Hole Detection in Aquaculture Net Cages from Video Footage

Arild Madshaven[1], Christian Schellewald[2], Annette Stahl[1]

[1] Dept. of Engineering Cybernetics, Norwegian University of Science and Technology, NTNU, Trondheim, Norway
[2] SINTEF Ocean AS, Trondheim, Norway

ABSTRACT

Frequent inspection of salmon cage integrity is essential in order to early detect and prevent the possible escape of farmed salmon—minimizing the risk of any negative impact for the remaining wild stock of salmon. Current state-of-the-art computer vision-based approaches can detect net-irregularities under "optimal" net and illumination conditions but might fail under real-world conditions. In this paper, we present a novel modularized processing framework based on advanced computer vision and machine learning approaches that allows to effectively detect potential net damages in video recordings from cleaner-robots traversing the net cages. The framework includes a deep learning-based approach to segmenting interpretable net structure from background, transfer learning facilitated classification of potential holes from irrelevance, and computer vision based modules for irregularity detection, filtering, and tracking. Filtering and classification are vital steps to ensure that temporally consistent holes within net structure are reported—and irrelevant objects such as by-passing fish are ignored. We evaluate our approach on representative real-world videos from real cleaning operations and show that the approach can cope with the difficult lighting conditions that are typical for aquaculture environments.

Keywords: Aquaculture, Computer Vision, Deep Learning, U-net, Transfer Learning, Segmentation

1. INTRODUCTION

The rearing of Atlantic Salmon in aquaculture sea cages is an important and fast growing industry in Norway [1]. It comes with several challenges that need to be addressed in order to minimize it's environmental footprint, and economic and ecological costs. Fish escapes under net cage damage is one major concern and can be prevented by regular net inspections. Net inspections are often carried out by a team of divers or manual inspection of video captured by Remotely Operated Vehicles (ROVs) equipped with cameras [2]. The former approach is usually related to higher costs and longer delays than the latter, in addition to greater health, safety, and environment (HSE) concerns (for instance [3, 4]). Underwater drones may in principle serve to completely automate the process of continuous net integrity inspection if a robust algorithm can process its video stream and evaluate the pictured net structure. In this paper we introduce a modular framework for automatically analyzing the net integrity of a sea cage based on video streams from cleaner-robots. The challenges in automatic processing and analysis of underwater net cage structure [5] are manifold; due to the turbidity of the water, caustics, reflections along with possible low light conditions the video quality might be poor even with high quality cameras, causing the net structure to appear "broken" in some video frames. The water current and waves might cause spatial deformations in the net structure. Fish regularly occlude the net and may appear very similar to holes. In addition, heavy algae growth often covers the net structure to a certain degree. These are all reasons why *proof-of-concept* hole detection algorithms in *well defined* environments and *robust* hole detection algorithms intended for *real* environments constitute different difficulty levels. Figure 1 illustrates the proposed hole detection framework pipeline for a video footage taken during a real industrial net cleaning operation.

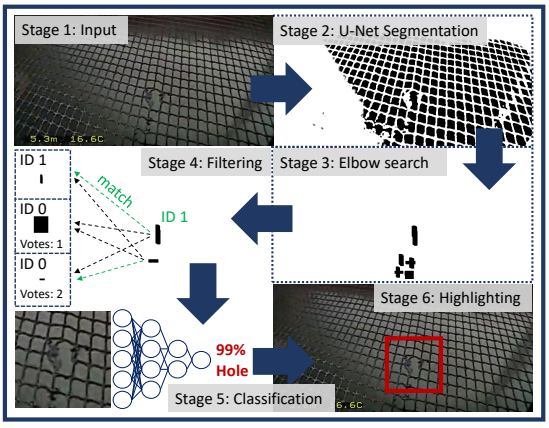


Figure 1: The proposed automated hole-detection framework is able to detect and track a potential hole in difficult real world video-streams.

1.1 Contributions

We propose a novel framework for automatic analysis of the net integrity based on video recordings from cleaner-robots in a sea cage. Furthermore, we introduce two strategies for net segmentation: Firstly, an attentional mechanism which is critical for determining which parts of the recording actually represent net structure that is suitable for further analysis. This is implemented as neural network based segmentation of the video-frames into three classes: bright-net, dark-net, and non-net. This strategy, which we call Three-Class Attention Segmentation (3CAS), is followed by an adaptive thresholding algorithm [6] to achieve coherent segmentation of the net structure as white pixels on black background regardless of the color gradients in the original image. The call for three classes originates in the observation that dark-net on bright background, bright-net on dark background, and irrelevance, frequently coappear in real recordings. Our second proposed strategy for net segmentation, which we call Net Thread Segmentation (NTS), combines the attention and binarization into a single operation, and yielded favourable results also reducing overhead with respect to the processing time. Both segmentation strategies were successfully implemented for our use-case based on the MultiRes U-Net [7].

2. RELATED WORK

The automatic analysis of net cage structure integrity is an important step towards more autonomy in aquaculture. Different sensor modalities such as cameras or acoustic based sensors are used to address the problem. Computer vision based approaches analyzing the images or video streams are mainly based on image processing techniques [8–10]. Paspalakis et al. proposed in [9] two strategies to detect net tear. Their first approach was designed to be paralellizable: the frame was binarized using Otsu's method [11] and then divided into a grid of overlapping cells. By counting the number of net pixels per cell, they recognized holes where the net pixel count was significantly low relative to the net pixel count of other cells in that image. Their second approach utilized the Hough Line Transform [12] to recognize the most prominent straight lines in the image, identifying holes where such lines were far apart from the nearest net structure pixel in the binary image. Betancourt et al. [13] resembled other works with respect to several aspects such as the initial binarization of each frame with Otsu's method. Following binarization, they applied the Hough Line Transform to recognize the mesh structure, and from this they identified the knot points in the net, the properties of which where used to reveal holes. The authors tested their scheme on a real fish cage. Their results section depicts test-images, on which their algorithm performed decently—reconstructing the net structure with high accuracy and recognizing 79% of present holes. However, real world videos tend to violate some of the underlying assumptions as the net structure may appear bent and more irregular or even broken in occasional frames. In addition, challenges such as algae growth and occluding fish (which are crucial to cope with in real-life fish cage inspection applications) need to be addressed, as they complicate the process of reconstructing mesh structure features such as knot points, and in certain situations resemble net holes. The authors of [14] applied an ensemble of image processing modules, consisting of distortion correction, underwater image dehazing, marine growth segmentation, net-opening structure analysis, and blocked percentage estimation. To evaluate the proposed method, several underwater images were collected and labeled with pixel-wise annotations. A first attempt to utilize deep learning for net hole detection was performed by [15] where "You Only Look Once" (YOLO) [16] was exploited to detect net structures and net hole areas in well confined and controlled underwater lab conditions. Within our developed method we make use of neural networks for both segmentation of net-structure and classification of scene content (i.e. is the determined irregularity actually a fish or a piece of seaweed?). However, also more classical computer vision and image processing techniques for tracking and thresholding among others are integrated and used in our proposed framework.

3. APPROACH

In our approach we established five main modules (as displayed in figure 2) all cooperatively addressing the problem of robust hole detection in realistic environments. The first module is the deep learning based net segmentation, either based on 3CAS and subsequent binarization, or direct NTS, translating every video frame into a binary image. Black pixels represent mesh holes, called *background*, and white pixels represent net structure and irrelevant parts of the image. The second module is the local irregularity detector, a module that analyzes the binary image and scans it for atypical patches of background thereby finding potential holes. The third module is a spatiotemporal filter which tracks local irregularities and verifies those irregularities that persist in space and time. The fourth module is concerned with tracking verified irregularities and the computation of their trajectories. It keeps track of all verified irregularities that are still active in the current frame, and predicts the position of active irregularities if they are not directly discovered. The fifth module is based on the scene interpreter—a deep convolutional network originally trained on millions of images [17] and then specialized

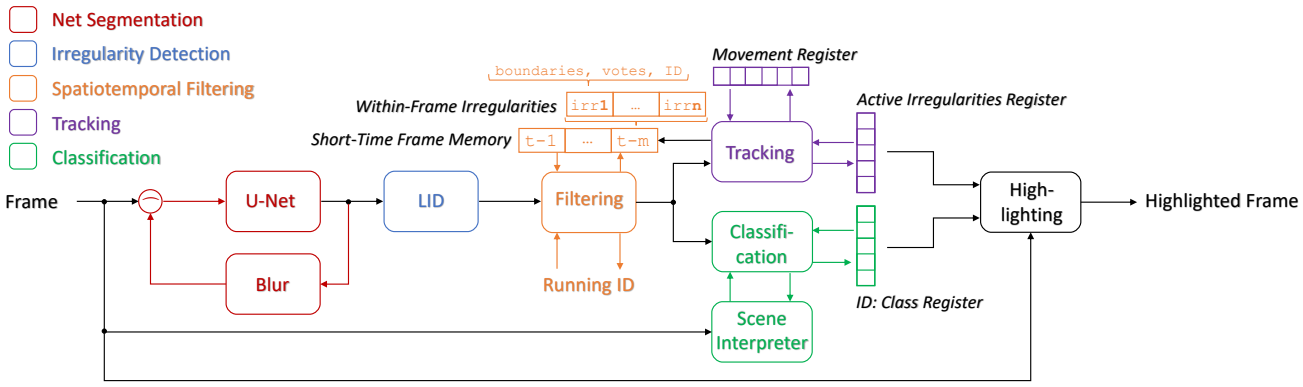


Figure 2: The suggested framework consists of five main modules: *net segmentation* (interpreting the original frame as a binary image), *irregularity detection* (detecting irregular patterns in the binary image), *spatiotemporal filtering* (verifying irregularities that persist in space and time), tracking (registering the trajectories of currently visible irregularities and projecting them onto the frame if need be), and *classification* (discriminating plausible holes from noise).

on discriminating images of fish, or elsewise non-hole objects like floating seaweed or equipment, from images of potential real holes. Passing all stages, a hole can be reported, highlighted, and archived. The modules are described in the following subsections in more detail.

3.1 Neural Network Based Cage-Net-Segmentation

In realistic applications, only a certain part of the video frame depicts interpretable net structure. In order to analyze such images for net damage, one must solve the problem of attention; the algorithm needs to decide which parts of the image to consider for further analysis, and which parts to disregard. Specifically, our subsequent irregularity detection scheme does not analyze the *net structure* per se, but the properties of the background areas *in-between* net structure. A sufficient binary representation of the frame is therefore one where foreground-pixels represent net structure and irrelevant areas, and background pixels represent in-between net structure areas. Below we describe the two different attentional mechanisms that we developed for this purpose, both exploit adapted implementations of the MultiRes U-Net [7].

3.1.1 Three-Class Attention Segmentation

The 3CAS approach provides a three-channel output image, an *attention mask*, classifying each pixel of the video frame as either bright-net, dark-net, or non-net. Non-net refers to all regions that constitute non-net areas or net-areas that might not be reliable enough for determining if a net-irregularity is present or not. Thus, in our implementation of 3CAS, the MultiRes U-Net [7] was modified to comply with three classes by introducing additional filters in the final layer to obtain the desired 3-channel output. The left-to-middle part of figure 3 shows an instance from the training dataset of

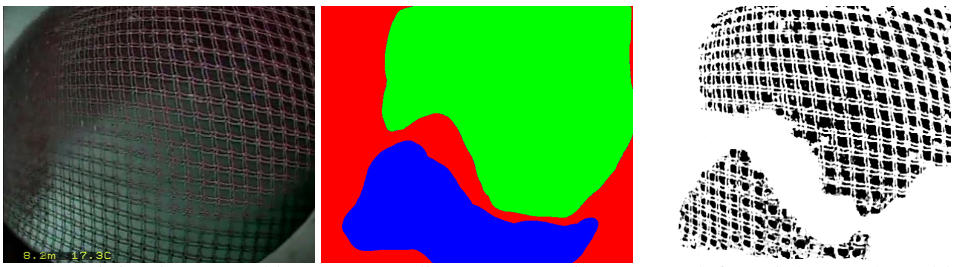


Figure 3: An example training image and its corresponding segmentations. From left to right: Real-world example image, 3CAS ground truth, and NTS segmentation result.

3CAS, where each pixel of the ground truth mask keeps a certain RGB-color value: non-net is encoded as [1,0,0] (red), bright-net as [0,1,0] (green), and dark-net as [0,0,1] (blue). Note that this image also illustrates that bright and dark net regularly coexist in real-world underwater fish-cage scenes. To obtain a coherent binarization of the original video frame (one where net structure is always represented by white pixels and background always by black pixels) we applied an

adaptive thresholding scheme to the video frame [6] combined with the suggested 3CAS masks to disregard irrelevance, and to invert dark-net areas.

3.1.2 Net Thread Segmentation

The NTS procedure was designed to seamlessly binarize video frames containing bright-net, dark-net, and non-net in a single operation. Whereas 3CAS provides attention masks to guide subsequent binarization modules, NTS carries itself out the binarization of net structure and removes non-net components of the video frame. The rightmost part of figure 3 shows an NTS segmentation on a previously unseen image, where each pixel of the ground truth mask keeps a binary value; 0 for background and 1 for anything else.

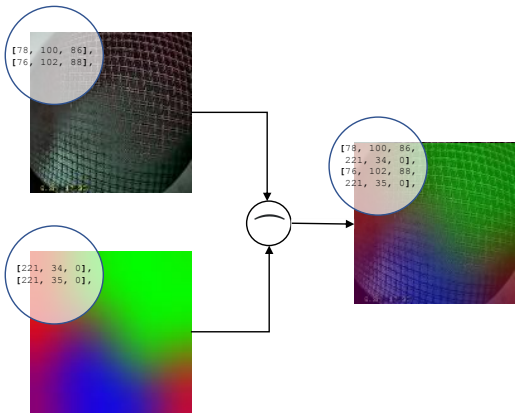


Figure 4: Illustration of pixel-wise concatenation of an image and a blurred lag mask carrying previous segmentation information. Each pixel of the 3CAS input images was expanded from 3 channel RGB-values to 6 channels.

To encourage spatiotemporal continuity, in both segmentation schemes additional *lag masks* containing segmentation information from the previous frame were incorporated in the current input image, effectively facilitating information transfer from the previous segmentation result. In this manner we accumulate and exploit knowledge from the previous video frame sequence instead of evaluating each frame independently. We modified the input layer of the MultiRes U-Net to comply with this idea, increasing the number of channels per input pixel from 3 to 6 in the 3CAS model (cf. figure 4), and from 3 to 4 in the NTS model. Instead of directly appending the previous segmentation results they are first blurred with a Gaussian filter. This compensates for a possible high discrepancy between the previous and current segmentation in particular if the scene changes rapidly. Additionally, we introduced a regularization that acts during training by replacing 25% of the lag masks with all-black masks, and 25% with all-white masks. This method regulated the models' trust towards the image information of the current video frame contra the previous lag mask, effectively preventing deadlock-situations where only the lag mask is considered.

3.2 Local Irregularity Detector

The local irregularity detector (denoted LID in figure 2) analyzes the binary image produced by the segmentation module and detects in it irregular background regions. The module works on single video frames, independently, and outputs a morphologically [18] closed binary image called the *irregularity space* (compare Stage 3 in Figure 1) where black pixels represent irregular areas. Furthermore, we introduce an adaptive variable called the *elbow* to describe the largest morphological structuring element that allows *K patches of background* to persist in the binary image after applying the morphological closing operation to it. Effectively, we close the binary image with a structuring element of increasing size until K such patches can no longer be counted in the closed image, and we refer to the previous kernel size as the elbow. Having attained the elbow value, we increase the kernel size with X% and apply a new closing operation. The parameter X decides the relationship between the K-th largest patch and the detected irregularity. Intuitively, a single broken mesh thread can result in a hole that is 100% wider or taller than its neighbors, but spatial deformations in the net structure can significantly lower this number. We set X = 50 and hence require an irregularity to be at least 50% larger than the K-th largest patch. Lowering this number increases the number of false reports, but increases also the recall in terms of not overlooking true irregularities. Increasing X yields in turn a more conservative hole detection scheme that reports only severe damage.

In figure 5 it can be observed that the number of persisting patches decreases as the kernel size is incremented, but the presence of irregularities brings forth a plateau in the plot. The detection of this plateau with the elbow approach is analogous to finding the *K-th largest patch and assuming that no regular patch is X% larger*. Since the elbow represents

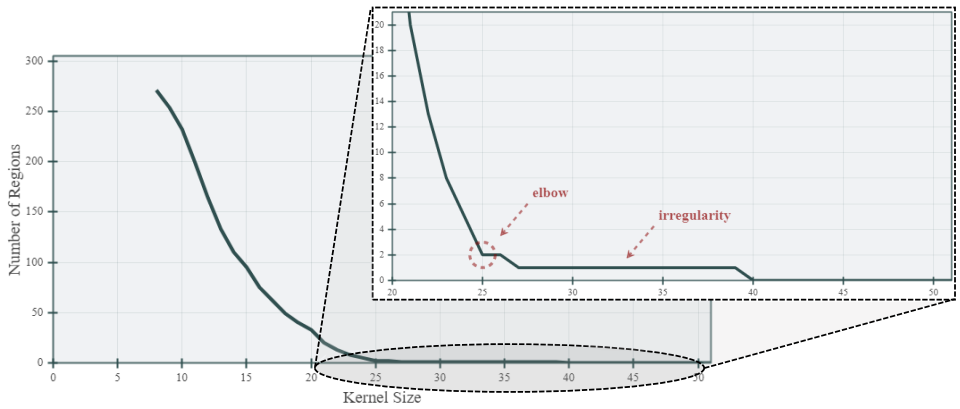


Figure 5: Holes manifest themselves as plateaus in the plot when we apply the closing operator to the binary image with an iteratively increasing kernel size and count the persisting background regions. Our irregularity detection scheme reveals these plateaus by finding the *elbow*. We close the binary image with an iteratively increasing kernel size until we can no longer count *K* regions (we set K=3). In this particular case, this kernel size (elbow value) is 24 (the largest kernel size to which 3 regions persist after closing). We then increase the kernel size by X% (we set X=50) and suggest that regions which persist after a new closing operation are irregular. As for this example, the one region that persists after closing with kernel size $elbow \cdot 1.5 = 36$ is an irregularity.

the size of the K-th largest piece of background in the binary image, it is temporally stable (since the zoom level varies little from frame to frame) and the search for the elbow can thus be optimized by initializing the search at the elbow value of the previous frame. As for the parameter K, choosing it too large means a higher computational demand since more regions need to be counted for each iteration. The lower the K, however, limits the number of irregularities that can be detected. With K = 3, we obtained a fast-converging scheme that was capable of detecting two irregularities per *tile*.

One drawback to this approach is the assumption that a hole is always larger than the K-th largest piece of background in the *global* image. This is not always the case, for instance if the hole is further away from the camera than intact net structure in the foreground. We coped with this issue by splitting the binary image into 16 tiles, and evaluated each tile independently. With this strategy, irregularities were judged by their appearance relative to their immediate neighborhood and not the global image.

Single broken net threads can result in rectangular holes, which are not easily discoverable with square or disk-shaped structuring elements. We therefore employed a procedure with rectangular kernels which grew first in the horizontal direction and later in the vertical direction. With such kernels, we enabled the discovery of irregularities that deviated from their neighborhood in either height or width but not necessarily in both directions simultaneously.

Furthermore—if utilizing 3CAS, we propose to use the elbow to derive a fitting neighborhood size for the adaptive thresholding algorithm [6]. The neighborhood size should be large enough to encapsulate an area consisting of both net structure and background, but small enough to preserve the adaptive qualities when shifted across an image with a color gradient. Intuitively, the elbow as identified with a cross-shaped kernel reflects the properties of the mesh structure in the image. Our experiments concluded that an adaptive neighborhood size of six times the size of the elbow yielded satisfactory binary results regardless of the distance between net and camera.

3.3 Spatiotemporal Filtering

The spatiotemporal filtering module determines which distinctive irregularities, detected with the local irregularity detector, are consistent with detected irregularities in previous frames. This allows short term tracking over consecutive frames, a necessary precondition for later verification. Note that longer term tracking, capable of coping also with missing observations is handled within the tracking module. The first step within this module isconsidering the irregularity space which and how many different irregular regions are present in the current frame. All disjoint regions are stored as bounding boxes with their respective position and size. Subsequently, we match the current set of detected irregularities with the irregularities detected in previous frames.

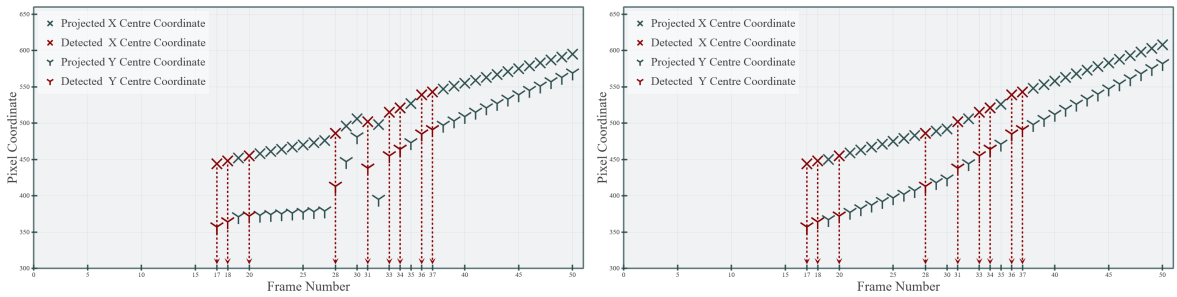


Figure 6: Left: Detected and projected center coordinates with movement hypothesis based on instantaneous movement between the two most recent reports. Right: Detected and projected center coordinates with movement hypothesis based on median movement in five preceding reports.

In our implementation we compared the intersection over union (IoU) ratio of the bounding boxes (also known as Jaccard index [19]) to determine if previous regions match (=overlap) with current regions. For coping with fast-moving scenes one has to use lower thresholds on the IoU for tracking the irregularities (i.e. a threshold of 0.15 worked well for our real world scenes). In order to increase robustness three consecutive frames stored in a *short-time frame memory* are explored to match the irregularities. For a matched or re-identified irregularity we increase its accumulated vote count and update its verification number if the matched irregularity is also verified. To verify that a real irregularity is present, we require a vote count of seven to be reached, meaning that the irregularity must satisfy the overlap criterion for at least seven frames. Once verified, the irregularities get a unique ID assigned. This verification procedure effectively nullified the occurrence of sporadic appearing irregularities mainly caused by occasional poor segmentation. All verified irregularities are stored in the *active irregularities register* with a timer that resets every time the irregularity is re-discovered.

3.4 Tracking

The tracking module keeps track of the trajectories of detected and verified irregularities over time. In the case of the disappearance of an irregularity (that is not at the border) this module also predicts and inserts its anticipated location based on the previous observations. This is performed by considering its apparent speed and size. For instance, a poor segmentation may cause a verified irregularity to be overlooked by the local irregularity detector for a number of frames. In this case, our implementation updates its boundaries within the short-time frame memory and also considers it as active irregularities register is updated based on the motion hypothesis corresponding to the considered irregularity.

We calculated the relative movement of an irregularity as the spatial difference between its center coordinate (x^c, y^c) in frame $t-1$ compared to frame t. Our first movement hypothesis utilized the instantaneous movement from the past two reports to project the irregularity onto the current scene. Our second hypothesis calculated the median movement from the past five reports, instead, which provided a significantly better tracking of the irregularity. These two hypotheses are visualized on the left and right side of figure 6 respectively, showing instantaneous and median movement projection, respectively.

3.5 Classification

Although non-net objects such as fish and equipment are supposed to be handled by the segmentation module, a separate module was developed to effectively verify or falsify arising hole reports. The pre-trained VGG16 [20] model was fine-tuned on 300 images of the three classes *net*, *fish*, and *irrelevance* (see figure 7). By assuming that holes can exclusively exist within *net*, we only verify hole reports if the surrounding area in the video frame is evaluated as such by the scene interpreter. By introducing the *ID: class register*, we effectively map irregularity IDs with a class and subsequent highlighting can decide whether or not to focus on the irregularity based on its proposed class.

For instance, fish can be highlighted with green color, and holes with red.

The module was tested on a set of 300 test images, and its performance saw slight improvement with data augmentation, increasing precision scores on fish class from 0.83 to 0.84 and net class from 0.89 to 0.91, with irrelevance class put at 0.91. Recall scores improved on fish class, from 0.86 to 0.89, and on net class, from 0.89 to 0.91, whilst irrelevance class decreased some, from 0.87 to 0.86.

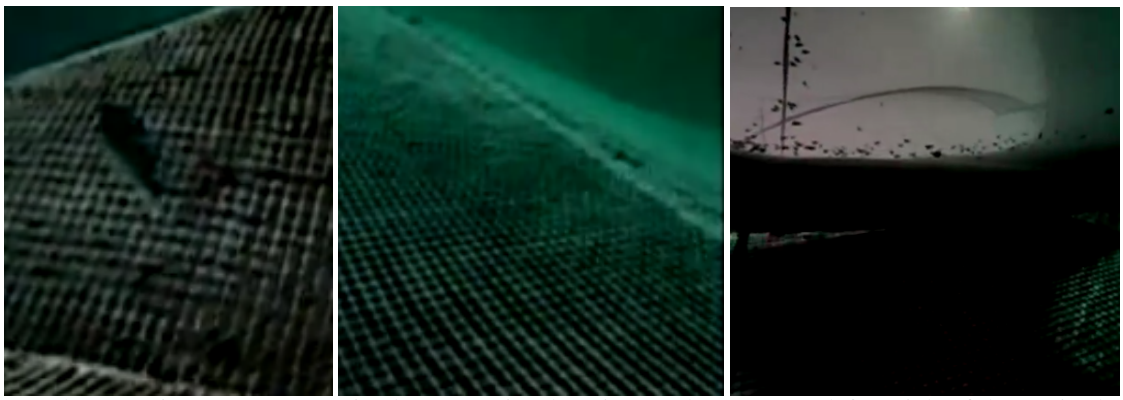


Figure 7: The scene interpreter was trained to recognize three image classes. From left to right: fish, net, irrelevance.

4. RESULTS

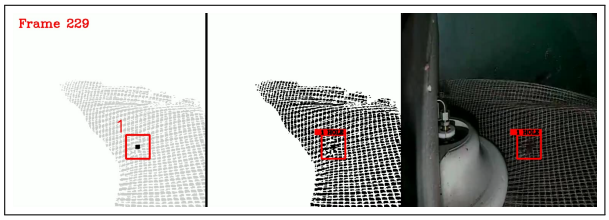
The effectiveness of the framework was investigated on ten ten-second test videos. These were extracted from videos of two cleaning operations and were not utilized during the U-Net training stages. The videos displayed holes in the net structure (No. 1, 2, 5, and, 9), and swimming fish (No. 8 and 9). Five videos (No. 3, 4, 6, 7, and 10) were hole- and fish-free. We investigated binary image quality (representation of the net structure), robustness and effectiveness of the local irregularity detector, irregularity classification (net, fish, and irrelevance), and tracking quality. Additionally, in a second test trial using the same video material we added salt-and-pepper noise, corrupting 2% of the pixels of each frame. These tests were executed to investigate the noise influence on the performance of the framework. The performance scores from all tests are summarized in table 9.

Within our suggested framework we incorporated previous scene knowledge in the segmentation module through the integration of "lag masks" containing information from past segmentations. The Net Thread Segmentation (NTS) performed better in comparison to the Three-Class Attention Segmentation (3CAS), providing stable results and seamlessly combining segmentation, binarization, and denoising in a single operation. However, compared to the 3CAS approach, NTS turned out to be more sensitive to salt-and-pepper noise—ultimately leading to an increased number of detected irregularities. A solution likely to improve this tendency is to increase the training data foundation and to introduce additional regularization as noisy input frames during U-Net training stages. The developed scene interpreter discriminates effectively floating potential non-holes highly accurately, but apparent stationary irregularities in a moving scene are difficult to distinguish from real holes.

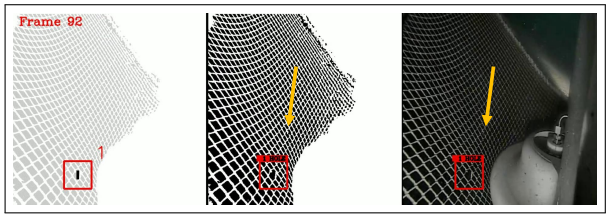
In the future we intend to increase the robustness of our approach towards different types of noise by enlarging the training dataset with images disturbed by the considered noise-types. Naturally, the training data foundation will be significantly extended when we are able to perform more experiments. A parallelized version of our approach can be designed by independently analyzing the different overlapping tiles of a image. The tracking module might be improved by integrating more sophisticated motion hypotheses allowing to cope better with circumstances when the movement is non-constant.

5. CONCLUSION

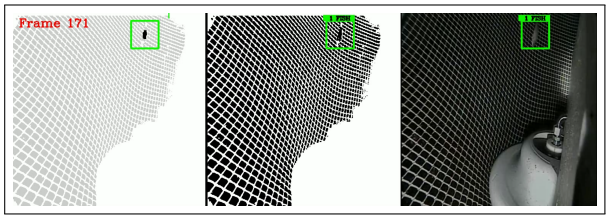
We presented a novel framework, consisting of five main modules, for a robust visual-based net hole detection in realistic aquaculture underwater environments. Two alternative strategies (NST and 3CAS) facilitating deep learning for net structure segmentation were applied on noise-free as well as noisy video data. Specifically, the MultiRes U-Net with access to lag masks led to excellent performances and produced well defined binary representations of the test videos. The local irregularity detector, which utilizes a morphological scheme along with a measure (the *elbow*) for a typical background patch analyses every binary image. The combination of the local irregularity detector with the introduced spatiotemporal filtering to report only those irregularities that sustain themselves in both space and time, led to robust detection. The problem of over-reporting when noise was introduced, leading to higher computational costs, could be addressed in future investigations by tuning specific parameters of the detector, and by regularization strategies during U-Net training stages. More sophisticated motion hypotheses could be deduced to achieve better tracking under circumstances when the movement is not constant.



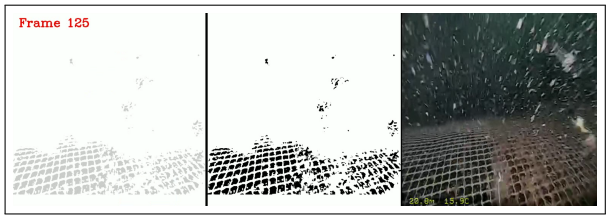
(a) Reported hole in test video 1.



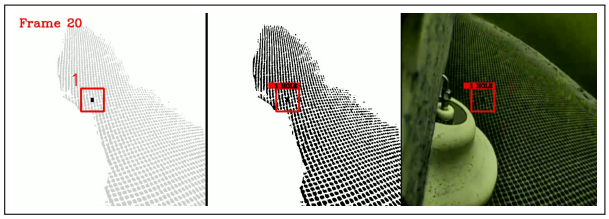
(b) Reported hole and overlooked hole in test video 2.



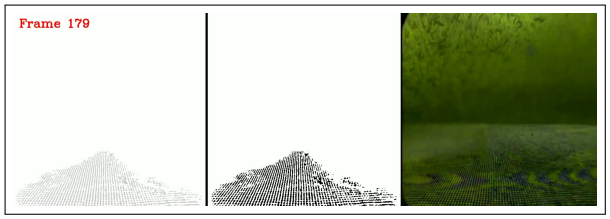
(c) Reported fish in test video 3.



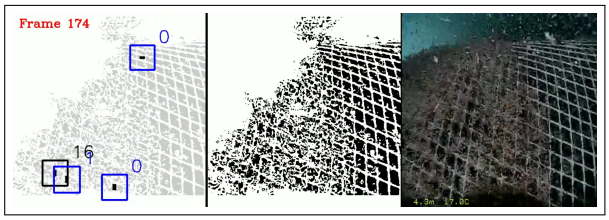
(d) Scene from test video 4.



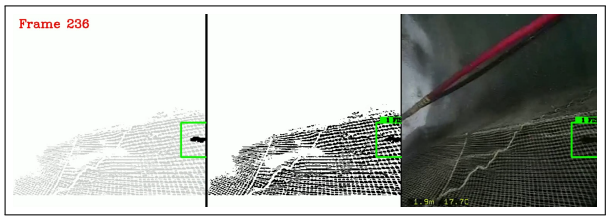
(e) Scene from test video 6.



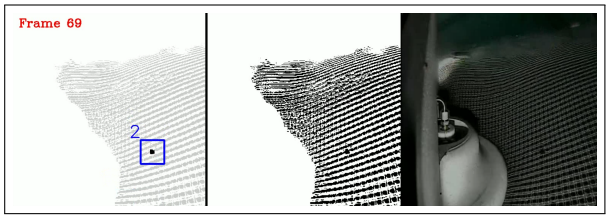
(f) Scene from test video 7.



(g) Scene from test video 7.



(h) Reported fish in test video 8.



(i) Plausible hole in test video 9 accumulating votes.

(j) Scene from test video 10.

Figure 8: Each video excerpt shows three views; the leftmost view is the irregularity space in which local irregularities arise and accumulate votes. Red squares indicate verified holes (as classified by the scene interpreter), green squares are verified fish, black squares are verified irrelevance, and blue squares are unverified irregularities which are in the process of accumulating votes. The number accompanying each square is the vote count of unverified irregularities, or the ID of verified ones. The middle view shows the binary representation as proposed by NTS, and the rightmost view is the output view where only verified fish and holes appear.

No.	View	Segmentation	NIRR	HTP	HFP	HFN	FTP	FFP	NTP	NFP	SP30F
1	S	Mostly Excellent	152	1	0	0	0	0	0	0	2.38
1N		Mostly Poor	659	0	0	1	0	1	7	0	2.87
2	P	Mostly Excellent	169	1	1	1	0	0	0	0	2.47
2N		Decent	1181	1	0	1	0	0	10	0	11.63
3	P	Excellent	109	0	0	0	0	1	0	0	2.43
3N		Mostly Good	523	0	0	0	0	0	2	0	2.87
4	F	Very Good	584	0	0	0	0	0	1	0	3.06
4N		Poor	2154	0	0	0	0	0	18	0	20.39
5	S	Very Good	148	2	0	0	0	0	1	0	2.20
5N		Poor	1409	0	1	2	0	0	18	0	6.99
6	F	Good	74	0	1	0	0	0	0	0	2.18
6N		Poor	881	0	0	0	0	0	6	0	4.00
7	F	Decent	1769	0	0	0	0	0	19	0	22.53
7N		Poor	2416	0	0	0	0	0	25	0	21.60
8	F	Excellent	126	0	0	0	1	0	0	0	2.21
8N		Poor	2764	0	0	0	0	0	32	0	17.43
9	S	Excellent	291	1	0	1	2	0	1	0	2.44
9N		Mostly Good	834	0	0	2	1	0	6	0	3.88
10	F	Varying	993	0	0	0	0	0	13	0	3.19
10N		Poor	2691	0	0	0	0	0	26	0	45.67

Figure 9: Noisy tests are tagged with N-suffixes. Views: S(tarboard), P(ort), F(ore). NIRR is the number of local irregularities (not necessarily verified). Subsequent short-hands are True Positives (-TP), False Positives (-FP), and False Negatives (-FN), for Holes (H-), Fish (F-), and Nonsense (N-), all applicable to verified irregularities only. SP30F is mean execution time [s] per 30 frames. The NVIDIA Titan X GPU was utilized during testing.

ACKNOWLEDGMENTS

We would like to thank Frøy Gruppen for providing the video footage from net-cleaning operations.

REFERENCES

[1] "Skattlegging av havbruksvirksomhet : utredning fra utvalg oppnevnt ved kongelig resolusjon 7. september 2018 : avgitt til finansdepartementet 4. november 2019," (2019).

[2] T. Haugene, Evaluation of Methods for Robust, Automatic Detection of Net Tear with Remotely Operated Vehicle and Remote Sensing, Master's thesis, Norwegian University of Science and Technology, Trondheim (2014).

[3] W. Haugerud, "Kan rov'er erstatte dykkere i havbruksnæringen?," (11 2019). Accessed: 01.12.2020.

[4] R. A. H. Jakobsen, Automatic Inspection of Cage Integrity with Underwater Vehicle, Master's thesis, Norwegian University of Science and Technology, Trondheim (2011).

[5] A. Duda, J. Schwendner, A. Stahl, and P. Rundtop, "Visual pose estimation for autonomous inspection of fish pens," in OCEANS 2015 - Genova, 1–6 (2015).

[6] S. Z. Li and A. Jain, eds., Local Adaptive Thresholding, 939–939, Springer US, Boston, MA (2009).

[7] N. Ibtehaz and M. S. Rahman, "Multiresunet : Rethinking the u-net architecture for multimodal biomedical image segmentation," Neural networks **121**, 74–87 (2020).

[8] Y.-P. Zhao, L.-J. Niu, H. Du, and C.-W. Bi, "An adaptive method of damage detection for fishing nets based on image processing technology," Aquacultural Engineering **90**, 102071 (2020).

[9] S. Paspalakis, K. Moirogiorgou, N. Papandroulakis, G. Giakos, and M. Zervakis, "Automated fish cage net inspection using image processing techniques," IET Image Processing **14**(10), 2028–2034 (2020).

[10] V. Jovanović, V. Risojević, Z. Babić, E. Svendsen, and A. Stahl, "Splash detection in surveillance videos of offshore fish production plants," in 2016 International Conference on Systems, Signals and Image Processing (IWSSIP), 1–4 (2016).

[11] N. Otsu, "A threshold selection method from gray-level histograms," IEEE Transactions on Systems, Man, and Cybernetics **9**(1), 62–66 (1979).

[12] J. Illingworth and J. Kittler, "A survey of the hough transform," Computer vision, graphics, and image processing **44**(1), 87–116 (1988).

[13] J. Betancourt, W. Coral, and J. Colorado, "An integrated rov solution for underwater net-cage inspection in fish farms using computer vision," SN Applied Sciences **2**(12), 1–15 (2020).

[14] W. Qiu, V. Pakrashi, and B. Ghosh, "Fishing net health state estimation using underwater imaging," Journal of Marine Science and Engineering **8**(9) (2020).

[15] Q. Tao, K. Huang, C. Qin, B. Guo, R. Lam, and F. Zhang, "Omnidirectional surface vehicle for fish cage inspection," in OCEANS 2018 MTS/IEEE Charleston, 1–6 (2018).

[16] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), (June 2016).

[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," Commun. ACM **60**, 84–90 (May 2017).

[18] R. M. Haralick, S. R. Sternberg, and X. Zhuang, "Image analysis using mathematical morphology," IEEE Transactions on Pattern Analysis and Machine Intelligence **PAMI-9**(4), 532–550 (1987).

[19] P. Jaccard, "The distribution of the flora in the alpine zone. 1," New phytologist **11**(2), 37–50 (1912).

[20] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," (2015).

Paper ID 24 - AUTHORS' BACKGROUND

Name	Title	email/ Personal website	Research Field
Arild Madshaven	Master Student	arild.madshaven@gmail.com	Cybernetics, Robotics, Computer Vision, Machine Learning
Christian Schellewald	PhD, Researcher	christian.schellewald@sintef.no https://www.sintef.no/christian.schellewald	Computer Vision, Machine Learning, Physics
Annette Stahl	Associate Professor	annette.stahl@ntnu.no https://www.ntnu.no/ansatte/annette.stahl	Robotics, Computer Vision, Machine Learning, Applied Mathematics