Journal of *Marine Science and Engineering*

MDPI

Article

Automatically Guided Selection of a Set of Underwater Calibration Images

Laurent Beaudoin [1,2,*], Loïca Avanthey [1,2,*], Corentin Bunel [1] and Charles Villard [1,2]

[1] SEAL (Sense, Explore, Analyse and Learn) Research Team, EPITA Computer Engineering School, 94270 Le Kremlin-Bicêtre, France; laurent.beaudoin@epita.fr (L.B.); loica.avanthey@epita.fr (L.A.); corentin.bunel@epita.fr (C.B.); charles.villard@epita.fr (C.V.)

[2] ACTE (Data Acquisition and Processing) Research Team, LASTIG Lab, IGN, 94160 Saint-Mandé, France

* Correspondence: laurent.beaudoin@epita.fr (L.B.); loica.avanthey@epita.fr (L.A.)

Abstract: The 3D reconstruction of underwater scenes from overlapping images requires modeling the sensor. While underwater self-calibration gives good results when coupled with multi-view algorithms, calibration or pre-calibration with a pattern is still necessary when scenes are weakly textured or if there are not enough points of view of the same points; however, detecting patterns on underwater images or obtaining a good distribution of these patterns on a dataset is not an easy task. Thus, we propose a methodology to guide the acquisition of a relevant underwater calibration dataset. This process is intended to provide feedback in near real-time to the operator to guide the acquisition and stop it when a sufficient number of relevant calibration images have been reached. To perform this, pattern detection must be optimized both in time and success rate. We propose three variations of optimized detection algorithms, each of which takes into account different hardware capabilities. We present the results obtained on a homemade database composed of 60,000 images taken both in pools and at sea.

Keywords: underwater blind calibration; pattern detection; realtime selection

Citation: Beaudoin, L.; Avanthey, L.; Bunel, C.; Villard, C. Automatically Guided Selection of a Set of Underwater Calibration Images. *J. Mar. Sci. Eng.* **2022**, *10*, 741. https://doi.org/10.3390/jmse10060741

Academic Editor: Weicheng Cui

Received: 29 April 2022
Accepted: 25 May 2022
Published: 27 May 2022

1. Introduction

Obtaining 3D point clouds from pairs of overlapping images of the seabed requires the use of a representative model of the sensor to reproject the pixels accordingly [1–3].

Most underwater camera housings use a flat port, which is cheaper and more reusable than hemispherical ports that must be specifically designed for a particular camera; however, these flat ports introduce significant distortions because of the phenomenon of refraction at their water–glass and glass–air interfaces [4–9].

Much work has been performed to address these issues. The methods closest to physical reality are those that use axial models, which take into account the multiple foci [10–12] but whose parameters remain complex to estimate. This is why in practice, the main strategies employed generally use single viewpoint (SVP) models incorporating corrections [13–21]. In fact, Ref. [22] shows that the estimation of the parameters of these compensated SVP models by multi-view methods (also known as self-calibration) makes it possible to obtain very satisfactory results.

However, self-calibration is challenged when the scenes are weakly textured, for example. The same issues arise when there are not enough different points of view on the same points within the acquired data, which can be the case with dynamics scenes [23]. In these cases, it is necessary to carry out a calibration or a pre-calibration of the model with a known pattern [24].

Regardless of the camera model used (which greatly depends on the type of optics used), the literature advises that a pattern should ideally contain at least 42 anchor points for better robustness to the noise generated by measurement errors; it has been shown that a minimum of about twenty different shots are needed to ensure a richer information context [25].

Refs. [26–28] showed that to obtain a good calibration result, it is necessary to avoid degenerate cases by introducing pitch, roll, and yaw variations and to be aware of having a spatial distribution of the pattern on the whole area of the sensor.

However, it is complicated and bulky on most small camera sensors to set up live video feedback of the sensor view in parallel with the acquisition of calibration photographs; therefore, these acquisitions are carried out "blindly". In order to statistically hope to obtain a set with good spatial distribution with pitch, roll, and yaw variation, a large number of images is usually taken.

Despite this, even after several hundred acquisitions, some information is often missing. Moreover, with such a quantity of calibration images at high resolution, the estimation of the model is time-consuming (several hours of calculation on a laptop computer). In addition, in these datasets, many images are useless (unrecognizable patterns, degenerate cases, etc.) and many others are redundant (almost identical positions). This can cause noise in the estimate and even cause it to fail.

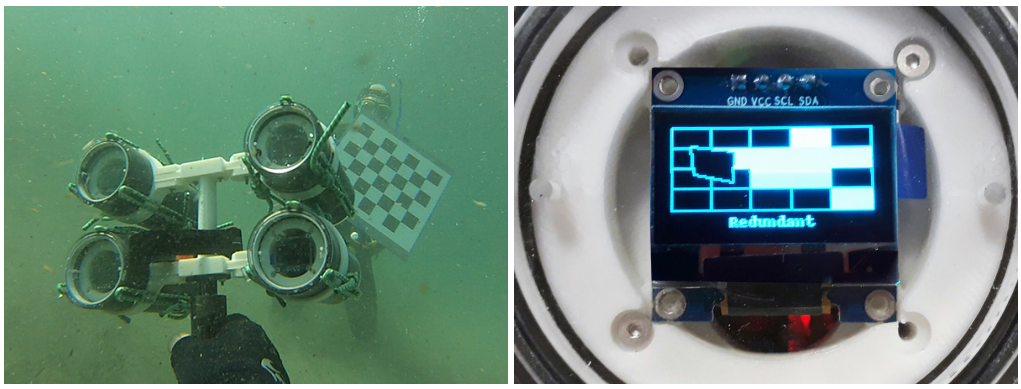
Therefore, it is important to ensure during the acquistion of the calibration set that all the necessary but sufficient information to proceed with the calibration is acquired. Thus, systematically guiding the acquisition makes it possible to optimize this task and automate it. The image acquisition methodology to form a calibration set presented in the following sections eliminates in near real-time all unnecessary information, guides the operator by indicating the missing information, and stops the acquisition process when there is sufficient relevant information.

Section 2 provides an overview of the proposed methodology. Section 3 details the optimization of the chessboard detection and Section 4 details the selection of relevant acquisitions.

In Section 5, we present the results of the evaluation of our methodology by using a homemade calibration database. This database contains approximately 60,000 images that we acquired during 10 years of field missions. Finally, Section 6 concludes and discusses the perspectives.

2. Overview of the Proposed Method: On-the-Fly Selection of a Relevant Set of Calibration Images

In our study, the calibration images are acquired either by a diver or by a robot, at nadir (the pattern is placed on the bottom) or sideways (the pattern is leaned on rocks or a wall or is held by a diver, see Figure 1, left). Acquisitions of underwater calibration images must be performed at the observation distance [5,24,29]. By thus fixing the GSD (Ground Sample Distance) according to the sensor used, this impacts the size of the tiles of the pattern, which must be large enough to allow their detection; however, a compromise is needed on the number of tiles used (and thus of the number of intersections) so as not to end up with chessboards that are too cumbersome. The chessboard patterns we use have between 30 and 72 anchor points (6×7 and 9×10 tiles).



(a) **(b)**

Figure 1. In situ acquisition by divers (**a**) and feedback on the localization of the validated data (plain tiles) on a embedded small screen (**b**).

With each new image acquired by the system, our embedded selection program proceeds to the detection of the chessboard. If it finds it, it analyzes its position and its orientation to decide if it is a new necessary relevant information to record. A small screen mounted on the back of the acquisition system indicates to the operator the areas where information is missing (see Figure 1, right). The flowchart of this algorithm is shown in Figure 2.

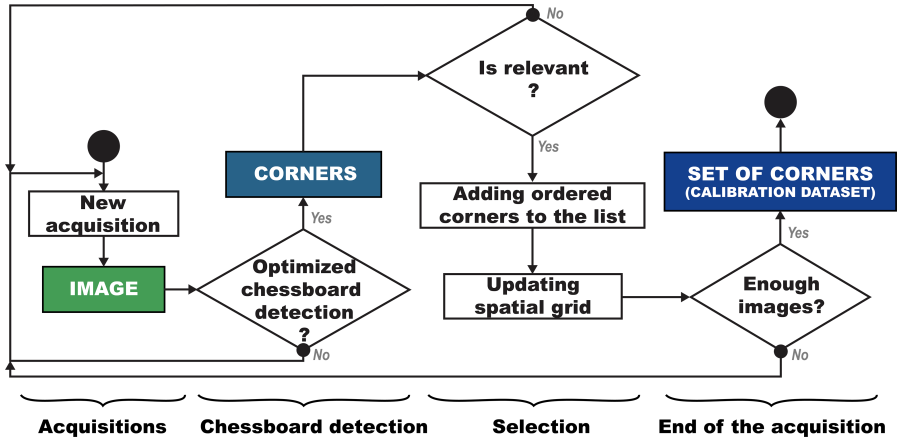


Figure 2. Flowchart of the algorithm that automatically selects a set of calibration images during an acquisition.

From an operational point of view, we seek to shorten the time spent on the acquisition as much as possible. For this, it is necessary on the one hand that the detection can be performed in near real-time on images that are taken at full resolution and on the other hand to maximize the detection rate.

The proposed algorithm has been implemented in the OpenCV 4.0 library [30]. This library dedicated to embedded image processing offers in its fourth version a good abstraction for optimizations such as parallelization. When building the project, it automatically detects the possible optimizations according to the available hardware and enable them. Here, for example, we refer to SSE (Streaming SIMD Extensions) or AVX (Advanced Vector Extensions) on CPUs but also to CUDA (NVIDIA) or OpenCL on GPUs.

3. Chessboard Detection Optimization

The first step of the proposed algorithm is to filter the input images to only keep those on which all the pattern anchor points have been detected, located, and ordered. This task is often the most expensive one out of the whole process as it is directly related to the number of pixels in the images.

In this section, we detail how we optimize this detection step by improving its performances, both in processing time and in success rate. We propose three pipeline variations for this step, each with advantages depending on the capabilities of the hardware it is running on, as shown in Section 5. The flowchart of this step with the three pipeline variations is illustrated in Figure 3.

3.1. Overview of the Three Corners Detection and Localization Pipelines

To reduce computation time, all pipeline variations are preceded by a step of downsampling the incoming images. We have shown in [31] that the best compromise between computation time and detection rate is obtained when the width of a tile of the pattern measures approximately 10 pixels. This saves at least 80% computation time while maintaining a detection rate of at least 80%.

The first pipeline variation (called full OpenCV in the rest of the article) is based on a classic image processing algorithm (*findChessboardCorners* algorithm [32]). The efficiency of this algorithm in computation time has been greatly improved in the latest versions of OpenCV.

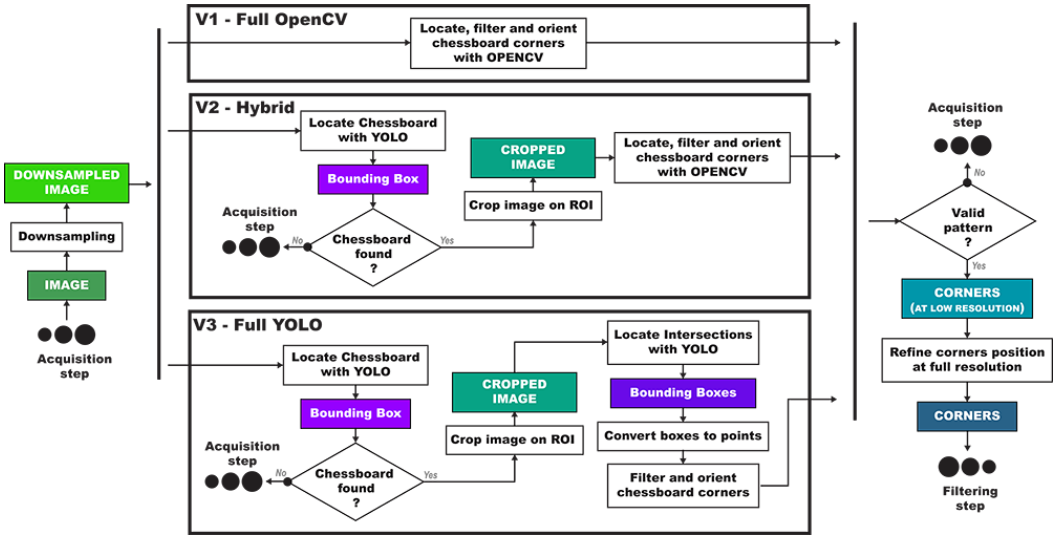


Figure 3. Detailed flowchart of the optimized chessboard detection step in its three variations: full OpenCV (V1), hybrid (V2), and full YOLO (V3).

The second pipeline variation (called hybrid) introduces an additionnal AI step to the previous pipeline. This step first focuses the attention on the part of the image where the chessboard is located by a deep learning algorithm (see Section 3.3). If found, then it continues with a classic search for anchor points with the *findChessboardCorners* algorithm on the selected region of interest (ROI). This strategy has two advantages: on the one hand, the drastic reduction in the size of the search area saves a lot of computation time. On the other hand, removing the background elements makes it possible to avoid false tracks and thus improves detection.

Finally, the third pipeline variation (called full YOLO) replaces in the previous variation the step of locating the position of the anchor points on the isolated region of interest by another deep learning algorithm (see Section 3.4). This is mainly intended to improve the detection rate.

At the end of each of these pipeline variations, the list of positions of the anchor points on the downsampled image is obtained (if the chessboard was found, otherwise the pipeline goes to the acquisition of the next input image). There are several points to consider at this stage.

On the one hand, we showed in [31] that the calibration must be performed on the images at the resolution of use (called full resolution in the paper). The coordinates of the anchor points are therefore converted to full resolution thanks to the sampling factor; however, this conversion causes positioning errors of a few pixels compared to detection carried out directly on full resolution images. On the other hand, in the full YOLO variation, the detection of the location of the anchor points by the neural network in the subsampled images may also present inaccuracies, here again of a few pixels at the most, coming from the data labeling or from the detection itself. These inaccuracies are cumulative with those of the full resolution conversion but remain of the order of a few pixels.

We showed in [31] that positioning errors of a few pixels can be corrected by using the refinement algorithm normally used to find the subpixel position of the corners if the analysis window is not too tight. This strategy allows the use of downsampling in our algorithm, as well as the manual labeling of intersections and their detection by a neural network, without loss of quality in the final results.

3.2. Presentation of the Deep Neural Network Used for the Hybrid and Full YOLO Variations of the Chessboard Detection

Deep learning algorithms are machine learning algorithms that use a large number of layers in their networks in order to extract increasingly higher-level features from the raw inputs [33,34]. Most of them are based on convolutional neural networks (CNNs).

For the detection of objects on images, there are mainly two categories of methods [35]: region-proposal-based methods [36–41] and regression-based methods [42–47]. Recent work uses these deep-learning algorithms of one or the other category on underwater images in order to circumvent image quality problems [48–58]. Their results, compared to those obtained by classical image processing, are very promising.

The YOLO algorithm [46] is a regression-based algorithm that reasons globally on the image. Unlike techniques based on region proposal and sliding windows (Region-based Convolutional Neural Networks—R-CNN [59]) the image is seen only once (YOLO is for "You Only Look Once") during training or to predict which objects are visible and where they are. YOLO therefore models object detection as a single regression problem. It divides the input image into a uniform grid and simultaneously predicts for each cell via a convolutional network the bounding boxes for objects whose centers fall in the cell, the confidence in these boxes and the class probabilities. Those results are provided in the form of tensors.

The network implicitly encodes the contextual information about the classes. After that, a non-maximum suppression algorithm (NMS) makes it possible to remove potential duplicates that overlap between delimiting areas. This ability to consider the whole image as an informative context and to learn generalizable representation of objects makes YOLO a particularly fast and robust algorithm in our application case.

Concretely in this work, we use version 4 of the YOLO algorithm [60]. This is a version designed to optimize parallel computation and to improve object detection. The neural network backbone which extracts the characteristics is based on CSPDarknet53 [61]. It is a CSPNet (Cross Stage Partial Network) based on a modified version of DenseNet [62], which is pre-trained on ImageNet [63].

The neck of the neural network which allows to merge the characteristics extracted by the backbone is composed of two networks, SPP [38] (Spatial Pyramid Pooling) and PANet [64] (Path Aggregation Network). The former increases the receptive field and helps separate contextual features, while the latter plays a role in parameter convergence by shortening the paths connecting low-level and high-level information.

Finally, the head of the neural network which predicts the bounding boxes and the classification of objects is the same as for YOLOv3 [65]. It is possible to create smaller or larger networks depending on the need (speed versus precision) and a large number of network parameters are configurable thanks to a configuration file.

3.3. Using AI to Focus the Attention around the Pattern Location

In the hybrid and full YOLO variations, we use the YOLO algorithm to focus the attention of the algorithm on the part of the image containing the chessboard. This avoids searching for anchor points over the entire area of the image.

We have trained our YOLO network with the Darknet framework. To perform this, we have manually labeled a small part of our dataset (<1000 images). We excluded 10% of these labeled images to be used for our test dataset. We have modified the framework to adapt to the detection of a single type of object (in this case the chessboard) while the original framework works on 80 types of objects. This allows us to lighten the network, speed up calculations and improve the results by avoiding false assignment. This change mainly impacts the number of filters on the last layers of the network (less numerous), the number of batches (also less numerous) and the steps which adjust the learning rate during the gradient descent. We have kept all the hyper-parameters defined by default in the frameworks. They have been highly optimized for general object detection via genetic algorithms [60]. The highest size (608 × 608 px) of the input images used for training gives the best learning results.

With these parameters we obtain a mean average precision (mAP) greater than 98% for this global chessboards detection step. This is not surprising as chessboards are very characteristic and standardized objects, especially in an underwater environment.

To run our model, we have chosen to use the DNN module (Deep Neural Networks) of the OpenCV library. This module is dedicated to machine learning in the field of images. It allows us to load a pre-trained model and run it on images to obtain the desired classifications. The fact that the OpenCV library reimplements popular neural network frameworks (such as TensorFlow, PyTorch, etc.) without integrating the learning phase (backpropagation) makes it possible to obtain a final framework that is more compact and therefore much lighter. It is a considerable advantage when looking to deploy it on embedded devices, such as in our case.

YOLO is practically independent of the number of pixels in the analyzed images because it uses a fixed size for its input images. When we run the network, we can choose a smaller input size among a proposed set. As the number of neurons loaded in the first layer is usually directly related to the number of pixels in the input image, selecting a smaller size consequently deactivate entire parts of the network.

Thus, the fewer the number of neurons activated in the first layer are, the faster the prediction is; however, it is also less precise. As the task of our model is simple, we set it to the smallest size allowed by the network (320 $\times$ 320 px).

In our workflow (see Figure 3 for a reminder), either the network does not detect a pattern and the pipeline then goes through the acquisition of the next image, either it detects a pattern (see Figure 4 for examples) and then focuses on the Region Of Interest (ROI) detected (corresponding to the bounding box, see Figure 5) to search for the anchor points of the chessboard.

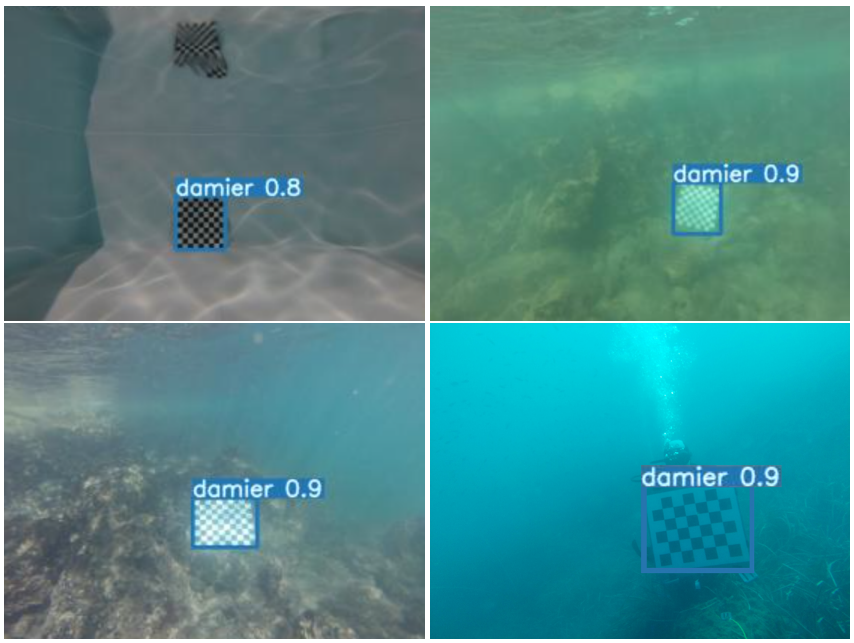


Figure 4. Four examples of bounding boxes detected by YOLO that represent chessboards in underwater images.

3.4. Using AI to Fully Detect the Chessboard Intersections

In the full YOLO variation, the search for anchor points is carried out by a second image analysis network. So, we trained a second YOLO network with the darknet framework to detect the intersections. To perform this, we have manually labeled about 10,000 intersections on a part of the cropped images focused on the chessboards. In total, 90% of these labeled data were used for training and 10% was kept for the test dataset. This second network works directly on the cropped images of chessboards. In this way, the problem is simplified and practically made independent of the type background. For this second model, we obtain a mAP greater than 98.5%.

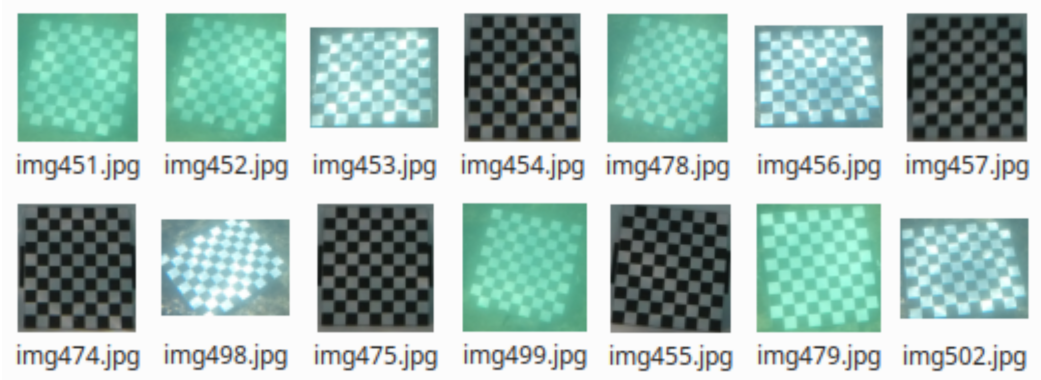


Figure 5. Focus of attention: the bounding boxes are used to automatically crop the images on the region of interest (the chessboard) to simplify the search for the intersections (anchor points).

This task is a bit more complex than the detection of the chessboard. Thus, to obtain a good prediction of the position of the intersections, we switch to one of the highest sizes allowed by the model (576 × 576 px).

After filtering the output of the neural network, we convert the detected bounding boxes (see Figure 6) into points. We eliminate any outliers by sorting the points and checking if the original searched pattern is complete. Ordering the intersections of the chessboard is an essential step to be able to associate them during the calibration. This process use an adjacency matrix and exploits an *a priori* knowledge of the pattern.

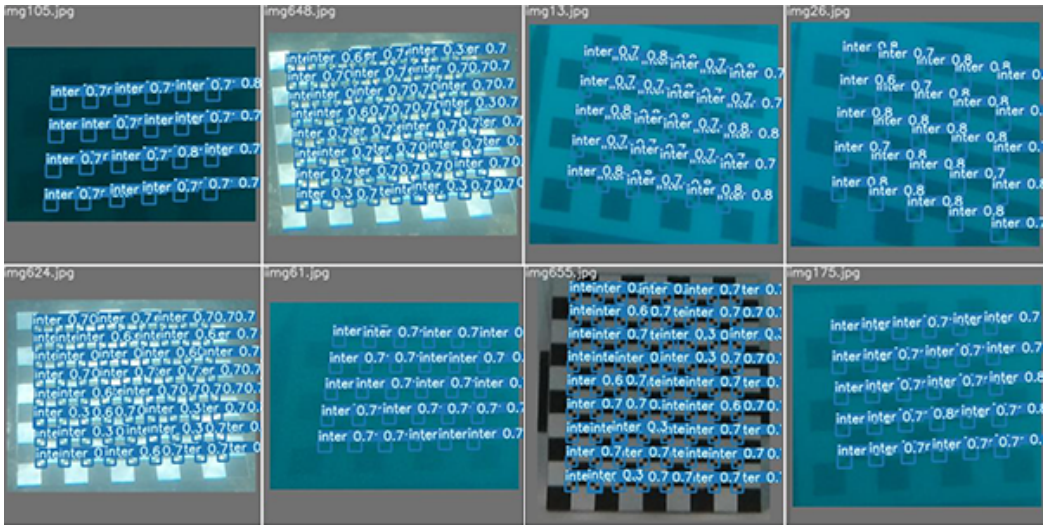


Figure 6. Examples of bounding boxes detected by YOLO that represent the intersections of chessboards in cropped underwater images.

If the pattern is not identified (too many or not enough points or points incorrectly ordered), the chessboard is considered as not detected (partial detection) and the pipeline goes through the acquisition of the next image. Otherwise, if the chessboard is validated (see Figure 7 for examples of results), then the pipeline goes through the selection step to find out if the image is relevant within the calibration set.

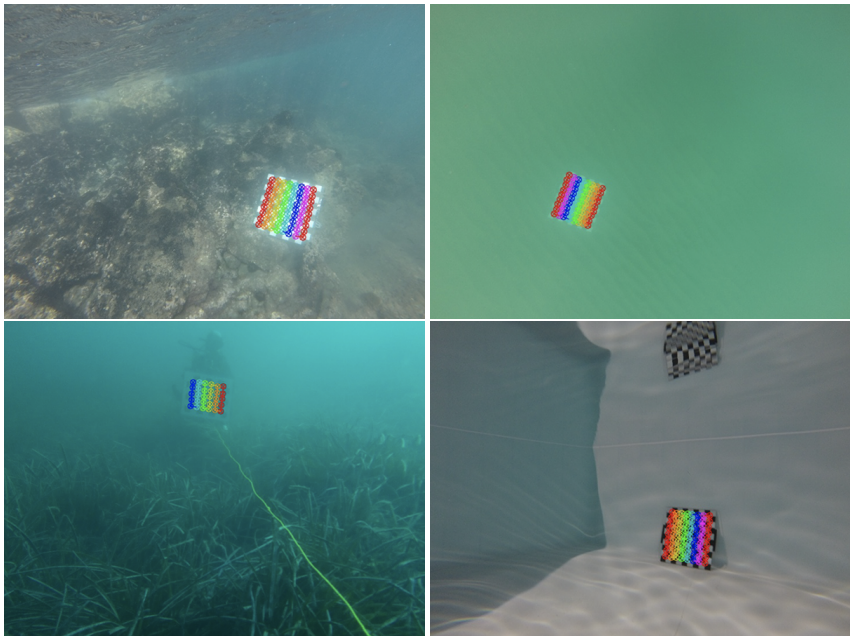


Figure 7. Final part of the chessboard detection: four examples of identified patterns whose detected intersections have been correctly ordered.

4. Automatic Selection of a Set of Relevant Images

If a pattern was detected during the previous phase, the list of the positions of the anchor points of the pattern in the subsampled image (pixel coordinates) is obtained. This information is used to keep only a reduced set of relevant images for the estimation of the parameters of the model. Indeed, as seen in Section 1, the quality of the estimation mainly relies on a good spatial distribution of the anchor points position over the sensor and a great variability of their three-dimensional global orientation. The flowchart of this selection step is summarized in Figure 8.

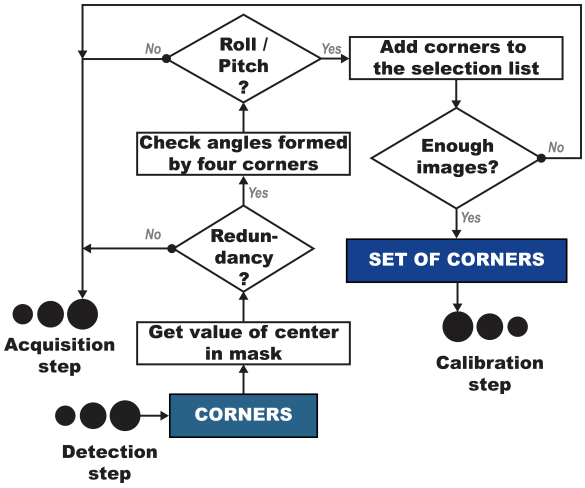


Figure 8. Detailed flowchart of the selection step.

Concerning the spatial distribution, we ensure it by the use of a mask previously segmented into areas. The number of areas is chosen according to the optics and the observation distance. In practice, we saw in Section 1 that a minimum of 10 to 20 areas are needed to ensure a good quality of the results. The coordinates of the center of the pattern indicates the area to which it belongs. The pattern is rejected if this area already contains information (redundancy). If the pattern passes this test, an orientation test starts.

To save computation time, this step can be performed just after the focus of attention in the hybrid variation or in the full YOLO variation. Indeed, the center of the ROI resulting of the focus corresponds approximately to the center of the chessboard. Then, checking if the information is already present at this place in the mask at this stage makes it possible to avoid the step of searching for the anchor points.

The second test ensures the diversity of three-dimensional orientation: it checks that the pattern presents non-zero roll or pitch movements. To perform that, we verify that the footprint of the quadrilateral formed by the four corners of the pattern corresponds to a trapezoid shape and not to a shape too close to a parallelogram. If not, the pattern is rejected as a possible degenerate case and the pipeline goes through the next acquisition.

If the pattern passes this second test, it means it is relevant for calibration. The area it occupies is marked as covered in the mask. Then, its pixel coordinates are converted and refined to full resolution (see Section 3.1) and saved in the list of selected patterns.

Finally, the last test checks if the desired number of patterns have been reached. If this is not the case, the missing areas are indicated to the operator and the pipeline goes through the next acquisition. Otherwise, the operator is notified that the acquisition set is complete. At the end of the whole process, the lists of the selected images and the pixel coordinates refined at the full resolution of all associated image–model correspondences are available.

5. Results

For the assessment of our results, we use a database we created and which contains about 60,000 images corresponding to calibration sequences acquired by our different cameras in various environmental conditions: Mediterranean Sea, swimming pool, outdoor pool, at different times and seasons, from poor to clear visibility. These calibration sequences are taken at observation distances between 1 and 5 m. In addition to offering a wide variety of pattern positions, pattern lighting, and backgrounds, they include many images where the patterns are missing, cut off by the edges, or obstructed, for example. We use calibration targets of two different sizes. One measures 45 × 50 cm and is composed of 9 × 10 tiles (72 intersections) and the other measures 36 × 42 cm and is composed of 6 × 7 tiles (30 intersections). For underwater use, we replaced the white frame with a gray frame in order to not saturate the sensor and to be more robust in non-homogeneous lighting.

5.1. Overall Detection Rate Increased by AI

The different variations of the chessboard detection step have been tested on our database. The results are presented in Table 1.

Table 1. Detection rate obtained over the calibration images in our database by the three pipeline variations. The gain is compared to the Full OpenCV variation.

	Full OpenCV Variation	Hybrid Variation (YOLO + OpenCV)	Full YOLO Variation
Detection rate	24%	35%	76%
Gain		+49%	+220%

The full OpenCV variation only detects 24% of the detectable chessboards. With the hybrid variation, the overall detection rate increases to 35%. It corresponds to an increase of almost 50% compared to the results of the full OpenCV variation. In addition, with the full YOLO variation, we obtain an overall detection rate of 76%. It corresponds to a gain of around 220% compared to the full OpenCV variation and around 115% compared to the hybrid variation.

The database that we have established is representative of the great variability of the experimental conditions that we have encountered. This diversity is found in the results of detection of the calibration pattern obtained by the various algorithms. Thus, the OpenCV algorithm can have very low detection rates (less than 5%) when visibility is poor,

the lighting is inhomogeneous, or in the presence of caustics. The gain provided by the integration of the AI in the process is especially significant in datasets where visibility is poorer (lack of light, turbidity, slightly higher viewing distance). On these particular and very unfavorable datasets, the hybrid variation still manages to maintain a minimum detection rate of around 30%. For the full YOLO variation, this minimum rate is 50%.

5.2. Efficiency of Chessboard Detection Variations according to Hardware Capacities

The other important performance criterion from an operational point of view is the computing time. To compare the three variations (full OpenCV, hybrid, full YOLO) on this criterion, we decided to run them on three types of hardware architecture configurations representative of what is available in operation: simple computation on CPU, parallelized computation on CPU, and parallelized computation on GPU. The results obtained are shown in Table 2 (the GPU used to produce these results is a Nvidia GTX 1650 with a i7 3770 CPU).

Table 2. Average time in milliseconds to perform detection on an image according to the different hardware configurations and the three pipeline variations.

	Simple Computation on CPU	Parallelized Computation on CPU	Parallelized Computation on GPU
Full OpenCV	∼345 ms	∼345 ms	∼345 ms
Hybrid	∼2060 ms	∼300 ms	∼35 ms
Full YOLO	∼8750 ms	∼1200 ms	∼90 ms

We observe that the execution time of the full OpenCV variation, which is already highly optimized, remains constant whatever the configuration used. Indeed, the code does not seem to have been designed to take advantage of an advanced parallelization.

On the other hand, deep learning algorithms are very resource intensive (6 to 25 times slower than the full OpenCV variation on the simple CPU configuration); however, the YOLO v4 algorithm has been specifically conceived to exploit parallelization, particularly on GPUs, in order to present performance compatible with real-time use. This is shown with the first gain in time with the parallelization computation on CPU: 10 times faster for the hybrid variation, 7 times faster for the full YOLO variation. In this configuration, the hybrid variation takes the advantage over the full OpenCV variation.

By switching to GPU parallelization, the gain in time is, as expected, even more important: 60 times faster for the hybrid variation and 100 times faster for the full YOLO variation. In this configuration, these two variations greatly outperformed the full OpenCV variation.

We observe faster results with the hybrid variation when using parallel optimizations. This is due first to the fact that we use only one neural network rather than two as in the full YOLO variation. Furthemore, the *findChessboardCorners* algorithm is in this case used on a very small ROI, which drastically reduces its execution time compared to when it is used on the entire image, as in the full OpenCV variation.

These results help to guide the choice of the pipeline variation to use depending on the embedded hardware selected for the acquisition. If the hardware solution is only CPU-based, a detection by the full OpenCV variation would be preferred. In addition, in case of failure, a second try can be performed with the full YOLO variation.

On the other hand, if the performance of the CPU is high, it will be interesting to favor the hybrid variation on the first try before supplementing with the full YOLO variation if needed. In addition, if the hardware solution has a GPU, then the hybrid variation as a first try would be preferred if the GPU has average performance; however, if the GPU is efficient, then the full YOLO variation can be directly used.

To push further in the optimization of the calculation time, it would be interesting to detect intersections directly by a single neural network rather than by two successive ones; however, further work is necessary to tackle the high number of false positives given by YOLO in this case.

5.3. Automatic Selection of a Set of Relevant Images

We have tested the automatic selection step on the calibration datasets present in our database. The objective of the selection step is to retain only about twenty to forty images when processing datasets that can contain hundreds to thousands of images. The images are evaluated by the selection step one after another, after they have passed the detection step.

In terms of statistics, if we analyze an entire typical calibration sequence, the automatic selection algorithm classifies about 40% of the detected pattern as too close to degenerate cases (not enough roll or pitch). The vast majority of the remaining patterns are classified as redundant. From a qualitative point of view, these statistics show the difficulty of blindly acquiring a set of images reaching the two quality criteria corresponding to a good spatial distribution and a great variability in the three-dimensional orientations of their patterns.

Figure 9 shows examples of patterns detected without selection during unguided blind acquisitions and examples of detected pattern selections during guided acquisitions. We can note that even when using large datasets, not all of the desired information is retrieved via unguided acquisitions. This shows the importance of being able to use the selection program embedded in near real-time to guide the operator to obtain the desired data in a minimum time.

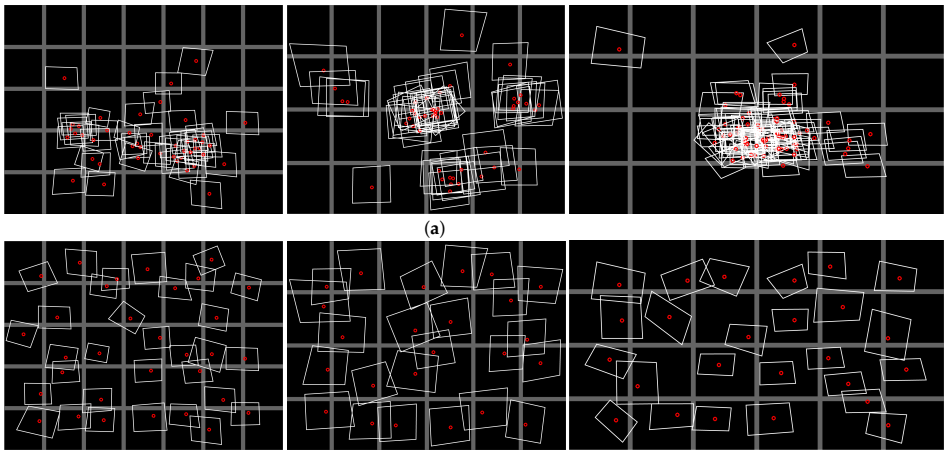


Figure 9. Six examples of localization of the fingerprints (white frames) of the patterns detected in a set of calibration data on the grid of the sensor zones (gray grid): (**a**) not guided blind acquisition, (**b**) guided blind acquisition with selection.

In practice, using the guidance, the acquisition is performed in less than 50 to 120 shots to obtain all the images of a complete set of about 25 to 60 well-distributed and non-degenerate patterns (the patterns on the edges are the hardest to obtain).

Table 3 shows the results obtained on the estimation of the intrinsic parameters from calibration sets with selection and calibration sets without selection for a GoPro H3+ camera using a Fisheye model. The parameters concerned are the focal length (f), the position of the principal point (c_u, c_v), the coefficients of radial distortion (k_1, k_2, k_3), and those of tangential distortion (p_1, p_2). In this experiment, the results obtained with the selection are completely comparable to those obtained without selection and with overall lower standard deviations.

We then used these calibration results to reconstruct in 3D an experimentation site on which we were able to make in situ distance measurements on control points (following the methodology described in [66]). To take into account the fact that the greater the distances measured, the more they are tainted with errors (mainly because of the tension that must be applied to the measurement cable), it is customary to express the errors in percentage of total measured length. In our case, we obtain an average error of 4% if we use the guided calibration and of 3% for the unguided one. These results deserve to be reinforced by other experiments. Nevertheless, the guided calibration does not seem to significantly affect the quality of the 3D reconstruction.

Table 3. Means obtained for each intrinsics coefficient of a Fisheye model and the corresponding standard deviation (σ) on calibration sets without and with selection, for a GoPro H3+ camera.

		$\mathbf{f}$	c_u	c_v	k_1	k_2	k_3	p_1	p_2
Without selection	**value**	2.37	0.029	-0.016	-0.364	0.113	$-0{,}032$	1.5×10^{-3}	-1.1×10^{-4}
	σ	0.1	9.0×10^{-3}	8.0×10^{-3}	4.0×10^{-3}	9.0×10^{-4}	1.0×10^{-3}	2.0×10^{-4}	9.0×10^{-6}
With selection	**value**	2.349	-0.028	-0.018	-0.321	0.89	0.027	2.0×10^{-3}	1.0×10^{-5}
	σ	0.06	5.0×10^{-3}	7.0×10^{-3}	2×10^{-3}	1.2×10^{-3}	1.4×10^{-3}	1.0×10^{-4}	1.1×10^{-5}

6. Conclusions

We have proposed a methodology intended to guide the underwater acquisition of calibration images. Its objective is to create a relevant minimalist image dataset to proceed with the calibration of the camera model.

To process the image on-the-fly, our algorithms optimizes the detection of calibration patterns both in time and quality. We provided three pipeline variations that can be used depending on the capabilities of the chosen hardware (simple computation on CPU, parallelized computation on CPU, and parallelized computation on GPU).

The first variation (full OpenCV) is based on a classical algorithm and is suitable for simple computation on CPU. The second variation (hybrid) focuses first the attention of the classical algorithm on an ROI founded by a deep neural network.

The results computed over a homemade database of over 60,000 calibration images that we have built from our many acquisition campaigns show that this variation allows us to increase the detection rate by almost 50% and that it takes advantage of CPU and GPU parallelizations to improve computation time (time almost equivalent to the full OpenCV variation with CPU parallelization and almost 10 times faster with GPU parallelization); however, without these optimizations, it is six times slower than the full OpenCV variation.

Finally, the third variation (full YOLO) replaces the classical algorithm by another deep neural network after the focus of attention step. The results show that the detection rate of this variation was improved by 220% compared to the full OpenCV variation (115% compared to the hybrid variation) and that the computation time was almost four times faster than that of the full OpenCV variation (but three times slower with only CPU parallelization and 25 times slower without any parallelization).

These three variations are based on the first step of robust downsampling, which alone saves 80% of calculation time for a loss in detection rate of less than 20%. They are then followed by a selection step which checks whether each image being processed provides new information (spatial or orientation). The feedback is sent back to the operator to guide the acquisition of missing information. The results show that the calibration sets obtained by this selective guidance are more complete and more compact than those obtained completely blind.

The calibration tests carried out on different data sets, guided or not, have made it possible to validate the main hypothesis of this article, i.e., that it is possible to guide the acquisition to quickly acquire a minimum set of images of calibration without a significant loss of quality in the estimation of the calibration model and in the final reconstruction.

The proposed method can be generalized to all acquisition devices allowing embedded image processing to implement the algorithm. It can also be adapted to multi-camera systems provided that the housings and devices used allow the synchronization of acquisitions in addition to the screen connections.

Author Contributions: L.A. and L.B. conceptualized and designed the methodology; L.A., L.B., C.B., and C.V. conceived, designed and performed the experiments; L.A. and L.B. analyzed the data; L.A. and L.B. wrote the paper. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Contact the corresponding authors for the data.

Acknowledgments: The authors would like to thank Apolline Wasik and Kévin Guillet, engineering students of SEAL research Team, EPITA, France, who participated in the development of the algorithm, in the configuration and training of the network.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Faugeras, O. *Three-Dimensional Computer Vision—A Geometric Viewpoint*; MIT Press: Cambridge, MA, USA , 1993.
2. Hartley, R.; Zisserman, A. *Multiple View Geometry in Computer Vision*; Cambridge University Press: Cambridge, UK , 2003.
3. Corke, P. *Robotics, Vision and Control: Fundamental Algorithms in MATLAB*; Springer: Berlin/Heidelberg, Germany, 2011; Volume 73.
4. Jasiobedzki, P.; Se, S.; Bondy, M.; Jakola, R. Underwater 3D Mapping and Pose Estimation for ROV Operations. In Proceedings of the IEEE OCEANS Conference, Quebec City, QC, Canada, 15–18 September 2008; pp. 1–6. [CrossRef]
5. Treibitz, T.; Schechner, Y.Y.; Kunz, C.; Singh, H. Flat Refractive Geometry. *IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI)* **2012**, *34*, 51–65. [CrossRef] [PubMed]
6. Kawahara, R.; Nobuhara, S.; Matsuyama, T. A Pixel-Wise Varifocal Camera Model for Efficient Forward Projection and Linear Extrinsic Calibration of Underwater Cameras with Flat Housings. In Proceedings of the IEEE International Conference on Computer Vision Workshops (ICCVW), Sydney, NSW, Australia, 2–8 December 2013, pp. 819–824.
7. Agrafiotis, P.; Georgopoulos, A. Camera Constant in the Case of two Media Photogrammetry. *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.* **2015**, *XL-5/W5*, 1–6. [CrossRef]
8. Menna, F.; Nocerino, E.; Fassi, F.; Remondino, F. Geometric and Optic Characterization of a Hemispherical Dome Port for Underwater Photogrammetry. *Sensors* **2016**, *16*, 48. [CrossRef] [PubMed]
9. Menna, F.; Nocerino, E.; Remondino, F. Optical aberrations in underwater photogrammetry with flat and hemispherical dome ports. In *Videometrics, Range Imaging, and Applications XIV*; Remondino, F., Shortis, M.R., Eds.; International Society for Optics and Photonics (SPIE): Bellingham, WA, USA , 2017; Volume 10332, pp. 28–41. [CrossRef]
10. Agrawal, A.; Ramalingam, S.; Taguchi, Y.; Chari, V. A Theory of Multi-Layer Flat Refractive Geometry. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Providence, RI, USA, 16–21 June 2012; pp. 3346–3353.
11. Jordt-Sedlazeck, A.; Koch, R. Refractive Structure-from-Motion on Underwater Images. In Proceedings of the 2013 IEEE International Conference on Computer Vision, Sydney, NSW, Australia, 1–8 December 2013; pp. 57–64. [CrossRef]
12. Jordt, A.; Köser, K.; Koch, R. Refractive 3D reconstruction on underwater images. *Methods Oceanogr.* **2016**, *15–16*, 90–113. [CrossRef]
13. Gracias, N.; Santos-Victor, J. Underwater Mosaicing and Trajectory Reconstruction Using Global Alignment. In Proceedings of the IEEE OCEANS Conference, Honolulu, HI, USA, 5–8 November 2001; Volume 4, pp. 2557–2563.
14. Pessel, N.; Opderbecke, J.; Aldon, M.J. Camera Self-Calibration in Underwater Environment. In Proceedings of the International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision (WSCG), Plzen-Bory, Czech Republic, 3–7 February 2003; pp. 104–110.
15. Brandou, V.; Allais, A.G.; Perrier, M.; Malis, E.; Rives, P.; Sarrazin, J.; Sarradin, P.M. 3D Reconstruction of Natural Underwater Scenes Using the Stereovision System IRIS. In Proceedings of the IEEE OCEANS Conference, Marseille, France, 29 September–4 October 2007; pp. 674–679.
16. Drap, P.; Seinturier, J.; Scaradozzi, D.; Gambogi, P.; Long, L.; Gauch, F. Photogrammetry for Virtual Exploration of Underwater Archeological Sites. In Proceedings of the CIPA International Symposium, Athens, Greece, 1–6 October 2007.
17. Beall, C.; Lawrence, B.J.; Ila, V.; Dellaert, F. 3D Reconstruction of Underwater Structures. In Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS), Taipei, Taiwan, 18–22 October 2010; pp. 4418–4423.
18. Kunz, C.; Singh, H. Stereo Self-Calibration for Seafloor Mapping Using AUVs. In Proceedings of the IEEE Autonomous Underwater Vehicles (AUV), Monterey, CA, USA, 1–3 September 2010; pp. 1–7.
19. Drap, P. Underwater Photogrammetry for Archaeology. In *Special Applications of Photogrammetry*; InTechOpen: Rijeka, Croatia , 2012; pp. 111–136.
20. Skarlatos, D.; Demestiha, S.; Kiparissi, S. An 'open' method for 3D modelling and mapping in underwater archaeological sites. *Int. J. Herit. Digit. Era* **2012**, *1*, 1–24. [CrossRef]
21. Avanthey, L.; Beaudoin, L.; Gademer, A.; Roux, M. Tools to Perform Local Dense 3D Reconstruction of Shallow Water Seabed. *Sensors* **2016**, *16*, 712. [CrossRef]
22. Kang, L.; Wu, L.; Yang, Y.H. Experimental study of the influence of refraction on underwater three-dimensional reconstruction using the SVP camera model. *Appl. Opt.* **2012**, *51*, 7591–7603. [CrossRef]
23. Rende, S.F.; Bosman, A.; Menna, F.; Lagudi, A.; Bruno, F.; Severino, U.; Montefalcone, M.; Irving, A.D.; Raimondi, V.; Calvo, S.; et al. Assessing Seagrass Restoration Actions through a Micro-Bathymetry Survey Approach (Italy, Mediterranean Sea). *Water* **2022**, *14*, 1285. [CrossRef]

24. Luhmann, T.; Fraser, C.; Maas, H.G. Sensor modelling and camera calibration for close-range photogrammetry. *ISPRS J. Photogramm. Remote Sens.* **2016**, *115*, 37–46. [CrossRef]
25. Bradski, G.; Kaehler, A. *Learning OpenCV: Computer Vision with the OpenCV Library*; O'Reilly: Newton, MA, USA , 2008.
26. Zhang, Z. A Flexible New Technique for Camera Calibration. *IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI)* **2000**, *22*, 1330–1334. [CrossRef]
27. Scaramuzza, D.; Martinelli, A.; Siegwart, R. A Flexible Technique for Accurate Omnidirectional Camera Calibration and Structure from Motion. In Proceedings of the IEEE International Conference on Computer Vision Systems (ICVS), New York, NY, USA, 4–7 January 2006; p. 45. [CrossRef]
28. Servos, J.; Smart, M.; Waslander, S.L. Underwater Stereo SLAM with Refraction Correction. In Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS), Tokyo, Japan, 3–7 November 2013; pp. 3350–3355.
29. Jordt-Sedlazeck, A.; Koch, R. Refractive Calibration of Underwater Cameras. In Proceedings of the European Conference on Computer Vision (ECCV), Florence, Italy, 7–13 October 2012; Volume 5, pp. 846–859.
30. OpenCV (Open Computer Vision Library), Intel, USA . Official Website. Available online: https://opencv.org/opencv-4-0/ (accessed on 24 May 2022).
31. Avanthey, L.; Beaudoin, L. Underwater Calibration in Near Real Time: Focus on Detection Optimized by AI and Selection of Calibration Patterns. In Proceedings of the IGARSS 2020—2020 IEEE International Geoscience and Remote Sensing Symposium, Waikoloa, HI, USA, 26 September–2 October 2020; pp. 1576–1579. [CrossRef]
32. Shilkrot, R.; Escrivá, D.M. *Mastering OpenCV 4: A Comprehensive Guide to Building Computer Vision and Image Processing Applications with C++*, 3rd ed.; Packt Publishing: Birmingham, UK , 2018.
33. Deng, L.; Yu, D. Deep Learning: Methods and Applications. *Found. Trends Signal Process.* **2014**, *7*, 197–387. [CrossRef]
34. LeCun, Y.; Bengio, Y.; Hinton, G. Deep Learning. *Nature* **2015**, *521*, 436–444. [CrossRef] [PubMed]
35. Zhao, Z.Q.; Zheng, P.; Xu, S.t.; Wu, X. Object Detection With Deep Learning: A Review. *IEEE Trans. Neural Networks Learn. Syst.* **2019**, *30*, 3212–3232. [CrossRef] [PubMed]
36. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Columbus, OH, USA, 23–28 June 2014; pp. 580–587.
37. Girshick, R.B. Fast R-CNN. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015; pp. 1440–1448.
38. He, K.; Zhang, X.; Ren, S.; Sun, J. Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition. *IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI)* **2015**, *37*, 1904–1916. [CrossRef]
39. Lin, T.Y.; Dollár, P.; Girshick, R.; He, K.; Hariharan, B.; Belongie, S. Feature Pyramid Networks for Object Detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Venice, Italy, 22–29 October 2017; pp. 936–944.
40. 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. (TPAMI)* **2017**, *39*, 1137–1149. [CrossRef]
41. He, K.; Gkioxari, G.; Dollár, P.; Girshick, R. Mask R-CNN. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017; pp. 2980–2988.
42. Erhan, D.; Szegedy, C.; Alexander, T.; Anguelov, D. Scalable Object Detection Using Deep Neural Networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Columbus, OH, USA, 23–28 June 2014; pp. 2155–2162.
43. Yoo, D.; Park, S.; Lee, J.Y.; Paek, A.S.; Kweon, I.S. AttentionNet: Aggregating Weak Directions for Accurate Object Detection. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015; pp. 2659–2667.
44. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A.C. SSD: Single Shot MultiBox Detector. In Proceedings of the European Conference on Computer Vision (ECCV), Amsterdam, The Netherlands, 8–16 October 2016; pp. 21–37.
45. Najibi, M.; Rastegari, M.; Davis, L.S. G-CNN: An Iterative Grid Based Object Detector. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 26 June–1 July 2016; pp. 2369–2377.
46. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 26 June–1 July 2016; pp. 779–788.
47. Shen, Z.; Liu, Z.; Li, J.; Jiang, Y.G.; Chen, Y.; Xue, X. DSOD: Learning Deeply Supervised Object Detectors from Scratch. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017; pp. 1937–1945.
48. Li, X.; Shang, M.; Qin, H.; Chen, L. Fast accurate fish detection and recognition of underwater images with Fast R-CNN. In Proceedings of the OCEANS, Genova, Italy, 18–21 May 2015; pp. 1–5 .
49. Qin, H.; Li, X.; Yang, Z.; Shang, M. When Underwater Imagery Analysis Meets Deep Learning: A Solution at the Age of Big Visual Data. In Proceedings of the OCEANS, Genova, Italy, 18–21 May 2015; pp. 1–5 .
50. Dai, J.; Wang, R.; Zheng, H.; Ji, G.; Qiao, X. ZooplanktoNet: Deep Convolutional Network for Zooplankton Classification. In Proceedings of the OCEANS, Monterey, CA, USA, 19–23 September 2016; pp. 1–6.
51. Mahmood, A.; Bennamoun, M.; An, S.; Sohel, F.; Boussaid, F.; Hovey, R.; Kendrick, G.; Fisher, R.B. Coral Classification with Hybrid Feature Representations. In Proceedings of the IEEE International Conference on Image Processing (ICIP), Phoenix, AZ, USA, 25–28 September 2016; pp. 519–523.
52. Moniruzzaman, M.; Islam, S.M.S.; Bennamoun, M.; Lavery, P. Deep Learning on Underwater Marine Object Detection: A Survey. In Proceedings of the Advanced Concepts for Intelligent Vision Systems, Antwerp, Belgium, 18–21 September 2017; pp. 150–160.

53. Wang, C.C.; Samani, H.; Yang, C.Y. Object Detection with Deep Learning for Underwater Environment. In Proceedings of the International Conference on Information Technology Research (ICITR), Moratuwa, Sri Lanka, 10–13 December 2019; pp. 1–6.
54. Chen, L.; Liu, Z.; Tong, L.; Jiang, Z.; Wang, S.; Dong, J.; Zhou, H. Underwater Object Detection using Invert Multi-Class Adaboost with Deep Learning. In Proceedings of the International Joint Conference on Neural Networks (IJCNN), Glasgow, UK, 19–24 July 2020; pp. 1–8.
55. Han, F.; Yao, J.; Zhu, H.; Wang, C. Underwater Image Processing and Object Detection Based on Deep CNN Method. *J. Sens.* **2020**, *2020*, 6707328. [CrossRef]
56. Rizos, P.; Kalogeraki, V. Deep Learning for Underwater Object Detection. In Proceedings of the 24th Pan-Hellenic Conference on Informatics, Athens, Greece, 20–22 November 2020; pp. 175–177.
57. Yeh, C.H.; Lin, C.H.; Kang, L.W.; Huang, C.H.; Lin, M.H.; Chang, C.Y.; Wang, C.C. Lightweight Deep Neural Network for Joint Learning of Underwater Object Detection and Color Conversion. *IEEE Trans. Neural Networks Learn. Syst. (TNNLS)* **2021**, 1–15. [CrossRef]
58. Li, M.; Mathai, A.; Lau, S.L.H.; Yam, J.W.; Xu, X.; Wang, X. Underwater Object Detection and Reconstruction Based on Active Single-Pixel Imaging and Super-Resolution Convolutional Neural Network. *Sensors* **2021**, *21*, 313. [CrossRef]
59. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Region-Based Convolutional Networks for Accurate Object Detection and Segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2016**, *38*, 142–158. [CrossRef]
60. Bochkovskiy, A.; Wang, C.Y.; Liao, H.Y.M. YOLOv4: Optimal Speed and Accuracy of Object Detection. *arXiv* **2020**, arXiv:2004.10934.
61. Wang, C.Y.; Liao, H.; Yeh, I.H.; Wu, Y.H.; Chen, P.Y.; Hsieh, J.W. CSPNet: A New Backbone that can Enhance Learning Capability of CNN. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Virtual, 14–19 June 2020; pp. 1571–1580.
62. Huang, G.; Liu, Z.; Van Der Maaten, L.; Weinberger, K.Q. Densely Connected Convolutional Networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Venice, Italy, 22–29 October 2017; pp. 2261–2269.
63. Deng, J.; Dong, W.; Socher, R.; Li, L.J.; Li, K.; Li, F.F. ImageNet: A large-scale hierarchical image database. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Kyoto, Japan, 27 September–4 October 2009; pp. 248–255.
64. Liu, S.; Qi, L.; Qin, H.; Shi, J.; Jia, J. Path Aggregation Network for Instance Segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CCVPR), Salt Lake City, UT, USA, 18–23 June 2018; pp. 8759–8768.
65. Redmon, J.; Farhadi, A. YOLOv3: An Incremental Improvement. *arXiv* **2018**, arXiv:1804.02767.
66. Beaudoin, L.; Avanthey, L. Underwater Field Equipment of a Network of Landmarks Optimized for Automatic Detection by AI. In Proceedings of the IGARSS 2020—2020 IEEE International Geoscience and Remote Sensing Symposium, Virtual, 26 September–2 October 2020; pp. 1572–1575. [CrossRef]