ICAR Research Report n° 0001

Some considerations to design an autonomous buoy system to enumerate pelagic sharks at Fish Aggregating Devices

Constantin Fite[1,2]
Mehdi Yedroudj[1,2]
Marc Chaumont[1,3]
Pierrick Serres-Noïtaky[4]
Frédéric Comby[1]
Fabien Forget[2]
Gérard Subsol[1]
Laura Mannocci[1,2,5]
Manuela Capello[2]
Mariana Tolotti[2]
Laurent Dagorn[2]

[1]Research-Team ICAR, LIRMM, Univ. Montpellier CNRS, Montpellier, France.

[2]MARBEC, Univ. Montpellier, CNRS, Ifremer, IRD, Sète, France.

[3]Univ. Nîmes, France.

[4]NOPIBOTIC, Montpellier, France.

[5]FRB-CESAB, Montpellier, France.

C. Fite, M. Yedroudj, M. Chaumont, P. Serres-Noïtaky, F. Comby, F. Forget, G. Subsol L. Mannocci, M. Capello, M. Tolotti, L. Dagorn. Some considerations to design an autonomous buoy system to enumerate pelagic sharks at Fish Aggregating Devices. ICAR Research Report n°0001, August 2024. https://www.lirmm.fr/icar/reports.html

August 13 2024, v1.1

Some considerations to design an autonomous buoy system to enumerate pelagic sharks at Fish Aggregating Devices

Constantin Fite[1,2], Mehdi Yedroudj[1,2], Marc Chaumont[1,3], Pierrick Serres-Noïtaky[4], Frédéric Comby[1], Fabien Forget[2], Gérard Subsol[1], Laura Mannocci[1,2,5], Manuela Capello[2], Mariana Tolotti[2], and Laurent Dagorn[2]

[1]Research-Team ICAR, LIRMM, Univ. Montpellier CNRS, Montpellier, France
[2]MARBEC, Univ. Montpellier, CNRS, Ifremer, IRD, Sète, France
[3]Univ. Nîmes, France
[4]NOPIBOTIC, Montpellier, France
[5]FRB-CESAB, Montpellier, France

August 13, 2024

Abstract

Fish Aggregating Devices (FADs) are floating objects used by fishers to facilitate their catches. The majority of the global industrial tropical tuna purse-seine catches currently occurs at FADs. One of the main adverse ecological impacts of FAD fisheries is the bycatch of vulnerable species such as pelagic sharks. Detecting the presence of sharks at FADs remotely, constitutes a key step to reduce their catches.

In this paper, we explain how an image-based shark detection module can be embedded on an autonomous buoy associated to a FAD. We discuss the general design, the hardware and software selection, the implementation of the detection algorithm based on Deep-Learning techniques and we detail the implementation inside the buoy. Some experiments allowed us to discuss energy consumption. This system could be integrated with other new technologies to mitigate bycatch in industrial tropical tuna purse-seine fisheries.

1 Introduction

About 66 percent of the global tuna landings is captured by the tropical tuna purse seine fishery (TTPSF) (see Figure 1) representing annually 5.1 million tonnes [JRR21]. In the mid 2000s, large commercial purse seiners started to intensively use man made floating objects, known as Fish Aggregating Devices (FADs), to locate and capture schools of tuna. This fishing mode, representing 38% of global tuna catches [MDJR$^+$21], relies on the associative behavior of pelagic species with floating objects, including target species, to concentrate and localize tuna schools. Since the 2010s, echosounder buoys have been deployed at drifting FADs to remotely provide information on their position and estimate the aggregated biomass underneath them [LMSM14]. The use of echosounder buoys further enhanced the efficiency of the industrial TTPSF [WGKG20]. As a consequence, purse-seiners shifted their fishing effort on drifting FADs with respect to other alternative fishing modes, targeting free swimming schools of tuna that are feeding on the sea surface or schools associated with marine mammals [Hal92, DHRM13]. Aggregations at FADs are multi-specific. About 20 species, regularly encountered in the open ocean, [FDM$^+$20] form part of FAD aggregations. Consequently, when deploying their nets to surround the tuna schools at FADs, purse seiners also capture other non-targeted species which are referred to as bycatch. Fishing at FADs thus incurs 2.6-6.7 times more bycatch (ocean dependent) than when fishing on free swimming schools of tuna [DHRM13]. The magnitude of the FAD-based fishery has generated conservation concerns (see Figure 2). In particular, the silky (*Carcharinus falciformis*) and oceanic white tip (*Carcharinus longimanus*) sharks are two major pelagic shark species that are incidentally captured

at FADs [DHRM13]. These shark species are both listed in the convention on International Trade in Endangered Species (CITES, Appendix II) and classified as Vulnerable and Critically Endangered by the international Union for Conservation of Nature (IUCN), with the oceanic whitetip shark recently listed as threatened under the United States Endangered Species Act [YCH$^+$18].

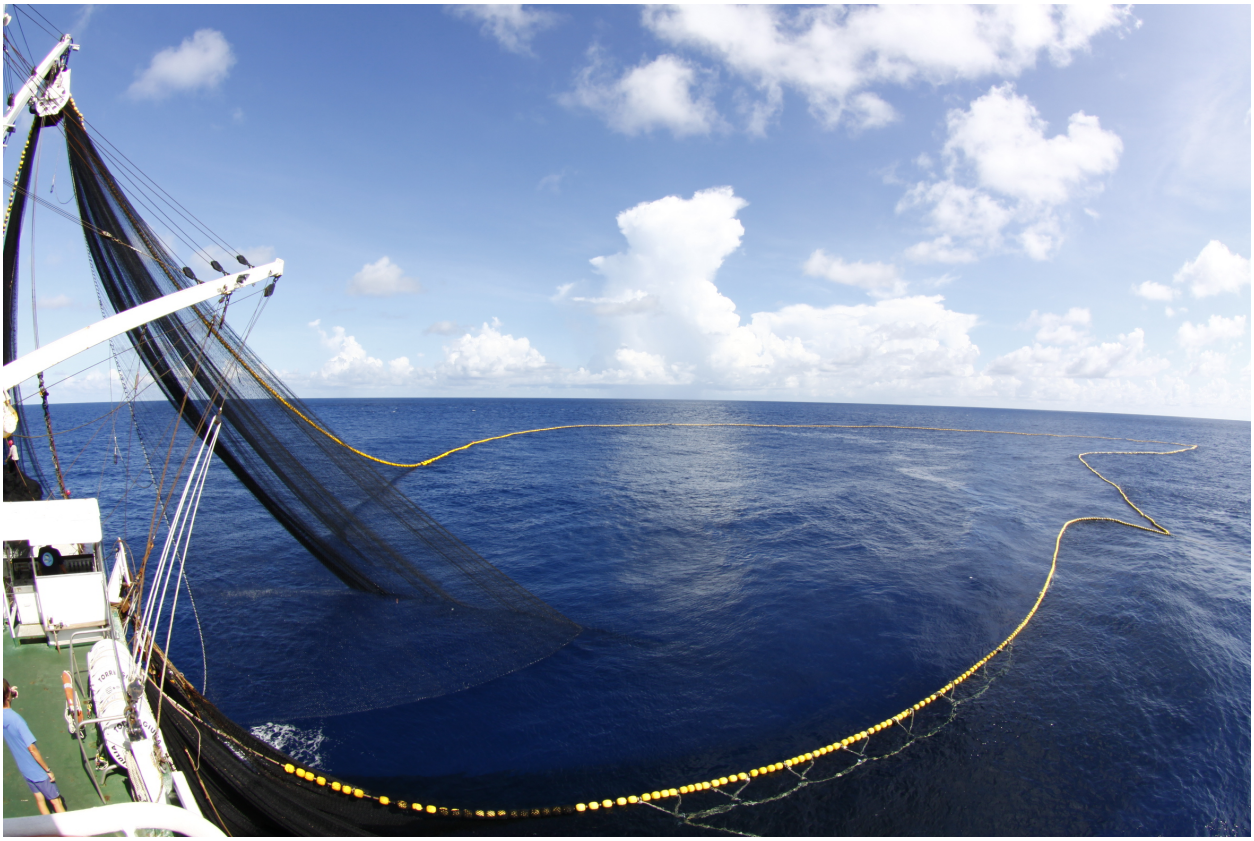


Figure 1: Purse seine deployment used to catch tropical tuna. Copyright ISSF, Fabien Forget (2012).

Research efforts are directed towards finding bycatch mitigation methods that could reduce fishing induced mortality of silky sharks in TTPSF. Studies which investigated mortality of silky sharks released from the purse seiners following good handling practices agreed that survival rates of released individuals was generally low (around 15%) [PFVD14, HIMH15, EBB16]. Other studies have subsequently explored technical mitigation measures by removing the sharks from the net. [RDM$^+$16, FCF$^+$15] also investigated the associative patterns and vertical distributions at FADs of targeted tuna species and major bycatch species, including silky sharks, in order to characterize and identify species-specific behavioural patterns that could help to improve the selectivity of the gear throughout the day. Silky sharks displayed similar associative patterns and a shallower vertical distribution than tunas, indicating few opportunities for technical mitigation solutions.

The fishing grounds of the TTPSF are extensive, so getting remote information at FADs is crucial to fishers in planning and adapting their fishing strategy. Moreover, recently, it was suggested that dynamic fishery closures, as opposed to static one could be far more efficient at protecting pelagic species [PWO$^+$22]. Echo-sounders attached to FADs can provide in real time the tuna biomass estimation at FADs and play an important role in the choice of the fishing area. Similarly, real time information on the presence of Endangered, Threatened and Protected Species (ETP) species could be helpful to avoid areas with high occurrence rates. [MBF$^+$21] investigated the possibility of applying machine learning on echo-sounder buoy outputs to categorize FADs with high or low bycatch risks. However, echo-sounder buoys are non effective at detecting or discriminating species, in particular sharks.

Underwater visual census at FADs by divers have been successfully used to characterize bycatch species [TSD$^+$07, FDM$^+$20]. The idea of using visual monitoring have been widely used for many ecological studies (see for example chapter 7 of [CHS18] for the sharks). In [BST$^+$19], the authors

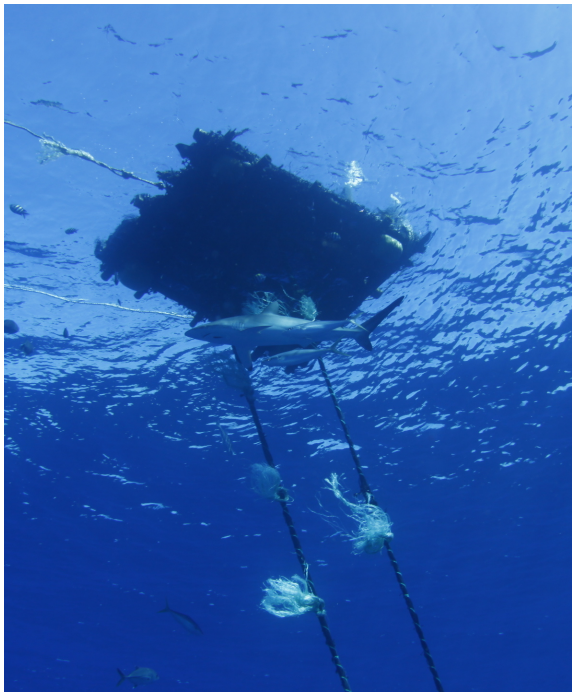


Figure 2: Subsurface view of a Fish Aggregating Device. Notice the presence of a silky shark. Copyright ISSF, Fabien Forget (2012).

designed an autonomous buoy prototype which incorporates a 360° image acquisition device based on a turning camera. They conclude that such a system is the best compromise between a pole-mounted video on a vessel and diver surveys. The number of fishes observed were less than those observed by divers but they hypothesize that with a longer duration of video acquisition, such system could approach the performances of divers at short distances from the FAD. They also emphasize the problems of energy consumption and data transmission.

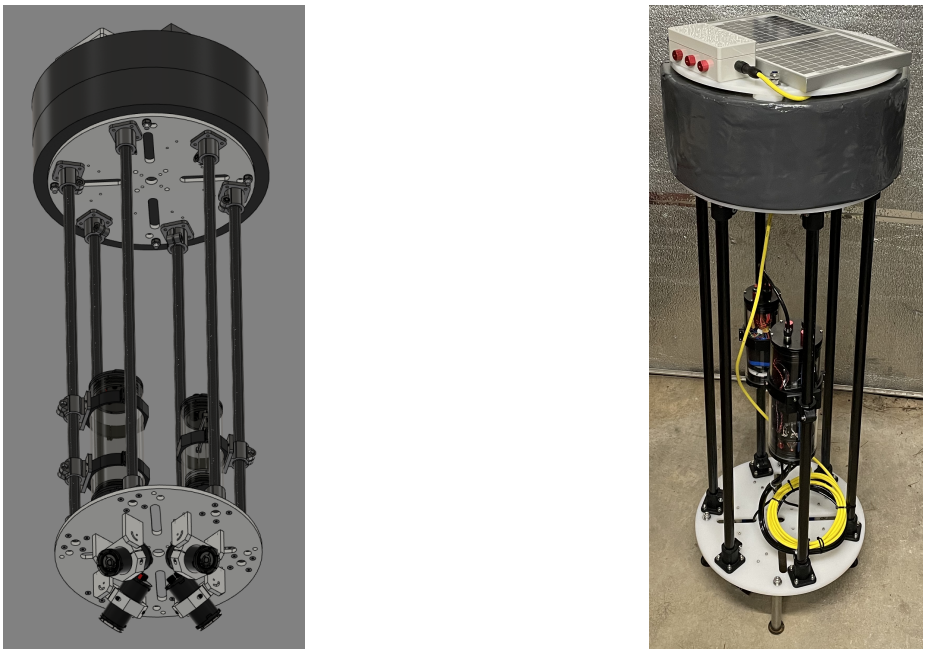
In this paper, we develop this idea by designing a shark monitoring system based on an autonomous buoy supporting 4 synchronized cameras at 90° from each other in order to have an overview of the fishes present under the FAD. In order to get a reliable estimation of sharks, it is necessary to perform an acquisition during several minutes with a frequency of several images per second. This will give sequences of more than a thousand images which have to be analyzed to assess the presence of sharks. As it is not possible to send this large quantity of images by satellite phone for visual inspection, the detection of sharks must be done automatically on-board and only the number of sharks in each image can be sent. Of course, this raises questions about the algorithm to develop to get reliable results, the embedded computer hardware to run in an efficient way and the fine tuning of power consumption with respect to the available energy.

In section 2, we present the general design of the system with its technical characteristics. In section 3, we present the detection and localization algorithm which is based on a deep learning architecture. In section 4, we detail how to integrate the algorithm and its hardware in an image processing module and we discuss the different solutions for switching on the system and scheduling the calculations while accounting for performance and energy consumption. In section 5, we describe some preliminary experiments in a shark tank and in the last section, we discuss future work.

2 General design of the system

Our system is presented in Figure 3. The buoy is made of an upper part with a partially emerged float unit. There are 6 rods (which can be removed and replaced by longer ones) which are fixed to the float unit for their upper part and to a rigid disk for their lower part. The rods are 1.10 m long and forms a cage which protects two cylindrical airtight cases fixed to two of the 6 rods. One airtight case (76 mm diameter × 250 mm) contains the battery (lithium ion 4S3P) and a converter, and the other one (101

mm diameter × 320 mm) contains the image processing module. Finally, 4 HD cameras are fixed on the lower part of the buoy, which are approximately to 1.2 to 1.3 m below the sea surface. The final prototype weight is 18.3 kg.



a) 3D Computer Assisted Design model b) Final prototype

Figure 3: *Illustration of our system. a) 3D CAD model. b) The actual prototype. The 4 cameras are located underwater at 1.2 to 1.3 meters from the float unit and point slightly downwards. The image processing module is in an airtight case.*

The objective is to build a system which is similar in shape and size compared to the standard buoys embedding an echo-sounder.

Note that FADs can drift for months or even years in the open sea. Thanks to solar panels the system can be self-sufficient. But the float unit design imposes a strong constraint on energy production as the solar panels are fixed above it (see Figure 3). The area of the solar panel is thus limited by the area of the float unitn which has a diameter of 50 cm. As shown in Figure 3b, the solar panels are rectangular with a surface of around 0.1 m^2 which gives a total nominal power of 10 W.

3 Image processing algorithm

3.1 Shark detection by Deep Learning

The aim of the algorithm is to detect in a frame of the video sequence all the sharks presents nearby the buoy. This last decade, Deep Learning methods, based on the use of neural networks, have proven to be much more efficient than traditional image processing ones for such a task, even if there are only a couple of applications for shark detection.

Generally, in the field of Deep Learning, object detection methods are divided into two categories [LOW+20]. The first category follows the traditional object detection pipeline: in a first step, bounding boxes, delineating image parts in which the object could be, are defined and in a second step, these image parts are classified according to a list of object classes which includes the target (here, a shark). Each step is performed by a separate network, so this category is referred as **two-stage object detection** methods. The second category handles the whole object detection task as a unique problem. Therefore, a single network is used to perform both localization by a bounding box and classification, hence the name **one-stage object detection** methods. In all methods, the output is a list of bounding boxes associated to a class and a detection score. By setting a threshold on this detection score, we can discard the least reliable bounding boxes.

Two-stage methods have been used for some years to detect fishes. For example, [LSQC15] proposed to used a network architecture called *Fast R-CNN* [Gir15] which is able to localize and classify 12 classes of reef fishes. An improvement of this architecture called *Faster R-CNN* [RHGS15] was used to detect (i.e. localize and classify) hammerhead sharks in videos acquired by divers [UVB+20]. But, when there are several sharks in a video frame, the results of detection remains limited with many missed sharks. There are also many cases where the algorithm correctly localizes the bounding box around a shark but classifies it as another fish species. Two recent methods based on *Faster R-CNN* are described in [JLC+22] and [VIMV24]. Performances are high for respectively eight and three shark species in BRUV (Baited Remote Underwater Video Station) recordings. Notice that these videos are acquired from a static point of view located on the seabed which may make them very different of videos acquired from a moving buoy.

We can find two applications of shark detection based on the one-stage network architecture *Yolo v3* [FR18]. In [PPBH20], the authors extend this architecture by adding layers which allow to perform detection at several scales and increase performance. The method was evaluated on the same application of hammerhead shark detection as above but on other video frames. The results of detection are quite limited (only 1/4 to 2/3 of sharks are correctly detected and classified). In [MSAF+21], *Yolo v3* is applied to detect great white sharks in an image dataset taken from Kaggle[1]. Detection results seem interesting (between 40% to 86% are correctly detected and classified) but only one video was used for the evaluation and the images contain only one shark individual which simplifies the detection task.

3.2 Image dataset description

We collected a dataset of 3,618 images, extracted from 64 videos. The videos were acquired with a GoPro camera by divers at a depth of about 5 meters ar drifting and anchored FAD in the Western Indian Ocean, the Maldives and the Pacific Ocean. Each image contains at least one shark (silky shark *Carcharhinus falciformis* or oceanic Whitetip shark *Carcharhinus longimanus*) or another fish (bigeye trevally *Caranx sexfasciatus*, pilot fish *Naucrates ductor*, rainbow runner *Elagatis bipinnulata*, copper saupe *Kyphosus vaigiensis*). For each image, marine ecology experts manually delineated a bounding box around each animal they were able to recognize and annotated it with a class label (see Figure 4). Notice that partially visible animals were also labelled if possible.

Images	Bounding boxes	Shark BB	Fish BB
3,618	6,623	4,909	1,714
Training set			
3,055	5,658	4,056	1,602
Test set			
563	965	853	112

Table 1: Our annotated dataset is composed of images containing at least one shark (silky shark, whitetip shark) or one fish (bigeye trevally, pilot fish, rainbow runner, copper saupe). For each image, bounding boxes (BB) were delineated by marine ecologists around every shark and fish.

The image dataset was then randomly partitioned into a training (85%) and a test sets (15%) as indicated in Table 1. We ensured that images from the training and the test set belong to separate videos for a fair evaluation. The native resolution of the images varied from 2,704×1,524 to 1280×960 pixels depending on video. All images were resampled to 640×640 pixels which is the size required by the detection algorithms tested in this study.

3.3 Performance assessment

We tested the two cited networks *Faster R-CNN* and *Yolo v3*. We compared them to *Retina-Net* [LGG+20] and *Yolo v5*[2] which works well with dense and small scale objects. Both can be considered as state-of-the-art one stage network architectures.

Yolo v5 comes with four different versions[3] that vary from the smaller version *Yolo v5s* to the extra-large version *Yolo v5x*. All versions are pre-trained on the *COCO val2017* image dataset [LMB+14] which

[1] https://www.kaggle.com/

[2] `https://github.com/ultralytics/yolov5`. Notice that we can find more recent extensions in [GLW+21].

[3] `https://github.com/ultralytics/yolov5/releases/tag/v4.0`

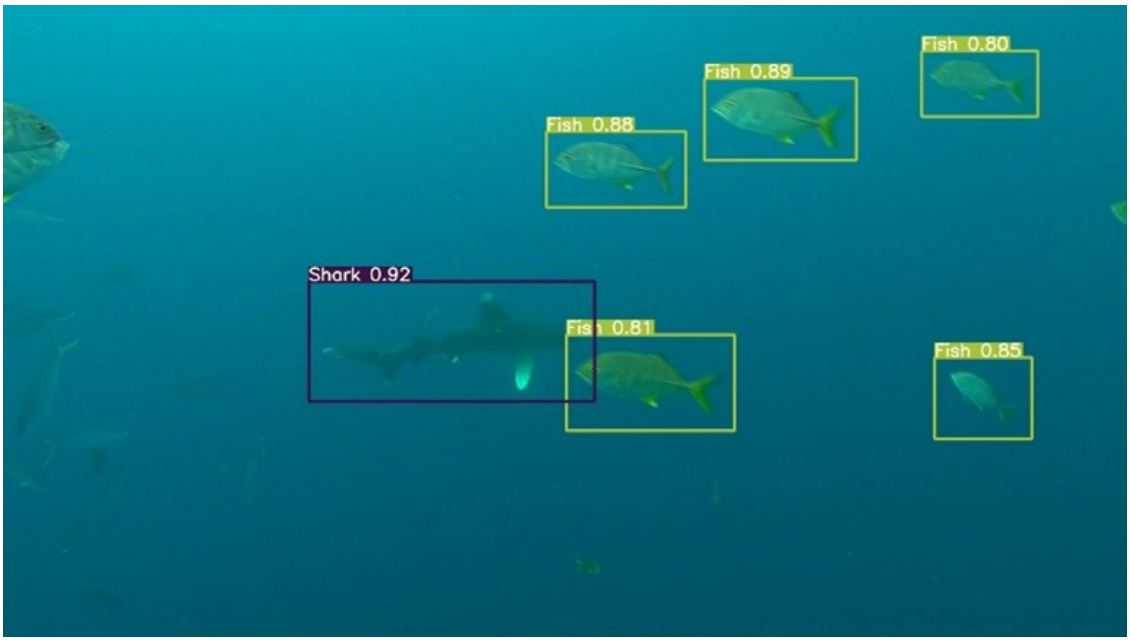


Figure 4: *Bounding boxes in an image. One was manually annotated as Shark, and the others as Fish.*

contains 120,000 images from 80 different classes such as person, bike, car, bus, cat, etc. We selected the extra-large network *Yolo v5x* which contains 87.7 million parameters in order to obtain the best results, although it takes longer to run compared to *Yolo v5s*.

Our objective was to detect sharks so we could think that it is sufficient to train the network with images labeled with bounding boxes around the sharks. The network computes then features to distinguish a shark from the rest of the image, that consists mostly in a blue background. Nevertheless, fishes, which are the other objects in the image, share the same kind of contrast with the background. So, the features computed to detect a shark are also active for a fish and may trigger a detection, resulting in a confusion between a fish and a shark. It is then very important to train the network to detect all the classes which may be confused. In the following, all the experiments will use at least the *Shark* and *Fish* classes.

We ran various experiments with the 4 network architectures - *Faster-RCNN* with a *ResNet* backbone, *Retina-Net* with a *SSD* backbone, *Yolo v3* and *Yolo v5x*. We used the overall training dataset and also some subsets to see the influence of the number of training frames. We tested 2-class (*Shark*, *Fish*) but also 3-class (*Silky shark*, *Whitetip shark*, *Fish*)) detection. All experiments were conducted on a powerful workstation integrating a Quadro RTX6000 NVIDIA owning 24 Gb of memory.

Results were evaluated by assessing the overlap between the bounding boxes detected by the network and the manually delineated bounding boxes with the Intersection over Union (IoU) indicator (also knwon as Jaccard index) [LMB+14] tuned to 0.5. It is then possible to see which bounding boxes are correctly detected and which ones are missed or misidentified. By varying the threshold on the detection score, we computed for each class the *Average Precision* AP@.5 (.5 stands for the IoU parameter). By averaging them over all the classes weighted by the number of detected objects, we obtained a global performance indicator called *mean Average Precision* (mAP@.5).

Architecture	mAP@.5	AP@.5 Shark	AP@.5 Fish
Retina-Net	59 %	91 %	27 %
Yolo v3	40 %	60 %	20 %
Faster-RCNN	74 %	92 %	56 %
Yolo v5x	90 %	95 %	86 %

Table 2: mAP and AP for the different detection networks evaluated on our image dataset.

For all the architectures, we obtained the best results when using the entire dataset and the 2-class

(*Shark* and *Fish*) version. This could be explained by the fact that the two shark species are visually similar. The results are presented in Table 2.

Our findings, revealed that *Yolo v5* is the most efficient network both for fish and shark detection. We also noticed that it is less time-consuming than the other architectures, which is consistent with some benchmarks[4]. *Yolo v5x* was then the selected network for our system.

The network takes about a day to train on our high-performance workstation but this is done just once on land; the resulting files are then uploaded on the embedded system.

4 Integrating software and hardware

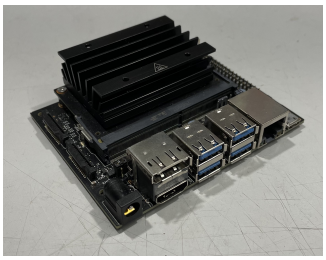
4.1 Choice of the embedded computer

The image processing module of the system integrates a minicomputer which runs the shark detection algorithm. Its features - volume, weight and electrical consumption - must fit with the general system specifications.

At the time of our research, we found find different minicomputers on the market, such as the Raspberry Pi[5], the Lattepanda[6],the Coral Dev Board[7] or the NVIDIA Jetson[8]. All those micro-computers costed less than 150 dollars and have low energy consumption.

As reported before, *Yolo v5* was one of the most efficient Deep Learning algorithms for shark detection. To be fast, it has to be implemented on a GPU (Graphic Processing Unit) [BD18] using the dedicated parallel computing platform CUDA developed by NVIDIA[9]. In Raspberry and Lattepanda mini-computers, the GPU was integrated to the CPU and was not compatible with the NVidia architecture. Coral Dev Board came only with a CPU but a TPU (Tensor Processing Unit), which is a separate hardware dedicated to massive multiplications and additions for neural networks, could be plugged on it. Looking to available benchmarks[10,11], the Coral Dev Board minicomputer with a TPU was faster than a NVIDIA Jetson Nano. Nevertheless, the compatibility of *Yolo v5* with a TPU was not tested. In contrast, a large community already deployed projects with *Yolo v5* running on NVIDIA Jetson minicomputers. We then selected a NVIDIA Jetson Nano minicomputer which is a small, powerful minicomputer for embedded Deep Learning applications. In particular, it requires a very limited electrical power, as little as 5 W.

a) One of the 4 cameras.



b) The Jetson Nano minicomputer.

Figure 4.1.a shows one of the 4 cameras of the system. They are connected to a switch that is linked via IP protocol to the Jetson Nano minicomputer displayed in 4.1.b with a RJ45 cable.

4.2 Shark detection module implementation

The performance tests in section 3.3 were obtained with the extra-large version *Yolo v5x*. We implemented it on the Jetson Nano but we faced two problems. First, we got a detection rate of only 1 frame per second (FPS). Considering all the cameras which send 4 synchronized images, the module is then able to detect a shark only every 4 seconds, that is clearly not efficient for moving animals. Secondly, the

[4] https://towardsdatascience.com/yolov5-compared-to-faster-rcnn-who-wins-a771cd6c9fb4
[5] https://www.raspberrypi.org/
[6] https://www.lattepanda.com/
[7] https://coral.ai/products/dev-board/
[8] https://developer.nvidia.com/embedded-computing
[9] https://developer.nvidia.com/cuda-zone
[10] https://tryolabs.com/blog/machine-learning-on-edge-devices-benchmark-report/
[11] https://developer.nvidia.com/embedded/jetson-nano-dl-inference-benchmarks

Jetson Nano has difficulties to manage such a extra-large architecture network. This requires too much memory which causes the algorithm to crash after the analysis of only some frames.

We then assessed the performances of the small and medium version of *Yolo v5* with standard parameters.

The results obtained with *Yolo v5s* are 5% to 6% lower in AP@.5 for the *Shark* and the *Fish* classes that with *Yolo v5x*. Nevertheless, we consider that this will ensure a sufficient detection quality considering the small memory occupation and a fast processing compatible with the acquisition rate of the 4 cameras.

4.3 Optimizing power consumption

The entire processing chain of treatment of our detection module is as follows:

- the Jetson minicomputer retrieves the images which are acquired by the 4 cameras;
- it runs the detection algorithm *Yolo v5s* which outputs labelled bounding boxes. The results (in particular the number of sharks) are stored in a text file;
- the system sends the content of this file via a satellite communication channel.

Video acquisition by cameras and computation on the Jetson minicomputer are independent processes which can be switched on or off by an external controller of the system. We use it to find the best processing pipeline combining a small power consumption with an optimal acquisition and detection rate. We analyzed the three following solutions:

1. *Sequential* pipeline: the 4 images of the different cameras are immediately sent to, and processed by the *YOLO v5s* network. Once the detection is done, the minicomputer treats the next group of 4 images. In this process, the 4 cameras and the minicomputer are then continuously running and nothing is switched on or off.
2. *Asynchronous* pipeline: each sequence of images from any of the 4 cameras is recorded and stored separately in a specific file. The detection by *YOLO v5s* is performed later after reading these 4 files. For this solution, cameras are switched-on during the acquisition process and then switched-off. After switching-off the cameras, the network is run on the 4 files.
3. *Concatenation/asynchronous* pipeline: images of the 4 cameras are acquired separately, but it is possible to concatenate them into one single image before recording and storing it. This unique but 4 times larger image is then asynchronously processed by the minicomputer as in the previous solution. This technique allows to reduce file opening and reading times.

For each solution, we evaluated the required power to process a 1 min video (constituted of the 4 video streams acquired by the 4 cameras). Camera resolution is set to standard HD that is $1,280 \times 720$ pixels. We hypothesized that an camera acquisition rate of 5 FPS is suitable for our application and we discuss this point at the end of this section. Each camera thus generates 300 images in 1 mn, making a total of 1,200 images to analyze.

Notice that by default, *Yolo v5s* resamples the input image to 640×640 pixels to analyze it. But then, in solution 3 where the resulting image contains $2,560 \times 1280$, the subsampling ratio would double with regards to solutions 1 or 2. So, for solution 3, we set the *Yolo v5* resampling parameter to $1,280 \times 1,280$ in order to be able to compare detection performances with solutions 1 and 2.

We measured power consumption over time with a voltmeter for the cameras and with the *jetson_stats* package[12] for the Jetson. Once a camera is launched, it has a fixed power of 2.3 W which gives 9.2 W for the 4 cameras. Once the Jetson is started, the detection at full power takes around 5 W. This gives us the power consumption curves over time presented in Figure 5.

In solution 1 (see Figure 5, top), the 4 cameras and the Jetson are continuously working after the starting phase. In the two other solutions (see Figure 5, middle and bottom), we can distinguish a camera acquisition phase at the beginning and then, a detection phase where only the Jetson is running.

In solution 3, the analyzed image is subsampled to a size of $1,280 \times 1,280$. This represents the same pixel number as for the 4 subsampled images of 640×640 in solution 2. Nevertheless, we can see by comparing Figure 5, middle and bottom, that the duration of the detection phase is different: it takes around 400 s in solution 2 but only 240 s in solution 3. Moreover, in solution 3, the detection process requires to transfer 4 times less images (even if they are larger) which optimizes the memory use of the Jetson and accelerates the *Yolo v5s* algorithm implementation.

[12] `https://github.com/rbonghi/jetson_stats`

After doing some experiments, we noticed that the sequential solution 1 is not able to treat all the 1,200 images of the 1 minute sequence. In fact, the detection phase itself takes more than 0.05 s by image, preventing to process the 4 images in less than the 0.2 s required by the acquisition rate of 5 FPS.

Pipeline	Duration in s total/detection	Energy in J=W.s
S1. Sequential (only 1 FPS)	180 / 60	2,384 J
S2. Asynchronous (5 FPS)	555 / 400	3,671 J
S3. Concatenation /asynchronous (5 FPS)	395 / 240	2,556 J

Table 3: Duration and energy consumption of the Jetson Nano to process 1 min long video.

All results are summarized in Table 3. If we want a detection rate of 5 FPS, the best solution, in terms of energy consumption, is clearly the *Concatenation/asynchronous* pipeline. It gives an average consumption of less than 6.5 W with a peak at around 13 W. As the nominal power given by the solar panels of the system is 10 W, it lets a good margin for the consumption of the other electronic devices (GPS and satellite communication in particular).

Note that this solution could be run at any acquisition rate as the detection process is done asynchronously after image acquisition. One major concern is to assess that a rate of 5 FPS is sufficient to not miss a shark crossing the field of view of a camera at a standard speed (which is approximated by 1 body-length per second). We selected a 300 s video sequence where there are 0, 1 or 2 sharks and no other fish. Shark trajectories and speed were also assessed as standard. We then evaluated the detection results obtained at 5 FPS and 25 FPS which is a standard rate for video recording. We found that all the detected bounding boxes, at both 5 and 25 FPS, are correctly located around the sharks (i.e. there is no false positive). If we sum the detection scores of all the detected bounding boxes in an image, the value should be as closest as possible to the actual number of sharks and quantifies the confidence of the overall detection process. In Figure 6, we display the actual number of sharks as purple lines and respectively in orange (up) and blue (bottom) the results for 5 FPS and 25 FPS. As there is no significant differences, we consider that a frame rate of 5 FPS is sufficient to detect all the sharks in an image and we use this acquisition rate to get low energy consumption and less data to transfer.

4.4 Making the system autonomous

Figure 7 illustrates the complete detection process. The master system, running on an Arduino platform, switches on the Jetson minicomputer which records and concatenates the video streams of the 4 cameras in an unique file, then switches off the camera, and launches the *Yolov5s* detection algorithm. For each frame, we get the number of bounding boxes identified as a shark which is written sequentially in a file. As a clock and a GPS are connected to the minicomputer, we add to the file the time and GPS position of the system to know when and where the observations were made.

Once the video file is analyzed, the Jetson minicomputer sends the file content, thanks to a dedicated satellite communication Iridium device. In order to reduce the transmitted data size, we use an off-the-shelf compression algorithm. Then, the minicomputer sends a message to the master system indicating that it is going to turn off and finally, the master system cuts the power. All day long, the power is restored and the minicomputer automatically repeats the whole chain.

The nominal power of the system is 10 W which corresponds to the power which can be produced by the solar panels under standard test conditions (sunlight intensity of 1,000 W.m^{-2}). We used the dedicated tool from the Photovoltaic Geographical Information System[13] with a 0° slope at the Maldives position to estimate the energy production by the solar panels of the system in this Indian ocean area which are crossed by a large number of FADs. The value is between 1.2 and 1.6 kWh a month which gives a minimal available energy of 40 W.h=144,000 J by day.

The day last around 12 hours at equatorial latitudes so the minimal sunlight intensity required to record videos at a depth of 1 m is reached for approximately 9 hours. We will assume that a video acquisition of 5 min is appropriate to detect sharks and estimate their number at a given time with a

[13] `https://re.jrc.ec.europa.eu/pvg_tools/fr/#PVP`

good reliability. According to Table 3, this will require at maximum $5 \times 395 = 1,975$ s (around 33 mn) of processing and $5 \times 2,556 = 12,780$ J for the detection phase. We can then perform one sequence of 5 min acquisition and processing per hour during 9 hours a day. This leaves more than 25% of the produced energy for the other devices (GPS ans satellite communication) and some time to send data. Moreover, if we measure exactly the energy consumption of the detection phase for 5 mn, it is only 11,600 J so 10% less as we do not not have to start the cameras and the minicomputer each minute.

5 Preliminary experiments

In order to test the whole system in the most realistic conditions, the system was deployed in the shark tank of the Planet Ocean aquarium[14] in Montpellier, France (see Figure 8).

The main purpose of this experiment was to evaluate the design of the system and its behavior in the water. It also allowed to evaluate the consumption of the detection module. The environment of the shark tank differs from the open sea one. In the aquarium, there are rocks, a sandy bottom, and light spots that create shadows whereas, in the ocean, there is a very uniform background, which creates a contrast with the sharks (see Fig. 9). Moreover, the shark species present in the aquarium are sand tiger shark *Carcharias taurus*, grey shark *Carcharhinus amblyrhynchos*, leopard shark *Triakis semifasciata* and we can observe sawfishes. These shark and fish species are not present in our training dataset and more generally not around FAD in tropical pelagic waters.

Notice that, due to some hardware problem, we had to replace, at the last minute, the Jetson Nano minicomputer by a more powerful version, the Jetson Xavier NX. If the Jetson Xavier NX minicomputer consumes 8 W, that is 2 W more than the Jetson Nano, it takes 4 times less to perform the detection. As a whole, we can estimate that it is 1.5 times more efficient in energy consumption.

The system was installed in the tank (see Figure 10 and 11). As planned, two people could easily manipulated it. The buoyancy and watertightness were assessed. Camera orientation was correct and allowed to see the bottom with a good stability (even if there is no wave in the tank).

Energy consumption was assessed by connecting the system to a computer located on the shore of the tank via an Ethernet cable. The reported measure gives a consumption of 7 W for the Jetson Xavier NX minicomputer alone, 10 W for the 4 cameras and around 6 supplementary W for running the detection algorithm. The total consumption during a complete switch-on cycle, switch-off of the camera and the detection process are thus oscillating between 13 W and 17 W, which is higher (by $\approx$ 3-4 W) than what we estimated, using the *jetson_stats* package. This difference may be explained by two reasons. First, the external controller consumption was not taken into account during the measurement by the software. Secondly, the cable linking the minicomputer to the power consumption measurement device was quite long and this may lead to an overestimation of the real consumption.

If we refer to Figure 3, we will have an increased consumption of around 3 W when the Jetson Xavier NX minicomputer begins its process during the first 155 s (that is 465 J more). As it is 4 times faster than the Jetson Nano, the consumption during the detection phase itself will be 13 W during $240/4 = 60$ s (that is 780 J) instead of 5.2 W during 240 s (that is 1248 J) with the Jetson Nano. In conclusion, the two minicomputer have a comparable energy consumption for the sequence of 5 min acquisition and processing per hour during 9 hours a day, which complies with the energy capacity of the system.

As there is an important mismatch between the images acquired in the tank and the ones which belong to the training dataset, the detection algorithm performed badly. We diminished the detection score threshold to a low value (0.4) in order to detect some sharks while avoiding false positives. As observed in Fig. 12, sharks are correctly detected only when there is a uniform background as it is the case in the open-water. The algorithm was not able to generalize well to the tank environment or to new shark species. Nevertheless, these first experiments showed us that the image acquisition and processing chain is fully operational.

6 Discussion and future work

In this paper, we explained how a shark *detection module* can be embedded on an autonomous buoy. We discussed the general design, the hardware and software selection to implement the detection algorithm and we detailed the implementation. Experiments allowed us to discuss energy consumption which is a

[14] `https://www.planetoceanworld.fr/`

key-factor. The presented solution is the result of many discussions between researchers in computer-science, robotics and marine ecology. We think that this system could be integrated with other new technologies to mitigate bycatch in industrial tuna fisheries [PBC+21].

Two contributions can be highlighted. First, we demonstrated the feasibility of embedding a state-of-the art detection algorithm in a minicomputer, with a sufficient detection rate and a small power consumption to be able to perform experiments in the open-sea. Secondly, we described and assessed a complete processing sequence, from the minicomputer starting which combines camera acquisition, object detection and classification and the transmission of the results via satellite phone.

The next step would be to test the system in open-water in the Indian ocean [FFY+22]. This will allow us to assess if all the specifications are really efficient, in particular the performances of shark detection and the power autonomy.

This system could also be extended to other threatened species that are bycaught in FAD-based fishing, such as turtles.

Declarations

Authors' contributions

All authors participated to the design of the autonomous buoy system and the analysis of results. Constantin Fite, Mehdi Yedroudj, Marc Chaumont and Gérard Subsol wrote the main manuscript text. All authors reviewed then the manuscript.

Funding

This project was funded by the European Project INNOV-FAD (European Maritime and Fisheries Fund, measure no 39, OSIRIS#PFEA390017FA1000004 and France Filière Pêche (FFP)) led by the marine ecology laboratory MARBEC (IRD/Ifremer/INRAE/CNRS/University of Montpellier), France.

We thank the Planet Ocean aquarium, Montpellier, France for giving us access to the shark tank for our preliminary experiments.

Availability of data and materials

At the moment, data and materials are not made available.

References

[BD18] Ebubekir Buber and Banu Diri. Performance Analysis and CPU vs GPU Comparison for Deep Learning. In *2018 6th International Conference on Control Engineering Information Technology (CEIT)*, pages 1–6, 2018.

[BST+19] Patrice Brehmer, Gorka Sancho, Vasilis Trygonis, David Itano, John Dalen, Ariel Fuchs, Abdelmalek Faraj, and Marc Taquet. Towards an Autonomous Pelagic Observatory: Experiences from Monitoring Fish Communities around Drifting FADs. *Thalassas: An International Journal of Marine Sciences*, 35(1):177–189, April 2019.

[CHS18] Jeffrey C. Carrier, Michael R. Heithaus, and Colin A. Simpfendorfer, editors. *Shark Research: Emerging Technologies and Applications for the Field and Laboratory*. CRC Press, Boca Raton, 1st edition edition, September 2018.

[DHRM13] Laurent Dagorn, Kim N. Holland, Victor Restrepo, and Gala Moreno. Is it good or bad to fish with FADs? What are the real impacts of the use of drifting FADs on pelagic marine ecosystems? *Fish and Fisheries*, 14(3):391–415, 2013. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1467-2979.2012.00478.x.

[EBB16] Corey Eddy, Richard Brill, and Diego Bernal. Rates of at-vessel mortality and post-release survival of pelagic sharks captured with tuna purse seines around drifting fish aggregating devices (FADs) in the equatorial eastern Pacific Ocean. *Fisheries Research*, 174:109–117, 2016.

[FCF+15] Fabien G Forget, Manuela Capello, John D Filmalter, Rodney Govinden, Marc Soria, Paul D Cowley, and Laurent Dagorn. Behaviour and vulnerability of target and non-target species at drifting fish aggregating devices (FADs) in the tropical tuna purse seine fishery determined by acoustic telemetry. *Canadian Journal of Fisheries and Aquatic Sciences*, 72(9):1398–1405, 2015.

[FDM+20] F. Forget, L. Dagorn, B. Mérigot, J. C. Gaertner, J. Robinson, P. D. Cowley, M. S. Adam, Y. Rilwan, M. Koonjul, V. Mangar, M. Taquet, and F. Ménard. Beta diversity of pelagic assemblages at fish aggregating devices in the open ocean. *African Journal of Marine Science*, 42(2):247–254, 2020.

[FFY+22] Fabien Forget, Constantin Fite, Mehdi Yedroudj, Mariana Tolotti, Marc Chaumont, Pierrick Serres-Noïtaky, Frédéric Comby, Manuela Capello, and Laurent Dagorn. Preliminary results of an autonomous buoy prototype to count pelagic sharks at FADs. In *IOTC - 3rd Ad Hoc Working Group on FADs. IOTC-2022-WGFAD03-11*, 2022.

[FR18] Ali Farhadi and Joseph Redmon. Yolov3: An incremental improvement. *Computer Vision and Pattern Recognition, cite as*, 2018.

[Gir15] Ross Girshick. Fast R-CNN. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015.

[GLW+21] Zheng Ge, Songtao Liu, Feng Wang, Zeming Li, and Jian Sun. YOLOX: exceeding YOLO series in 2021. *CoRR*, abs/2107.08430, 2021.

[Hal92] Martin A. Hall. The association of tunas with floating objects and dolphins in the Eastern Pacific Ocean. In *Part VI. Association of fauna with floating objects. Background document for the International Workshop on the Ecology and Fisheries for Tunas Associated with Floating Objects*, page 7, 1992.

[HIMH15] Mr Hutchinson, Dg Itano, Ja Muir, and Kn Holland. Post-release survival of juvenile silky sharks captured in a tropical tuna purse seine fishery. *Marine Ecology Progress Series*, 521:143–154, 2015.

[JLC+22] J. Jenrette, Z. Y.-C. Liu, P. Chimote, T. Hastie, E. Fox, and F. Ferretti. Shark detection and classification with machine learning. *Ecological Informatics*, 69:101673, 2022.

[JRR21] Ana Justel-Rubio and Loreno Recio. A Snapshot of the Large-Scane Tropical Purse Seine Fishing Fleets as of July 2021. Technical report, July 2021.

[LGG+20] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal Loss for Dense Object Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(2):318–327, February 2020. Conference Name: IEEE Transactions on Pattern Analysis and Machine Intelligence.

[LMB+14] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence" Zitnick. Microsoft coco: Common objects in context. In *Proceedings of the European Conference on Computer Vision, ECCV'2014*, pages 740–755, Zurich, Switzerland, September 2014. Springer International Publishing.

[LMSM14] Jon Lopez, Gala Moreno, Igor Sancristobal, and Jefferson Murua. Evolution and current state of the technology of echo-sounder buoys used by Spanish tropical tuna purse seiners in the Atlantic, Indian and Pacific Oceans. *Fisheries Research*, 155:127–137, jul 2014.

[LOW+20] Li Liu, Wanli Ouyang, Xiaogang Wang, Paul Fieguth, Jie Chen, Xinwang Liu, and Matti Pietikäinen. Deep Learning for Generic Object Detection: A Survey. *International Journal of Computer Vision*, 128(2):261–318, February 2020.

[LSQC15] Xiu Li, Min Shang, Hongwei Qin, and Liansheng Chen. Fast accurate fish detection and recognition of underwater images with Fast R-CNN. In *OCEANS 2015 - MTS/IEEE Washington*, pages 1–5, October 2015.

[MBF+21] Laura Mannocci, Yannick Baidai, Fabien Forget, Mariana Travassos Tolotti, Laurent Dagorn, and Manuela Capello. Machine learning to detect bycatch risk: Novel application to echosounder buoys data in tuna purse seine fisheries. *Biological Conservation*, 255:109004, March 2021.

[MDJR+21] Hilario Murua, Laurent Dagorn, Ana Justel-Rubio, Gala Moreno, and Victor Restrepo. Questions and Answers about FADs and Bycatch. Technical report, June 2021.

[MSAF+21] Nino E. Merencilla, Alvin Sarraga Alon, Glenn John O. Fernando, Elaine M. Cepe, and Dennis C. Malunao. Shark-EYE: A Deep Inference Convolutional Neural Network of Shark Detection for Underwater Diving Surveillance. In *2021 International Conference on Computational Intelligence and Knowledge Economy (ICCIKE)*, pages 384–388, March 2021.

[PBC+21] François Poisson, Pierre Budan, Sylvain Coudray, Eric Gilman, Takahito Kojima, Michael Musyl, and Tsutomu Takagi. New technologies to improve bycatch mitigation in industrial tuna fisheries. *Fish and Fisheries*, n/a(n/a), 2021.

[PFVD14] François Poisson, John David Filmalter, Anne-lise Vernet, and Laurent Dagorn. Mortality rate of silky sharks (Carcharhinus falciformis) caught in the tropical tuna purse seine fishery in the Indian Ocean. *Canadian Journal of Fisheries and Aquatic Sciences*, 71(6):795–798, 2014.

[PPBH20] Alvaro Peña, Noel Pérez, Diego S. Benítez, and Alex Hearn. Tracking Hammerhead Sharks With Deep Learning. In *2020 IEEE Colombian Conference on Applications of Computational Intelligence (IEEE ColCACI 2020)*, pages 1–6, August 2020.

[PWO+22] Maite Pons, Jordan T. Watson, Daniel Ovando, Sandra Andraka, Stephanie Brodie, Andrés Domingo, Mark Fitchett, Rodrigo Forselledo, Martin Hall, Elliott L. Hazen, Jason E. Jannot, Miguel Herrera, Sebastián Jiménez, David M. Kaplan, Sven Kerwath, Jon Lopez, Jon McVeigh, Lucas Pacheco, Liliana Rendon, Kate Richerson, Rodrigo Sant'Ana, Rishi Sharma, James A. Smith, Kayleigh Somers, and Ray Hilborn. Trade-offs between bycatch and target catches in static versus dynamic fishery closures. *Proceedings of the National Academy of Sciences*, 119(4):e2114508119, 2022.

[RDM+16] Victor Restrepo, Laurent Dagorn, Gala Moreno, Fabien Forget, Kurt Schaefer, Igor Sancristobal, Jeff Muir, and David Itano. Compendium of ISSF At-Sea Bycatch Mitigation Research Activities as of 12/2016. Technical report, 2016.

[RHGS15] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In *Proceedings of the Advances in Neural Information Processing Systems, NIPS'2015*, volume 28, pages 91–99, Montreal, Canada, December 2015. Curran Associates, Inc.

[TSD+07] Marc Taquet, Gorka Sancho, Laurent Dagorn, Jean-Claude Gaertner, David Itano, Riaz Aumeeruddy, Bertrand Wendling, and Christophe Peignon. Characterizing fish communities associated with drifting fish aggregating devices (FADs) in the Western Indian Ocean using underwater visual surveys. *Aquatic Living Resources*, 20(4):331–341, 2007.

[UVB+20] Gabriela Ulloa, Vicente A. Vásconez, Diego S. Benítez, Noel Pérez, and Alex Hearn. Hammerhead Shark Detection Using Regions with Convolutional Neural Networks. In *2020 IEEE ANDESCON*, pages 1–6, October 2020.

[VIMV24] Sébastien Villon, Corina Iovan, Morgan Mangeas, and Laurent Vigliola. Toward an artificial intelligence-assisted counting of sharks on baited video. *Ecological Informatics*, 80:102499, 2024.

[WGKG20] Gwenaëlle Wain, Loreleï Guéry, David Michael Kaplan, and Daniel Gaertner. Quantifying the increase in fishing efficiency due to the use of drifting FADs equipped with echosounders in tropical tuna purse seine fisheries. *ICES Journal of Marine Science*, 78(1):235–245, 12 2020.

[YCH+18] C. N. Young, J. Carlson, M. Hutchinson, C. Hutt, D. Kobayashi, C. T. McCandless, and J. Wraith. Status Review Report: Oceanic Whitetip Shark (Carcharhinus longimanus). Final Report to the National Marine Fisheries Service, Office of Protected Resources National Marine Fisheries Service. Technical report, 2018.

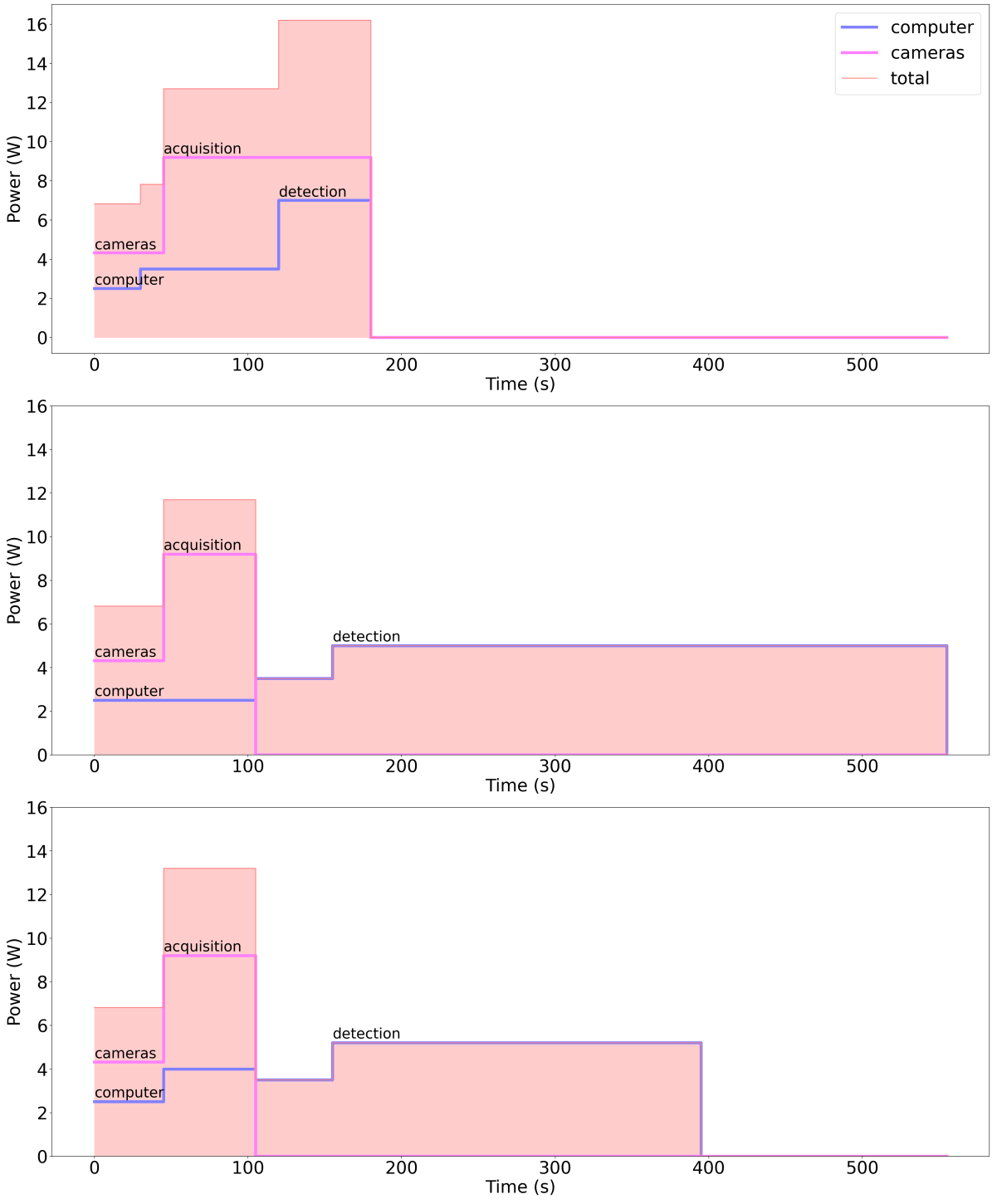


Figure 5: Power consumption (in W) over time (in s) for the 3 solutions to process a 60 s video with an acquisition rate of 5 FPS. The pink area represents energy consumption in J=W.s. Computer consumption can be decomposed in starting the Jetson microcomputer, initializing the software and running the detection algorithm itself. Cameras consumption consists in starting the cameras and performing the acquisition itself. From top to bottom: 1. *Sequential* pipeline (but it results that this solution is limited to process only 1 FPS), 2. *Asynchronous* pipeline and 3. *Concatenation/asynchronous* pipeline.

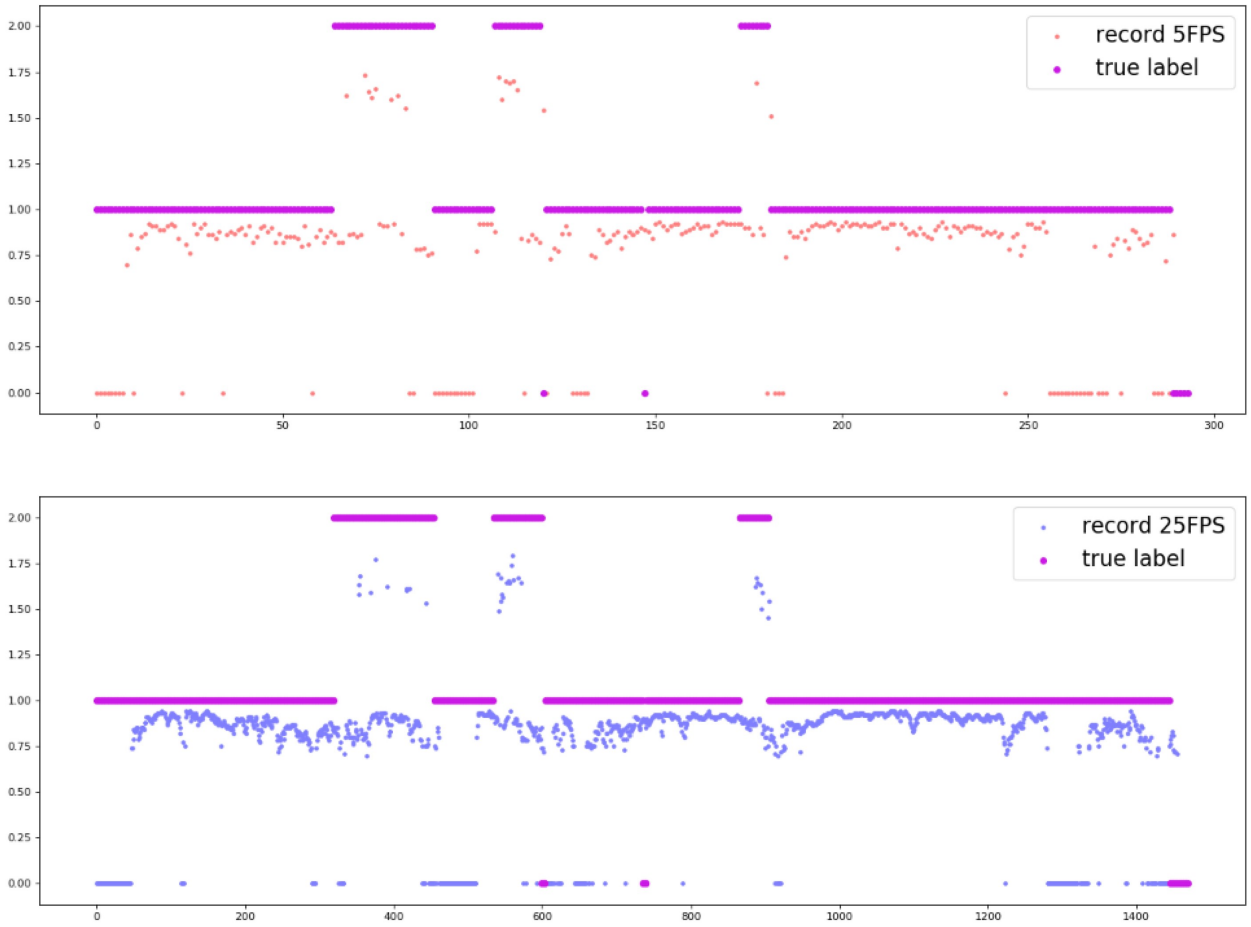


Figure 6: Performance comparisons at a rate of 5 (up) and 25 (bottom) FPS with the *concatenation/asynchronous* solution on a 300 s long video containing 0, 1 or 2 sharks by image. In orange and blue is displayed the sum of the detection scores of the detected bounding boxes. It should ne as closest as possible to the purple lines which indicates the actual number of sharks in the image. Notice that performances at 5 and 25 FPS can be considered as similar.

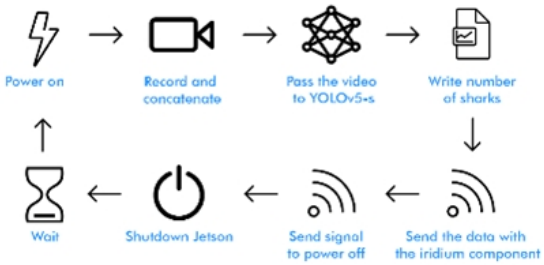


Figure 7: Complete system operation.

Figure 8: Upper view of the shark tank of the Planet Ocean aquarium in Montpellier, France.

Figure 9: Shark images in the aquarium (upper row) and in open water (bottom row). They are different in terms of color, shark or fish species, and background.

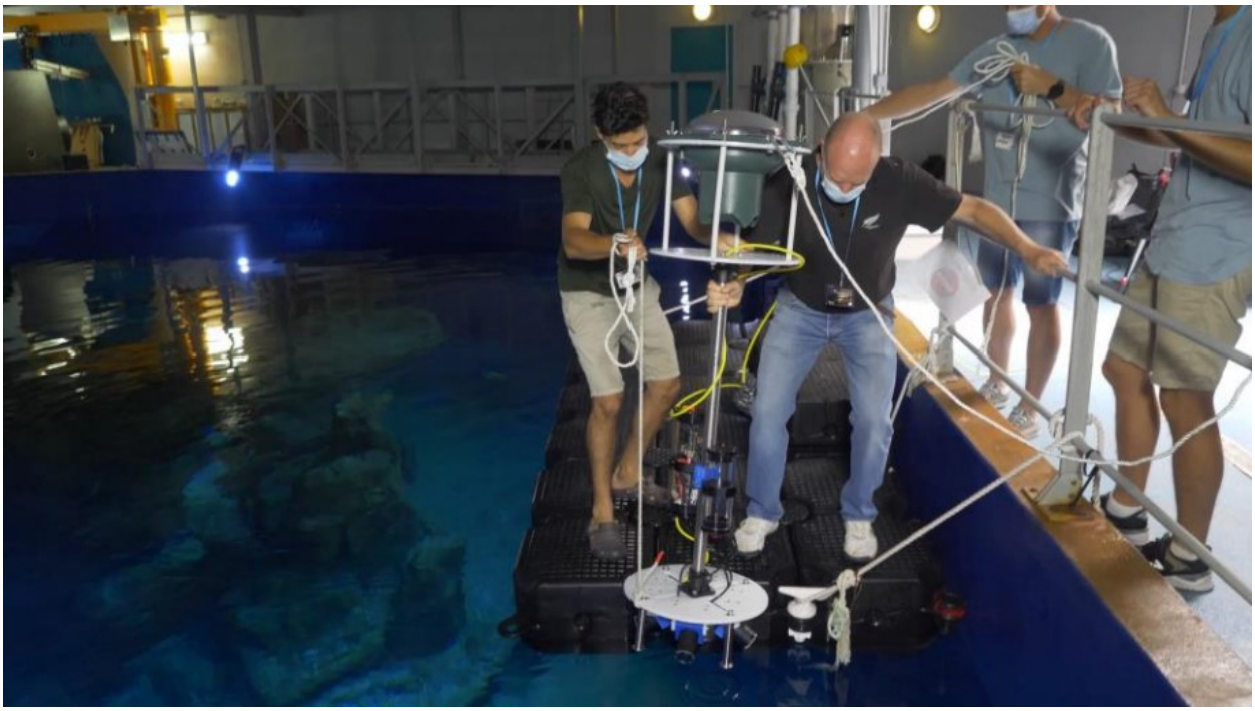


Figure 10: Installation of the system which was planned to be easily manipulated.

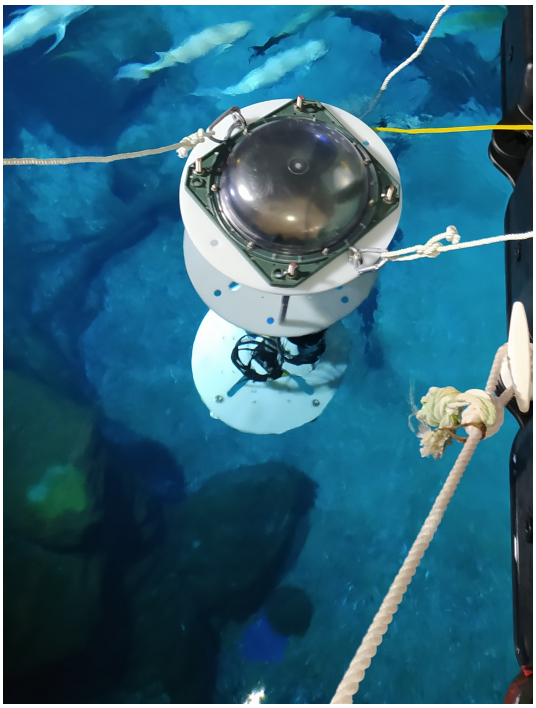


Figure 11: Upper view of the system in the aquarium.

Figure 12: Results of shark detection in the aquarium. The bounding box surrounds correctly a sawfish shark in the upper example but also.... the shadow of a sandbar shark which passes higher under a spot of light! Both are labelled as *Shark* and their detection score are displayed.