communities is becoming more and more crucial due to increasing human pressures. To this end, underwater camera technology has become a major tool to collect an important amount of marine data. As the size of the data collected outgrew the ability to process it, new means of automatic processing have been explored. Convolutional neural networks (CNNs) have been the most popular method for automatic underwater video analysis for the last few years. However, such algorithms are rather image-based and do not exploit the potential of video data. In this paper, we propose a method of coupling video tracking and CNN image analysis to perform a robust and accurate fish classification on deep sea videos and improve automatic classification accuracy. Our method fused CNNs and tracking methods, allowing us to detect 12% more individuals compared to CNN alone.

Keywords: marine ecosystems; convolutional neural networks (CNNs); BRUVS video data; fish classification; automatic processing; tracking

check for updates

Citation: Zouin, B.; Zahir, J.; Baletaud, F.; Vigliola, L.; Villon, S. Improving CNN Fish Detection and Classification with Tracking. *Appl. Sci.* **2024**, *14*, 10122. https://doi.org/10.3390/app142210122

Academic Editors: Zhaoqing Pan, Marcin Iwanowski and Sungho Kim

Received: 25 September 2024
Revised: 23 October 2024
Accepted: 24 October 2024
Published: 5 November 2024

1. Introduction

At a time when anthropogenic activities and global changes are exerting increasing pressure on marine ecosystems [1] the enhancement of underwater wildlife management and conservation has become more critical than ever [2]. The regular and consistent monitoring of these ecosystems is essential for detecting changes over time, understanding the ecosystems' health and functionality [3] and informing effective conservation strategies [4]. Traditional methods of underwater monitoring often prove to be labor-intensive [5], expensive [6], and sometimes lack the desired accuracy or resolution. Therefore, it is necessary to design innovative tools to monitor marine biodiversity frequently and on a large scale. Recent advances in underwater camera technology have facilitated the collection of an enormous amount of marine imagery [7]. However, given that the manual processing of such vast data is impractical due to constraints of time and resources, algorithms based on Machine Learning (ML) and deep learning (DL) have been developed to automate the data processing [8]. These algorithms can identify and classify underwater species [9,10], map habitats [11], and track movements [12] and behaviors [13], thereby creating extensive databases of information that can be analyzed to identify patterns and trends. Despite their promise, these algorithms currently face limitations, such as the need for large training datasets [14], challenges with low light or turbidity, difficulties in identifying rare or camouflaged species, and the need for ongoing validation and refinement. Furthermore, Convolutional neural networks (CNNs) are designed to analyze each image separately while data often comprises video formats with similar information between frames. As such, algorithms are not leveraging the temporal aspect of the video. One way to enhance automatic algorithms for counting and identifying fish could be to exploit this information by linking the detection models, like CNNs, with a tracking algorithm [15].

When dealing with tracking algorithms for multiple objects [16], the process entails monitoring objects within a video. This is achieved by analyzing each frame separately, conducting object detection on each frame, and subsequently attempting to establish correspondences between the objects detected in the current frame and those identified in the preceding frame. This matching process is facilitated by employing a specific measurement technique designed to enhance the accuracy of the object matching. If fish tracking in video has been a topic for a decade [17–19], the coupling of deep learning algorithms and tracking algorithms is a recent and current topic. CMFTNet [20] employs a deformable convolutional network backbone architecture combined with a counterpoised loss function to effectively detect and track individual fish in aquaculture videos. Its performance is evaluated using metrics such as MOTA and IDF1, although these do not include ecologically relevant metrics. Wang et al. [21] combined YOLOv5 with an adaptation of SiamRPN++, originally designed for single target tracking. The paper proposed a use case of this method to analyze the behavior of the porphyry seabream in aquaculture videos. FishMOT [22] proposed to couple a YOLOv7 detection algorithm with IoU evaluation serving as the tracking module. The algorithm is applied to Zebrafish seen from above, using the Trex 2D scenery [23] and the idtracker.ai [24] datasets.

To our knowledge, this is the first paper proposing a coupling between a deep architecture and a tracking algorithm applied in unconstrained baited remote underwater videos.

This process heavily depends on the accuracy of the detector. However, fish detection models are not usually able to be consistent and fail to detect that same object for a couple of frames [25], leading to extra ID assignments and thus a low tracking quality and poor counting quality. In our paper, we propose an end-to-end method consisting of a CNN detector, a tracker, and post-processing. The tracking module (tracker+post-processing) can easily be plugged in to any CNN architecture depending on the goal while being extremely cost-effective. We then compare the efficiency of our method to the state of the art to identify and count fish in underwater videos. Our method fusing CNN and tracking methods allowed us to detect 12% more individuals compared to CNN alone.

Key points:

- Use of temporal data through video tracking;
- Overall increase in coverage by 12%;
- Application to real-life unconstrained baited remote underwater videos used for ecological studies in the south pacific ocean;
- Cost-efficient computing architecture and easy-to-plug modules for any CNN architecture.

2. Material

We used 289 videos from GoPro Hero 4 cameras with a medium field of view in 1920 × 1080 at 60 frames per second and a 1200 lumens, 120-degree angle led light (Groupbinc). Each video is 15 s long, is centered around deepwater snapper fish (*Lutjanidae* family), shows one or multiple individuals moving on screen, and was recorded at depths varying between 47 and 552 m, recorded on deep slopes and seamounts marine habitats of New-Caledonia, South Pacific [26]. This video dataset was split into a training set of 159 (Table 1) videos for our CNN model training and a testing dataset made of 130 videos (Table 2). Such a division ensured that the testing and training datasets were independent and that the shown results are representative of future applications. Each video was then cut into frames at a 1 frame per second rate for the training. Then, all frames were annotated. The coordinates, enclosed by a bounding box, as well as the species name of each fish were recorded. We define a sample as one individual annotation. Finally, to test our tracking dataset, we cut our testing videos at a rate of 30 frames per second.

Table 1. Numbers of species samples used to train the detection model.

Species	Number of Occurences in Videos
Pristipomoides filamentosus	2345
Aphareus rutilans	375
Pristipomoides flavipinnis	268
Aprion virescens	186
Etelis coruscans	145
Pristipomoides argyrogrammicus	134
Parapristipomoides squamimaxillaris	68

Table 2. Numbers of species samples used to test the detection model.

Species	Number of Occurences in Videos
Pristipomoides filamentosus	1303
Pristipomoides flavipinnis	395
Aphareus rutilans	239
Etelis coruscans	117
Pristipomoides argyrogrammicus	114
Aprion virescens	74

3. Method

3.1. General Pipeline

Our testing pipeline was composed of three modules: a convolutional neural network (CNN) detection model, a tracking algorithm, and a post-processing algorithm (Figure 1).

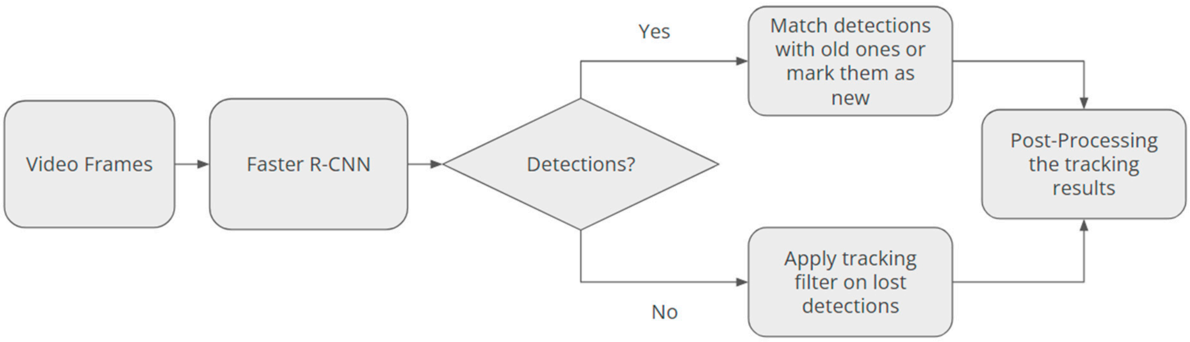

Figure 1. General pipeline of our algorithm.

3.2. CNN Training

In our study, we used a TensorFlow implementation of the Faster Region-Based Convolutional Neural Network (Faster R-CNN) designed for object detection [27], which had been pre-trained on the Common Objects in Context (COCO) dataset [28]. This model was used with a hybrid Inception module combined with a Nas ResNet configuration (Inception-ResNet V2), processing images in the 1024 × 1024 format. The model's architecture is available on TensorFlow [29] 2's model directory on GitHub. We fine-tuned this architecture using Baited Remote Underwater Video Stations (BRUVS) images annotated with deep-water snapper species, serving as our training dataset. The training and testing of the model were conducted using the open-source TensorFlow API in Python 3. Our computational hardware setup comprised four parallelized NVIDIA Quadro RTX 8000 cards, boasting 196 GB of CPU memory and 42 GB of GPU memory. The entire system operated on an Ubuntu-based operating system. The model underwent 200,000 iterations with a batch size of 8 and with a training dataset composed of 3521 images. The detection model was trained on the seven fish species.

3.3. Pipeline Breakdown

When the CNN detected a new individual, it attributed to it a unique identifier; a bounding box defined by its position and coordinates; and a species name. If the CNN detected an individual at a given time (t) and failed to detect it at ($t + 1$), the tracking module was activated. For each object, the module utilized the most recent bounding box obtained from the CNN model. Using the object's coordinates, the tracking algorithm initialized and maintained tracking for a predefined number of frames referred to as "timeout frames". During this time, the tracking module remained active, continuously updating the position of the object, unless the object was once again detected by the CNN (Figure 2).

Figure 2. Tracking same fish on three consecutive frames.

3.4. Faster R-CNN for Fish Detection

Video processing started by dividing the video into frames at a specified frame rate of 30 fps. These frames were subsequently fed into a Faster R-CNN, enabling the detection and classification of objects within each frame (region convolutional neural network [15]). R-CNN models consist of four tasks: (1) suggest regions that can contain objects of interest in the frame; (2) extract a fixed-length feature vector from those regions; (3) classify objects found in those regions; (4) and fit bounding boxes around classified objects (Figure 3).

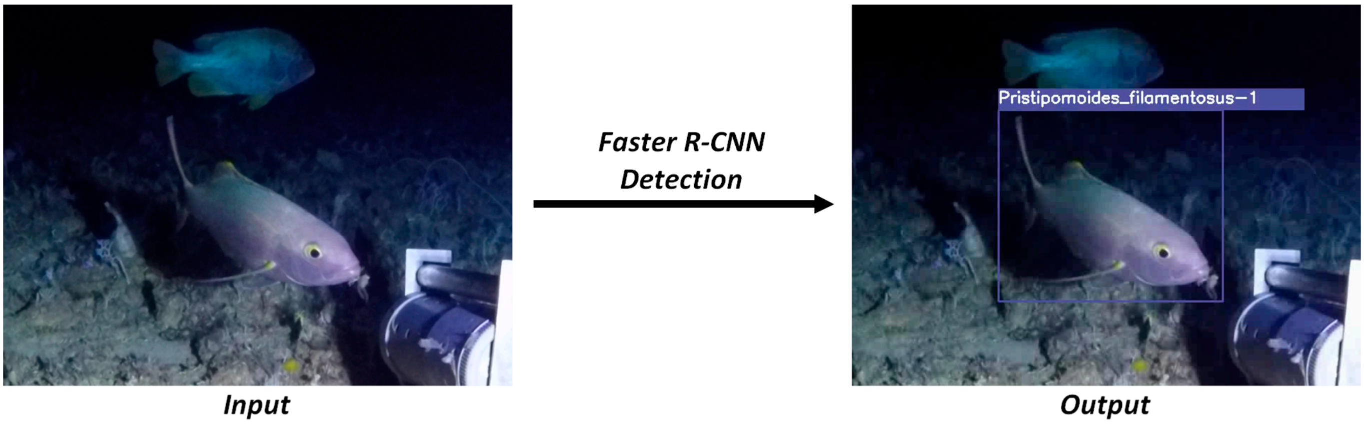

Figure 3. Faster R-CNN object detection on a fish image.

To perform the detection, we used the TensorFlow implementation of the state-of-the-art Faster R-CNN Inception ResNet V2 1024 × 1024 model.

3.5. Tracking Algorithm

The tracking algorithm goal was to keep each object tracked when the detector failed to detect it. Our tracking module took the latest detected bounding box of a fish and predicted the next possible position of the bounding box. The module was able to predict bounding box positions for a set number of frames. This time window was called *timeout frames*. If the detector was able to detect the fish individual during timeout frames, then CNN detection took over the tracking module (Figure 1). If the detector was not able to detect the individual before the end of the *timeout frames*, the individual was no longer considered on screen. This heuristic was very important because it prevented the creation of false negatives bounding boxes in the event of a tracking module failure. When the tracking module was triggered, it maintained the last species attributed to the individual until the detection was resumed again (see pseudo-code in Appendix A).

Detection confidence threshold: A value that we use in order to decide whether we are confident to accept the detected bounding box or throw it away. If we obtain a detection rate that is higher than or equals this threshold, we accept the detection and we take its bounding box and class. If we obtain a detection rate that is lower than this threshold, we assume the detection is not acceptable.

Timeout frames: The value of the timeout frames is chosen based on how many frames the detection model is approximately unable to detect the objects consecutively. The lower the timeout frames, the better, as it may result in more false positives that can decrease the performance of the tracking algorithm. A good value choice may lead to a significant decrease in false negatives at the cost of a slight increase in false positives.

We used the OpenCV legacy tracker [30] algorithm for our tracking module as it provided a more accurate tracking compared to other statistical methods such as Mean Shift [31]. This algorithm took a bounding box region and tried to predict the next possible position for that bounding box based on its color distribution. It also remembered the color distribution for every frame so the object was still trackable.

Given that the tracking system is activated only upon the initial detection's availability, it is possible that in some cases, there may be fish present in the initial frames that go undetected by the detector. Consequently, we may experience a loss of information right at the beginning of the video. To address this issue, we implemented a tracking process on the same videos but in reverse order, moving from the end of the video to the beginning. This approach enables the tracking module to follow and recover fish data that might have been missed during the normal forward tracking process (from the start to the end of the video).

3.6. Matching Metric

We employed Intersection over Union (IoU) (Equation (1)) as our matching metric for bounding boxes. This choice enabled us to automatically link bounding boxes that are in close proximity, ensuring that they correspond to the same object throughout the tracking procedure.

$$\text{Intersection(boxA,boxB)}/\text{Union(boxA,boxB)} \tag{1}$$

boxA and boxB refer to the area of bounding boxes in question. The IoU value is always between 0 and 1, with 0 being the IoU of two bounding boxes with no pixel in common and 1 being two perfectly aligned bounding boxes. The use of IoU allowed us to be sure when comparing two bounding boxes that they are similar in size and position.

3.7. Post-Processing

During the initial two stages of our pipeline (i.e., detection and tracking), we assigned a unique identifier and species label to each individual. The tracking module inherits the class assigned to the detected bounding box by the CNN as long as the *time frame* lasts. Consequently, upon the completion of the video processing, we obtain a comprehensive list encompassing all tracked objects in each frame. For each individual at any given time, this list included its respective IDs, bounding box coordinates, associated class, and bounding box type (obtained from either the model's detection or the tracking module). Nevertheless, it is crucial to acknowledge that this list may contain multiple IDs with approximate bounding box distances and sizes, potentially impacting the tracking score. To address this matter, we perform a post-processing step to eliminate potentially erroneous

IDs from the list. If two bounding boxes exhibit a significant overlap exceeding a selected IoU threshold, they are highly likely to correspond to the same object. This is independent of whether both bounding boxes are from the detector and/or tracker. In such instances, we retained the object associated with the detection bounding box type. Additionally, to refine the class information of each object, we employed a frequency analysis to determine the most representative class. The class that appears most frequently for one individual was selected as the most representative one. Subsequently, any objects with minority classes were reclassified to align with the most representative class. By conducting these post-processing steps, we ensured the accuracy and consistency of the tracked objects' identities, bounding box coordinates, and class information throughout the analysis.

3.8. Evaluation

To evaluate the accuracy of our algorithm, we calculated F1-Scores for each species. Calculating the F1-Score was performed through the following formula:

$$F1Score = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

Precision and recall were represented with the following formulas:

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

Detection accuracy was calculated with the following formula:

$$Det\ Acc = \frac{TP}{TP + FP + FN}$$

True positives (TP): Number of bounding boxes that are correctly detecting the objects;
False positives (FP): Number of bounding boxes that are incorrectly detecting the objects;
False negatives (FN): Number of bounding boxes that are incorrectly not detecting the objects.

It is important to note that true negatives are not important for evaluation because of the following:

The detection model produces a lot of bounding boxes where most of them would be considered true negatives;
Calculating the F1-Score does not require their number;
The F1-Score value varies from 0 to 1, with 1 being the perfect F1-score.

All calculations were conducted on a machine with the following configuration:
CPU: Intel Core-i7 8750H;
GPU: Nvidia GeForce GTX 1060;
RAM: 16 GB DDR4 2667 Mhz.

4. Results

On average, the recall improved significantly with the tracking in two directions (Table 3). In the following results, except for on Table 3, we will consider only the tracking in two directions and simply call it "module".

On average, the F1-Score increased by about 7% going from CNN to CNN + Module in both directions, ranging from a 6.3% increase on *Etelis coruscans* species to 38.7% on *Aprion virescens* species (Table 4).

This improvement mostly came from recall which increased by about 21.6% on average, with a standard deviation of 0.172, as well as a 11.3% increase on *Pristipomoides filamentosus* species being the lowest increase and a 55.9% increase on *Aprion virescens* species being the highest increase (Table 4). However, the decrease in precision for most species affected the F1-Score improvement as well, with *Etelis coruscans* exhibiting a decrease of 8.7%, whereas

Pristipomoides filamentosus only showed a decrease of 4.9%. *Aprion virescens* had the highest decrease by 12.2%, but it did not affect its F1-Score much (Table 4).

Table 3. Average results obtained from tracking using CNN + Module ("Module" below) and CNN + Module in two directions ("Module 2D" below).

Name	Precision		Recall		F1-Score	
	Module	Module 2D	Module	Module 2D	Module	Module 2D
Pristipomoides flavipinnis	0.95	0.934	0.838	0.868	0.889	0.897
Pristipomoides filamentosus	0.966	0.944	0.883	0.889	0.921	0.915
Aprion virescens	0.904	0.86	0.474	0.502	0.612	0.624
Etelis coruscans	0.956	0.913	0.775	0.785	0.84	0.831
Pristipomoides argyrogrammicus	0.935	0.921	0.94	0.951	0.937	0.934
Aphareus rutilans	0.911	0.88	0.774	0.816	0.834	0.843
Average	0.937	0.934	0.781	0.868	0.844	0.846

Table 4. Average of results obtained from tracking using CNN only and CNN + Module in two directions.

Name	Precision		Recall		F1-Score	
	CNN	CNN + Module	CNN	CNN + Module	CNN	CNN + Module
Pristipomoides flavipinnis	1	0.934	0.678	0.868	0.806	0.897
Pristipomoides filamentosus	0.993	0.944	0.799	0.889	0.876	0.915
Aprion virescens	0.98	0.86	0.322	0.502	0.45	0.624
Etelis coruscans	1	0.913	0.69	0.785	0.782	0.831
Pristipomoides argyrogrammicus	1	0.921	0.929	0.951	0.963	0.934
Aphareus rutilans	0.96	0.88	0.668	0.816	0.782	0.843
Total	0.989	0.909	0.681	0.802	0.791	0.846

Pristipomoides argyrogrammicus is an outlier as it is the only species showing a decrease of about 3% in the F1-Score. This is likely due to the false positives that the tracker added with very few false negative corrections. This was demonstrated by the fact that recall did increase by around 2.3%, while precision heavily decreased by 7.9% (Tables 4 and 5).

Table 5. Standard deviation obtained from tracking using CNN only and CNN + Module in two directions.

Name	Precision		Recall		F1-Score	
	CNN	CNN + Module	CNN	CNN + Module	CNN	CNN + Module
Pristipomoides flavipinnis	0	0.083	0.08	0.08	0.056	0.061
Pristipomoides filamentosus	0.015	0.071	0.195	0.124	0.134	0.1
Aprion virescens	0.034	0.028	0.24	0.153	0.251	0.118
Etelis coruscans	0	0.012	0.329	0.234	0.265	0.153
Pristipomoides argyrogrammicus	0	0.102	0.065	0.044	0.035	0.067
Aphareus rutilans	0.049	0.036	0.126	0.135	0.083	0.082
Total	0.016	0.055	0.172	0.128	0.163	0.11

On classes that represented an important part of the sampling, such as *Pristipomoides filamentosus* (Table 1), the impact of our module was less, with the F1-Score increasing from 0.876 to 0.915.

On the other hand, species with very low numbers of samples have seen significant increases in F1-Scores (Figure 4), as seen with *Aprion virescens* (F1-Score increased from 0.45 to 0.624) and *Pristipomoides flavipinnis* (F1-Score increased from 0.806 to 0.897).

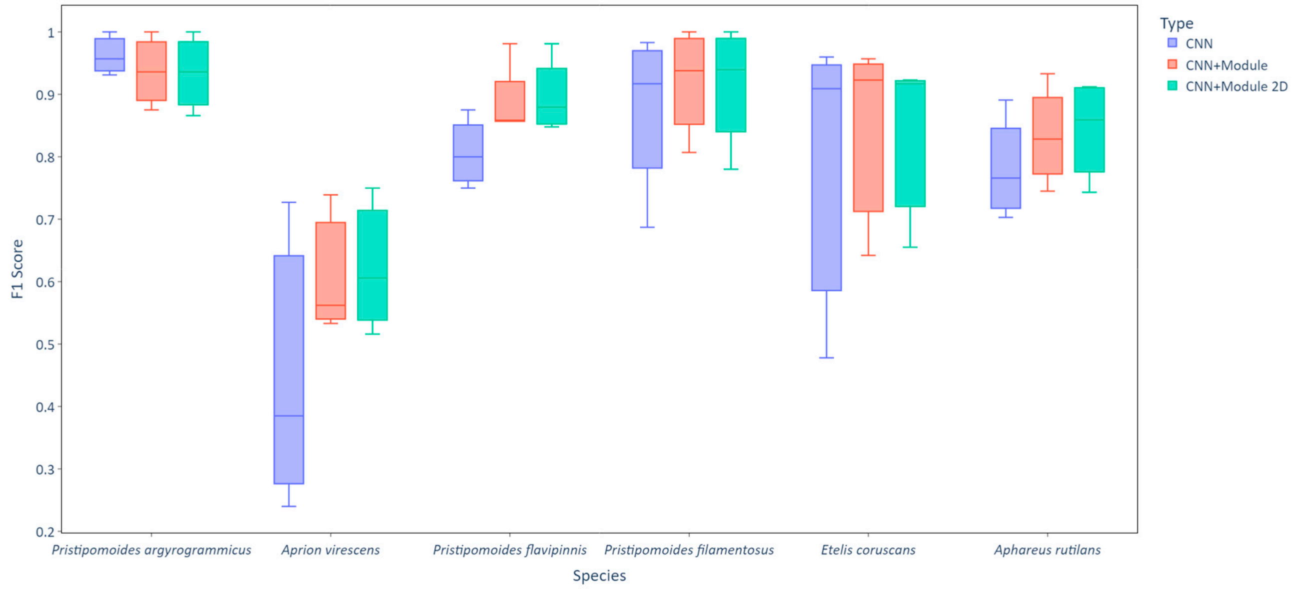

Figure 4. Change in F1-Score after adding tracking module to CNN and tracking in both directions (2D).

On average, the processing time increased by 10.9% (Tables 6 and 7).

Table 6. Average processing times while using CNN only, CNN + Tracker, and CNN + Tracker on reversed videos.

Video Class	Average Time (Seconds)		
	CNN Only	**CNN + Tracker**	**CNN + Tracker Reversed**
Pristipomoides flavipinnis	714	731	807
Aphareus rutilans	728	759	827
Etelis coruscans	725	744	784
Pristipomoides filamentosus	734	770	813
Aprion virescens	744	754	823
Pristipomoides argyrogrammicus	723	740	782

Table 7. Standard deviation for processing times while using CNN only, CNN + Tracker, and CNN + Tracker on reversed videos.

Video Class	Average Time (Seconds)		
	CNN Only	**CNN + Tracker**	**CNN + Tracker Reversed**
Pristipomoides flavipinnis	25.07	58.91	49.3
Aphareus rutilans	5.29	32.9	18.8
Etelis coruscans	2.97	30.03	6.29
Pristipomoides filamentosus	9.16	3.14	31.84
Aprion virescens	3.05	16.12	10.29
Pristipomoides argyrogrammicus	12.75	6.22	16.1

Detection accuracy also increased from CNN Only to CNN + Tracker in two directions from 62.3% to 73.3%.

5. Discussion

From the above results, we noticed an increase in the F1-Score by adding the module. This was the result of the bounding boxes produced by the module, allowing for a decrease in false negatives (non-detected objects). We also noticed a decrease in precision as the tracking module also propagated classification errors from the CNN. This study also assessed the impact on such algorithms for scarce data. With classes composed of numerous samples such as *Pristipomoides filamentosus*, the impact of the module was not significant. On the other hand, species with limited samples have seen significant increases in F1-Scores, as seen with *Aprion virescens* and *Pristipomoides flavipinnis*.

These outcomes were expected since we anticipated that the model would generate a higher number of detections for species with abundant samples. This essentially implies that the likelihood of the model overlooking these species on the screen is quite low, and therefore, the tracker would remain inactive for extended durations. In contrast, for species with limited sample data, we expected the opposite behavior, with the tracker being active for more extended periods. This would enable the correction of numerous false negatives as a consequence (Figures 5 and 6). As marine ecosystems are composed of more rare than common species, the use of those algorithms could help to overcome the data scarcity and the data imbalance between species. Furthermore, as those species have an important impact on the ecosystem [32,33], missing them during fish community census would greatly change the assessment of such communities.

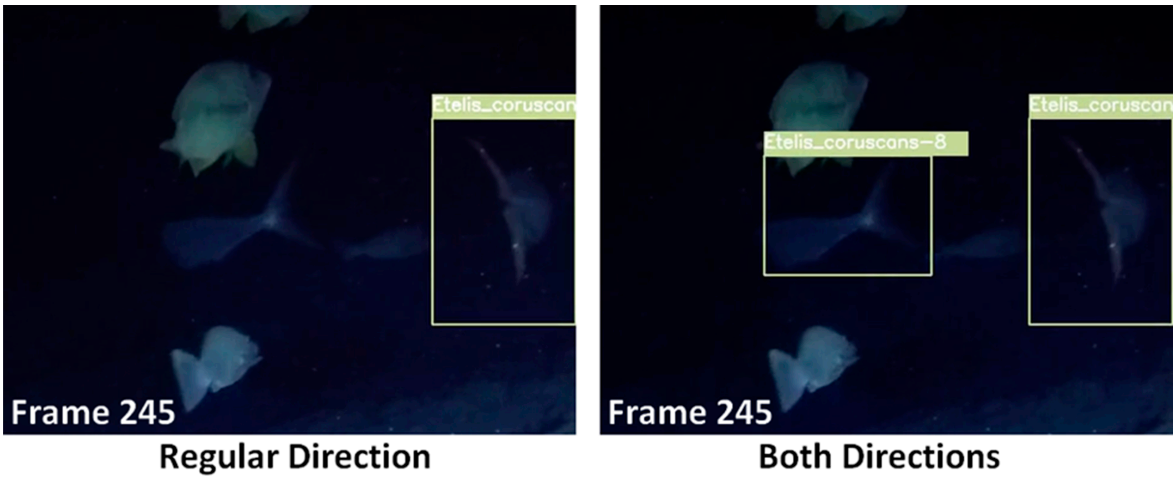

Figure 5. Detection of fish with ID 8 after tracking from the backward direction on frame 245.

Figure 6. Fish detected by the model on frame 251.

On the other hand, if the model detects the objects as early as possible, then the module will give more bounding boxes as false positives, hence the decrease in the F1-Score (Figure 7).

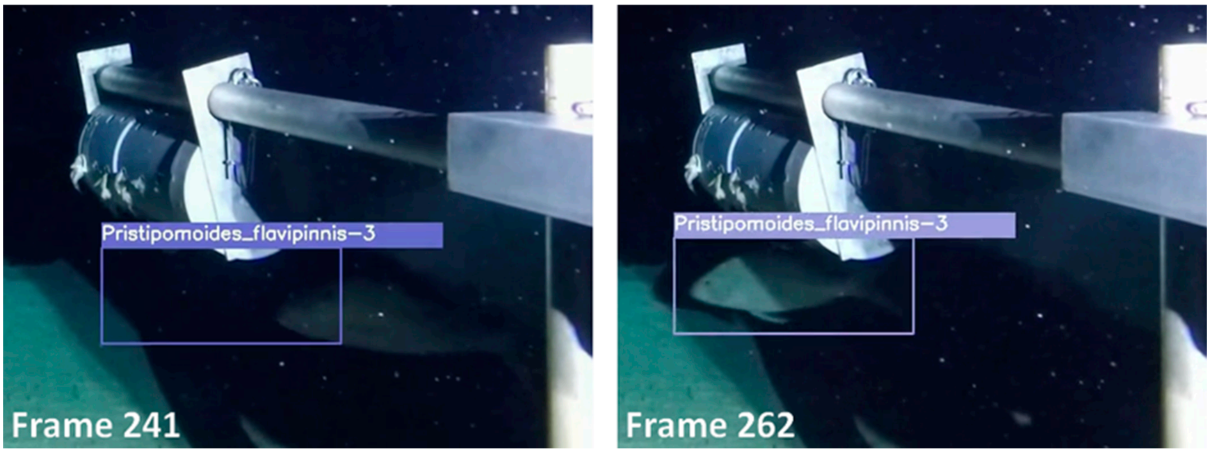

Figure 7. Example of an early false detection of the fish by the module before it is correctly detected by the model.

6. Conclusions

In this paper, we showed the possibilities of multiple object tracking [16,34,35] to improve fish classification and counting. As the state-of-the-art methods only use the image aspect [25], tracking could be the needed tool to leverage the temporal aspect of the videos. Tracking by detection with a deep learning model proved to be more efficient than any other classic way that involved prediction or statistics [10]. However, a well-trained model can likely miss an object and fail to detect it for a couple of frames if the conditions are changing constantly, such as lighting, turbidity, reflection, etc. For videos recorded underwater, such condition changes are happening frequently, making it difficult to detect the same fish consistently. Adding a tracking module could solve the issue for a handful of frames before the model is able to detect the fish once again. Our goal is to connect previous and future detections of the same fish in case there are periods without detections. This approach enables the continuous tracking of fish under the same ID, as demonstrated by our filtering method. Maintaining the same ID could also enhance classification accuracy by allowing us to correct misclassifications based on the predominant classes assigned to the same detected fish. We anticipated that the model would accurately classify the fish most of the time, with only occasional instances of misclassification occurring within minority classes. However, these instances are expected to be rare compared to the majority classifications. In short, it also gives access to the interaction between frames and thus compensates for a large, acknowledged deficiency, especially in marine data composed of many rare species [36].

Moreover, the coupling of tracking and CNNs with a light architecture make it applicable in real-time applications. Of course, the two-direction tracking (backward and forward tracking) is not applicable for real-time applications, but can be used for long campaigns at sea or for long video dataset analyses. One of the limitations of such a method could occur in highly turbid waters, as with all vision-based methods. Overall, our paper shows that coupling the detection model with a tracking module can improve the detection accuracy of fish in an underwater environment.

Author Contributions: Conceptualization, B.Z., L.V. and S.V.; Methodology, B.Z., J.Z., L.V. and S.V.; Software, B.Z.; Validation, S.V.; Investigation, B.Z., J.Z. and S.V.; Resources, L.V.; Data curation, B.Z., F.B. and S.V.; Writing—original draft, B.Z.; Writing—review & editing, J.Z., F.B., L.V. and S.V.; Visualization, B.Z.; Supervision, J.Z., L.V. and S.V.; Project administration, S.V.; Funding acquisition, J.Z. and L.V. All authors have read and agreed to the published version of the manuscript.

Funding: The study was funded by grant ANR "SEAMOUNTS" #ANR-18-CE02-0016, the French Oceanographic Fleet, and IRD core funding. The study was supported by the research consortium ANR-AFD "AIME".

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The raw data supporting the conclusions of this article will be made available by the authors on request, according to the lab politics.

Acknowledgments: Data were collected under permits 2019-733/GNC, 2020-503/GNC and 2020-1077/GNC delivered by the Government of New-Caledonia, 898-2019/ARR/DENV, 3066-2019/ARR/DENV, 844-2020/ARR/DDDT, and 1955-2020/ARR/DDDT delivered by the Southern Province of New-Caledonia, and 609011/2019/DEPART/JJC, 609011-18/2019/DEPART/JJC, and 609011-39/2020/DEPART/JJC delivered by the Northern Province of New-Caledonia.

Conflicts of Interest: Author Florian Baletaud was employed by the company Soproner, Groupe GINGER. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Appendix A

Pseudo-code:

Here is the procedure for the new method:

Initialize a variable called "begin" as True.

Iterate over each frame of the video and perform Object Detection.

If there are no detections and "begin" is True, go to step 2.

If there are no detections and "begin" is False, go to step 7.

If there are detections and "begin" is True, set the detected bounding boxes as reference boxes with new IDs (1 to n where n is the number of objects), then set "begin" as False and go back to step 2.

For each new bounding box,

If it matches a reference box above a specified IoU threshold, replace the reference box with the new box.

If there is no match, add the new box as a reference box with a new ID $n + 1$.

For each unmatched reference box,

Apply a tracking module to estimate the new box's position based on previous tracked boxes.

Add the resulting box with the same ID.

Go back to step 2 and repeat step 7 for the currently unmatched objects for a specified timeout frames or until the object is detected again.

Thresholds:

Detection Threshold: 0.5.

Timeout frames: 20.

IoU Threshold: 0.35.

References

1. Halpern, B.S.; Frazier, M.; Afflerbach, J.; Lowndes, J.S.; Micheli, F.; O'hara, C.; Scarborough, C.; Selkoe, K.A. Recent pace of change in human impact on the world's ocean. *Sci. Rep.* **2019**, *9*, 11609. [CrossRef]
2. Halliday, W.D.; Brittain, S.A.; Niemi, A.; Majewski, A.R.; Mouy, X.; Insley, S.J. The Underwater Soundscape of Minto Inlet, Northwest Territories, Canada. *ARCTIC* **2022**, *75*, 462–479. [CrossRef]

3. Emslie, M.J.; Bray, P.; Cheal, A.J.; Johns, K.A.; Osborne, K.; Sinclair-Taylor, T.; Thompson, C.A. Decades of monitoring have informed the stewardship and ecological understanding of Australia's Great Barrier Reef. *Biol. Conserv.* **2020**, *252*, 108854. [CrossRef]
4. Danovaro, R.; Fanelli, E.; Aguzzi, J.; Billett, D.; Carugati, L.; Corinaldesi, C.; Dell'anno, A.; Gjerde, K.; Jamieson, A.J.; Kark, S.; et al. Ecological variables for developing a global deep-ocean monitoring and conservation strategy. *Nat. Ecol. Evol.* **2020**, *4*, 181–192. [CrossRef] [PubMed]
5. Lønborg, C.; Thomasberger, A.; Stæhr, P.A.U.; Stockmarr, A.; Sengupta, S.; Rasmussen, M.L.; Nielsen, L.T.; Hansen, L.B.; Timmermann, K. Submerged aquatic vegetation: Overview of monitoring techniques used for the identification and determination of spatial distribution in European coastal waters. *Integr. Environ. Assess. Manag.* **2022**, *18*, 892–908. [CrossRef]
6. Terracciano, D.; Bazzarello, L.; Caiti, A.; Costanzi, R.; Manzari, V. Marine Robots for Underwater Surveillance. *Curr. Robot. Rep.* **2020**, *1*, 159–167. [CrossRef]
7. Jian, M.; Qi, Q.; Yu, H.; Dong, J.; Cui, C.; Nie, X.; Zhang, H.; Yin, Y.; Lam, K.-M. The extended marine underwater environment database and baseline evaluations. *Appl. Soft Comput.* **2019**, *80*, 425–437. [CrossRef]
8. Janiesch, C.; Zschech, P.; Heinrich, K. Machine learning and deep learning. *Electron. Mark.* **2021**, *31*, 685–695. [CrossRef]
9. Iqbal, M.A.; Wang, Z.; Ali, Z.A.; Riaz, S. Automatic Fish Species Classification Using Deep Convolutional Neural Networks. *Wirel. Pers. Commun.* **2021**, *116*, 1043–1053. [CrossRef]
10. Mouillot, D.; Chaumont, M.; Darling, E.S.; Subsol, G.; Claverie, T.; Villéger, S. A Deep learning method for accurate and fast identification of coral reef fishes in underwater images. *Ecol. Inform.* **2018**, *48*, 238–244. [CrossRef]
11. Saleh, A.; Sheaves, M.; Jerry, D.; Azghadi, M.R. Applications of deep learning in fish habitat monitoring: A tutorial and survey. *Expert Syst. Appl.* **2023**, *238*, 121841. [CrossRef]
12. Lopez-Marcano, S.; Jinks, E.L.; Buelow, C.A.; Brown, C.J.; Wang, D.; Kusy, B.; Ditria, E.M.; Connolly, R.M. Automatic detection of fish and tracking of movement for ecology. *Ecol. Evol.* **2021**, *11*, 8254–8263. [CrossRef] [PubMed]
13. Hu, J.; Zhao, D.; Zhang, Y.; Zhou, C.; Chen, W. Real-time nondestructive fish behavior detecting in mixed polyculture system using deep-learning and low-cost devices. *Expert Syst. Appl.* **2021**, *178*, 115051. [CrossRef]
14. Whang, S.E.; Roh, Y.; Song, H.; Lee, J.-G. Data collection and quality challenges in deep learning: A data-centric AI perspective. *VLDB J.* **2023**, *32*, 791–813. [CrossRef]
15. Pal, S.K.; Pramanik, A.; Maiti, J.; Mitra, P. Deep learning in multi-object detection and tracking: State of the art. *Appl. Intell.* **2021**, *51*, 6400–6429. [CrossRef]
16. Luo, W.; Xing, J.; Milan, A.; Zhang, X.; Liu, W.; Kim, T.-K. Multiple object tracking: A literature review. *Artif. Intell.* **2021**, *293*, 103448. [CrossRef]
17. Pérez-Escudero, A.; Vicente-Page, J.; Hinz, R.C.; Arganda, S.; de Polavieja, G.G. idTracker: Tracking individuals in a group by automatic identification of unmarked animals. *Nat. Methods* **2014**, *11*, 743–748. [CrossRef]
18. Rodriguez, A.; Zhang, H.; Klaminder, J.; Brodin, T.; Andersson, P.L.; Andersson, M. *ToxTrac*: A fast and robust software for tracking organisms. *Methods Ecol. Evol.* **2018**, *9*, 460–464. [CrossRef]
19. Rasch, M.J.; Shi, A.; Ji, Z. Closing the loop: Tracking and perturbing behaviour of individuals in a group in real-time. *bioRxiv* **2016**, 071308.
20. Li, W.; Li, F.; Li, Z. CMFTNet: Multiple fish tracking based on counterpoised JointNet. *Comput. Electron. Agric.* **2022**, *198*, 107018. [CrossRef]
21. Wang, H.; Zhang, S.; Zhao, S.; Wang, Q.; Li, D.; Zhao, R. Real-time detection and tracking of fish abnormal behavior based on improved YOLOV5 and SiamRPN++. *Comput. Electron. Agric.* **2022**, *192*, 106512. [CrossRef]
22. Liu, S.; Zheng, X.; Han, L.; Liu, X.; Ren, J.; Wang, F.; Liu, Y.; Lin, Y. FishMOT: A Simple and Effective Method for Fish Tracking Based on IoU Matching. *Appl. Eng. Agric.* **2024**, *40*, 599–609. [CrossRef]
23. Walter, T.; Couzin, I.D. TRex, a fast multi-animal tracking system with markerless identification, and 2D estimation of posture and visual fields. *Elife* **2021**, *10*, e64000. [CrossRef] [PubMed]
24. Romero-Ferrero, F.; Bergomi, M.G.; Hinz, R.C.; Heras, F.J.H.; de Polavieja, G.G. idtracker.ai: Tracking all individuals in small or large collectives of unmarked animals. *Nat. Methods* **2019**, *16*, 179–182. [CrossRef] [PubMed]
25. Villon, S.; Iovan, C.; Mangeas, M.; Vigliola, L. Toward an artificial intelligence-assisted counting of sharks on baited video. *Ecol. Inform.* **2024**, *80*, 102499. [CrossRef]
26. Baletaud, F.; Lecellier, G.; Gilbert, A.; Mathon, L.; Côme, J.-M.; Dejean, T.; Dumas, M.; Fiat, S.; Vigliola, L. Comparing Seamounts and Coral Reefs with eDNA and BRUVS Reveals Oases and Refuges on Shallow Seamounts. *Biology* **2023**, *12*, 1446. [CrossRef]
27. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards real-time object detection with region proposal networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2015**, *39*, 1137–1149. [CrossRef]
28. Lin, T.-Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; Zitnick, C.L. Microsoft Coco: Common Objects in Context. In Proceedings of the Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, 6–12 September 2014; Proceedings, Part V 13. pp. 740–755.
29. Abadi, M.; Agarwal, A.; Barham, P.; Brevdo, E.; Chen, Z.; Citro, C.; Corrado, G.S.; Davis, A.; Dean, J.; Devin, M.; et al. TensorFlow: Large-scale machine learning on heterogeneous systems. *arXiv* **2015**, arXiv:1603.04467.
30. Janku, P.; Koplik, K.; Dulik, T.; Szabo, I. Comparison of tracking algorithms implemented in OpenCV. *MATEC Web Conf.* **2016**, *76*, 04031. [CrossRef]

31. Snekha, C.S.; Birok, R. Real Time Object Tracking Using Different Mean Shift Techniques—A Review. *Int. J. Soft Comput. Eng. (IJSCE)* **2013**, *3*, 98–102.
32. Leitão, R.P.; Zuanon, J.; Villéger, S.; Williams, S.E.; Baraloto, C.; Fortunel, C.; Mendonça, F.P.; Mouillot, D. Rare species contribute disproportionately to the functional structure of species assemblages. *Proc. R. Soc. B Biol. Sci.* **2016**, *283*, 20160084. [CrossRef] [PubMed]
33. Zhang, Z.; Lu, Y.; Wei, G.; Jiao, S. Rare Species-Driven Diversity–Ecosystem Multifunctionality Relationships are Promoted by Stochastic Community Assembly. *mBio* **2022**, *13*, e0044922. [CrossRef] [PubMed]
34. Zheng, L.; Tang, M.; Chen, Y.; Zhu, G.; Wang, J.; Lu, H. Improving multiple object tracking with single object tracking. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 2453–2462.
35. Chu, Q.; Ouyang, W.; Li, H.; Wang, X.; Liu, B.; Yu, N. Online Multi-object Tracking Using CNN-Based Single Object Tracker with Spatial-Temporal Attention Mechanism. In Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017.
36. Iovan, C.; Mangeas, M.; Claverie, T.; Mouillot, D.; Villéger, S.; Vigliola, L. Automatic underwater fish species classification with limited data using few-shot learning. *Ecol. Inform.* **2021**, *63*, 101320. [CrossRef]