Deep Census: AUV-Based Scallop Population Monitoring

Christopher Rasmussen[*], Jiayi Zhao[*], Danielle Ferraro[†], & Arthur Trembanis[†]

University of Delaware
Newark, DE, USA
cer@cis.udel.edu

Abstract

We describe an integrated system for vision-based counting of wild scallops in order to measure population health, particularly pre- and post-dredging in fisheries areas. Sequential images collected by an autonomous underwater vehicle (AUV) are independently analyzed by a convolutional neural network based on the YOLOv2 architecture [18], which offers state-of-the-art object detection accuracy at real-time speeds. To augment the training dataset, a denoising auto-encoder network is used to automatically upgrade manually-annotated approximate object positions to full bounding boxes, increasing the detection network's performance. The system can act as a tool to improve or even replace an existing offline manual annotation workflow, and is fast enough to function "in the loop" for AUV control.

1. Introduction

Monitoring marine species, both at an individual and population level, is a critical part of protecting the marine ecosystem and managing modern fisheries. The Atlantic sea scallop (*Placopecten magellanicus*), pictured in Fig. 1(a), is highly important economically. Although mobile, it is not migratory, and it is commonly found on the sea floor in the mid-Atlantic at a 35-100 m depth on sand and gravel sediments. Wild scallops are typically caught using a dredge dragged along the seabed, and accurate and timely assessments of local population numbers and size/age distributions are important for setting sustainable quotas. Historically, censuses have been conducted by systematically sampling different locations by dredging, but recently there has been great interest in increasing the efficiency and coverage of this process (including for other species) through analysis of images obtained by fixed, towed, and AUV-borne cameras. Thus far, such analysis has been either manual [20, 6] or primarily based on hand-selected features [1, 10, 3]. In this paper, we demonstrate the efficacy of applying deep learning methods to achieve fast and accurate scallop detections over a range of substrates, despite suboptimal image quality.

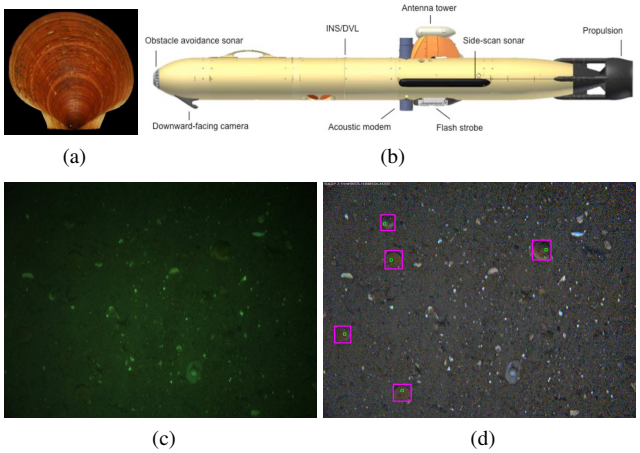


Figure 1: (a) *P. magellanicus*; (b) Gavia AUV; (c) Raw seabed image taken from AUV; (d) Corresponding image after contrast enhancement, with scallops manually annotated (green dots indicate original rough positions from [6])

In recent years, convolutional neural networks (CNNs) have shown great promise when applied to a wide range of computer vision tasks, including image classification, object localization, and object detection [11, 22, 23, 9]. Here, we are concerned primarily with the problem of *detection*, or placing bounding boxes around an unknown number of scallops in each image as illustrated in Fig. 1(d). Because dredging can cause habitat damage and mortality among uncaught scallops [6], detected scallops could be further categorized as *healthy* vs. *compromised*. Also, other creatures such as starfish, skates, crustaceans, and sharks may be of scientific interest. However, in this work we focus

[*]Dept. Computer & Information Sciences, College of Engineering
[†]Dept. Geological Sciences, College of Earth, Ocean, & Environment

on simply identifying healthy scallops, making the task a 1-category detection problem.

Our strategy is to leverage access to an existing dataset of $170,000+$ manually-annotated images captured in scallop-rich waters [6] in order to learn a sufficiently robust visual representation of *P. magellanicus* that the system can recognize individual scallops at different stages of their life cycle and on a wide variety of substrates. The images in the dataset were collected by a downward-pointing digital camera in the nose of an AUV (shown in Fig. 1(b)) moving at an altitude of a few meters above the seabed. Despite illumination with a flash strobe to compensate for low ambient light at the operating depth of 50+ m, the raw images are quite dim (a sample is shown in Fig. 1(c)) and the scallops are often difficult to discern even to a human eye. Other confounding issues include the small size of the scallops in the image; non-uniform illumination or vignetting from the flash; a wide range of similar-looking features such as clams, small rocks, shell hash, and sediment textures; partial burial of some scallops in sand or gravel; and occlusions by swimming creatures.

Despite these difficulties, we demonstrate that by training a YOLOv2 detection network [18] with minimal modifications on just a fraction of the annotated dataset, the system can achieve superior performance on the task. YOLOv2 is a recent neural network architecture that has been demonstrated (when trained on different data) to work extremely well and at high speed on benchmark datasets such as PASCAL VOC [5], which includes classes of objects as diverse as people, bicycles, chairs, and birds. Training on scallops is not completely straightforward, however, as the available annotations contain approximate position information only and must be converted to precise bounding boxes. One novelty of this paper is the description of a method for automatically *upgrading* these annotations by training a separate denoising auto-encoder network, thereby enabling efficient augmentation of the training data.

In the following sections we will review existing work on image-based detection of benthic creatures and deep learning methods for object detection, describe in detail the methods behind our deep scallop census system and the data used to train it, present an evaluation of the system's performance, and discuss several promising future directions for this work.

2. Background

2.1. Image-Based Benthic Creature Detection

A fixed underwater camera was used for decapod (e.g., lobster) detection in [1]. A top-hat transform was applied to images acquired at hourly intervals, followed by thresholding of the distance between the red and green color channels to obtain a binary saliency image. Fourier descriptors and SIFT features were extracted on large connected components, which were then classified using partial least squares discriminant analysis.

An early approach to scallop detection relied on their regular shape, employing a Hough transform to look for circles [4]. Kannappan *et al.* [10] presented a layered saliency-based approach to identifying scallops in AUV images in which top-down visual attention was followed by segmentation, shape extraction, and classification. The image analysis pipeline was hand-tuned, and their results showed a high number of false positives associated with acceptable recall levels.

Another computer vision system for scallop detection, based on the object identification paradigm, was presented in [3]. Images collected from the HabCam towed camera system [26] were subjected to illumination and color correction, then likelihood images based on grayscale and color histograms were formed, and finally four hand-selected operators were applied to identify candidate regions. For each candidate region, a variety of color, texture, and edge features were extracted and fed to a series of cascaded AdaBoost classifiers [7, 21] for the final detection decision. The multiple classifiers were tuned to different substrates and scallop types, but the results were generally quite good.

2.2. Deep Learning Detection Architectures

Early CNN-based object detectors operated either in a sliding window fashion [22] or by generating object "proposals" separately [28, 29] and then evaluating these hypotheses. These approaches achieved good results on difficult detection benchmarks, but were fairly slow and not well-suited to real-time deployment. More recently, it has been shown that CNNs have sufficient power to represent geometric information for localizing objects, opening the possibility of building state-of-the-art object detectors that rely exclusively on CNNs free of proposal generation schemes [19, 17, 18]. In such approaches, the network is trained end-to-end to predict both the appearance and geometric information of an object. At test time, given an input image, the entire network is only evaluated once instead of at different locations and scales of the image, enabling a large speed-up.

Redmon *et al.* developed one such unified CNN for real-time object detection which they called "You Only Look Once" (YOLO) [17]. The core idea of YOLO is that it considers the detection task as a regression problem. Specifically, the input image is divided into an $S \times S$ grid of cells, each of which contains information on B hypothetical object bounding boxes. Each such bounding box is parametrized by a 5-D vector $[x, y, w, h, P(Obj)]$, where $P(Obj) = 1$ if the center of any ground-truth object bounding box is inside the cell and $P(Obj) = 0$ otherwise. Each grid cell also includes a conditional class probability: $\Pr(c \mid$

$Obj)$, where $c \in \{C\}$. Accordingly, the class-specific confidence is given by: $P(c) = Pr(c \mid Obj)P(Obj)$. For each presented image, the output layer of the network is an $S \times S \times (B * 5 + C)$ tensor. Non-maximal suppression is used to remove duplicate detections, followed by thresholding on $P(c)$.

Using values of $S = 7$ and $B = 2$, YOLO achieved a very high mAP (mean average precision) on the 20-class PASCAL VOC 2007 [5] while running at a speed of 45 fps, faster than any comparable system. One shortcoming of YOLO is its difficulty at detecting small objects that are very close to each other since it only predicts two bounding boxes within one class per grid cell. This is a concern for us since each scallop occupies a very small portion of each image.

Recently, Redmon and Farhadi proposed another real-time detection system, YOLOv2 [18], that aims to improve on the localization and recall performance of YOLO. Several key modifications in YOLOv2 include batch normalization on every convolutional layer to enhance convergence and regularize the model, a higher input image resolution of 448×448 and finer grid with $S = 13$, and the replacement of YOLO's fully connected layers which directly regress bounding box coordinates with an "anchor box" concept adapted from [19]. YOLOv2 uses a new classification model as the front end which is somewhat similar to VGG models [23], but with global average pooling to make predictions and 1×1 filters to compress features representations between 3×3 convolutions. The classification model is called "Darknet-19" and has 19 convolution layers and 5 max-pooling layers.

YOLOv2's mAP on PASCAL VOC 2012 is significantly improved over YOLO and comparable to current leaders like Faster R-CNN with ResNet [9] and SSD512 [13], while running from 2 to 10 times faster.

3. Scallop Dredging Dataset

In 2014 and 2015, a Before-After Control-Impact (BACI) [24] study of the effects of dredging on incidental scallop mortality [6] was conducted off the east coast of the United States at bottom depths of 50-80 m in two areas (shown in Fig. 2(a)): the Elephant Trunk Closed Area (ETCA) and Nantucket Lightship Closed Area (NLCA).

During the study each area was divided into 3 sites, and at each site 3 types of experiments were run, denoted by *A*, *B*, and *C*. *A* experiments consisted of a pre-dredging AUV *mission*, followed by 1 dredging tow by a scallop fishing vessel, and then a post-dredging AUV mission hours later to assess the effects of the dredging. *B* experiments were the same, except that the dredge was towed through the site 5 times for a heavier impact. Finally, *C* experiments were controls where no dredging was done, but the AUV went on two missions. Over the course of the study, ETCA was surveyed twice and NLCA once, resulting in a total of 54 AUV missions which we refer to as `M1`, `M2`, ..., `M54`.

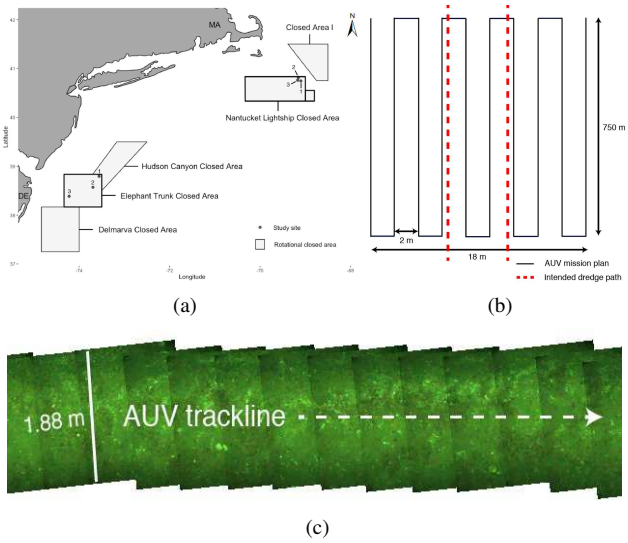


Figure 2: (a) Map of scallop dredging study areas; (b) Lawn-mowing track of typical AUV mission; (c) AUV trackline for image collection. Images are taken from [6].

In order to photographically cover the area where the dredging occurred, each AUV mission followed a preset "lawn-mowing" pattern as depicted in Fig. 2(b). Using its inertial navigation system (INS) and Doppler velocity log (DVL) for state estimation, the AUV followed 10 parallel lines 2 m apart and 750 m in length (ETCA), or 14 550 m-long lines (NLCA), at an altitude of 2.5 m above the sea floor. Given the AUV camera's intrinsic parameters and altitude, each raw 1280×960-pixel image such as that shown in Fig. 1(c) represents a seabed area of roughly 1.88 m $\times$ 1.45 m (an area of 2.73 m^2). The images were captured at 3.75 Hz with the AUV traveling at a speed which resulted in an average overlap of 45% between consecutive images in a line (Fig. 2(c)). Images were not captured while the AUV was turning to begin the next line, so there is a discontinuity between each line of images.

Post-processing and manual annotation Due to the depth of the AUV and the limitations of the strobe used to illuminate the seabed, raw captured images exhibit low contrast that makes scallop detection difficult (e.g., Fig. 1(c)). Accordingly, brightness and color contrast were enhanced on every image using one of two post-processing procedures: either a multiscale retinex algorithm or a stretch contrast function which works on each color channel independently, as detailed in [6]. A sample result of contrast enhancement is shown in Fig. 1(d).

In order to obtain ground-truth information for the

dataset, a team of 15 student annotators were trained to identify scallops (as well as other creatures) and mark them as healthy or compromised using an online annotation system [6]. Each annotator was directed to click *somewhere* inside the shell perimeter, but not necessarily the center. We call this a *rough position*. The green dots (inside the purple boxes) in Fig. 1(d) are a sample of these original scallop position annotations. At ETCA sites 1 and 2, images were downsampled by 2 before annotation – every other image was skipped. However, annotating a single image took an average of almost 30 seconds, so to speed the process the images were downsampled by 8 before annotation at ETCA site 3 and all NLCA sites. In total, 171,860 images were annotated in this fashion during the study at a cost of almost 1,150 person hours [6].

In order to train the YOLOv2 detector network, each scallop needs to have a tight bounding box. For this work, we created a tool to manually *upgrade* the existing scallop *rough position* data to *precise position and scale*. Bounding boxes were constrained to contain an existing rough position, and allowed to be non-square. Where scallops appeared to be partially buried or not completely inside the image, the annotator only indicated the visible part.

We applied this tool to `M49`, a post-dredging mission from NLCA containing 2,430 images with 4,267 original rough position annotations of all types of creatures. 1,736 of these images contained at least one healthy scallop, of which there were 3,863 total. Precise bounding boxes were added for these; samples for one image are shown as purple rectangles in Fig. 1(d). This entire process took roughly 20 hours.

Other missions which we will reference from the original study dataset:

- `M46`: `M49`'s pre-dredging twin with 1,857 annotated images that contained at least one healthy scallop; there were 4,835 healthy scallops in all.
- `M6`: An ETCA mission with 5,204 healthy scallop-containing images (because of less downsampling); 8,663 healthy scallops total.

4. Automatically upgrading rough positions to bounding boxes

Many studies have supported the rule of thumb that machine learning performance is improved through a larger and more diverse training set, both in general [2, 14] and more recently for CNN-based object detection/recognition [25]. One can also observe on the leaderboard for the widely-used PASCAL VOC detection challenge that mAP scores increase dramatically for algorithms which use COCO + PASCAL data (100K examples) vs. PASCAL alone (10K examples) [5, 12].

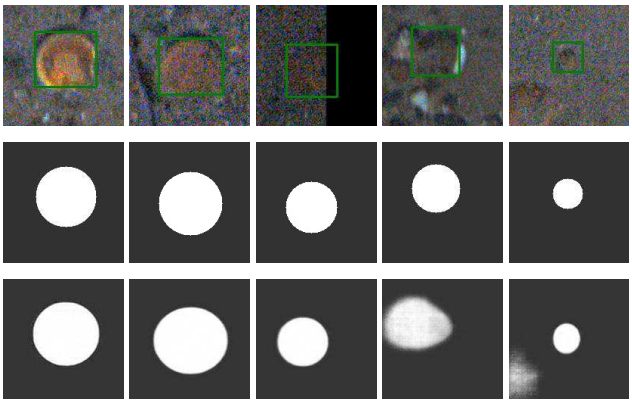


Figure 3: (1st row) Sample scallop subimages from `M49` test set centered on rough position annotations with ground truth bounding boxes overlaid; (2nd row) Corresponding circular masks used to train *heatmap* neural network; (3rd row) Corresponding outputs of heatmap network

In our dataset, the rough positions in the entire original set of annotations represent one to two orders of magnitude more training examples than `M49` alone, yet YOLOv2 training requires that each of these be converted to a bounding box. Based on our experience with applying the software tool described above to `M49`, doing this manually for the entire dataset could take up to 1000 hours, almost as much time as the original annotation project.

Instead, we propose to use a neural network to automatically upgrade the original rough positions to bounding boxes. We found that a 160×160 subimage centered at a rough position is sufficient to completely contain any size scallop in our dataset (see the first row of Fig. 3 for several sample subimages), so the task of outputting exactly one bounding box for each subimage known to contain a scallop is equivalent to pure *localization* [22]. The difficult work of deciding *whether* there is an object at a particular AUV image location, and *what kind* of object it is, has already been done by human annotators, so we want to exploit this.

Directly regressing a bounding box (i.e., outputting coordinates) from an image is a highly non-linear and difficult-to-learn mapping [27, 15]. We were unsuccessful in our attempts to do it with a variety of convolutional front-ends, including Darknet-19, attached through several fully-connected layers to an (x, y, w, h) output layer. Rather, we borrow an idea from [15], which labels joint locations in images of people, and train the network to output a *scallop likelihood image*, or "heatmap", where high-value pixels belong to the scallop and low-value pixels to the background. In [15], at training time each ground truth joint position is indicated by a fixed-variance Gaussian (each joint has a separate heatmap) and the loss is the sum of squared differences. At test time, each joint's predicted heatmap is

post-processed to obtain the most likely joint location as the brightest pixel location. We want to generalize this to learn *masks* of scallop pixels, such as shown in row 2 of Fig. 3.

One way to think of the heatmap network is as a kind of *denoising auto-encoder* [8]. Input images can be regarded as clean scallop masks transformed and corrupted by appearance + lighting + background variation, plus artifacts introduced by the contrast enhancement stage. The network must learn to extract the essential information–the geometric attributes of the scallop–from which it can reconstruct the original mask minus any image "noise." Here we use a simple auto-encoder architecture consisting of an AlexNet CNN [11] encoder (5 convolutional and 3 max-pooling layers with ReLU activations), 2 fully-connected layers of 4096 units each, and a decoder network that inverts the AlexNet structure through a series of upsampling "deconvolutions."[1]

5. Training procedures

Both the YOLOv2 and heatmap networks were trained and tested using the `Darknet` open source neural network framework [16]. The `yolo.2.0.cfg` configuration file defines the YOLOv2 network architecture [18]; we modified this to change the number of filters of the last convolutional layer because the number of object categories to be detected is one ("healthy scallop") instead of the 20 classes of the original PASCAL VOC task. Unless noted, all other parameters such as learning rate, batch size, etc. were not changed, and training started from random weights.

5.1. Heatmap network

Following [15], we first train on fixed-radius circular masks ($r = 10$) to learn position only. Example subimages were taken from the **HighRes** training set (see Sec. 5.2 below), with each one randomly distorted geometrically and photometrically. This phase lasted 50K iterations with a learning rate of 10^{-4}. Then, because of the need for a scale estimate, there is a second phase of fine-tuning in which the targets are circles scaled to match the maximum dimension of each scallop, as shown in row 2 of Fig. 3. When a scallop is only partially visible at the edge of the image, such as in column 3 of Fig. 3, we draw the entire inferred scallop mask and background. This phase lasted 475K iterations with the learning rate dropped to 10^{-5}.

At test time, the scallop heatmap (examples of which are shown in row 3 of Fig. 3) is thresholded to make a binary mask and the bounding box of the largest connected component is output. The threshold was chosen to maximize the median overlap between the predicted and ground truth bounding boxes as calculated by the intersection over union (IoU) formula over the training set .

[1] Using Darknet-19 [18] as the encoder/decoder block did not fit in GPU memory, and VGG-16 [23] is similarly large

5.2. YOLOv2 detector network

We conducted four major experiments varying the training and structure of the YOLOv2 detector network. The first three experiments below used only the `M49` annotations, while the fourth incorporated data from `M6` and `M46`.

NoEdges In this experiment any AUV image with a scallop too close to an edge (i.e, the distance from any part of the bounding box to any edge is $\leq$ 10 pixels) was discarded. This was under the assumption that cropped bounding boxes would undermine the learning process by suggesting incorrect dimensions for partially visible scallops. After filtering, the dataset was made up of 1,664 images containing 3,436 scallops, or 81.6% of the original `M49` data. The images were split 80/20 for training/testing: 1,331 images (and all of their scallops) in the training set, and 333 images in the test set. The YOLOv2 network was trained for 10K iterations with a batch size of 64. The entire training process took 12.5 hours.

Edges Here all of the original scallops in the `M49` dataset were used without regard for image edge proximity. Again using an 80/20 split, the training set had 1,389 images with 3,121 scallops and the test set had 347 images with 742 scallops. We trained YOLOv2 with this approach for 17K iterations.

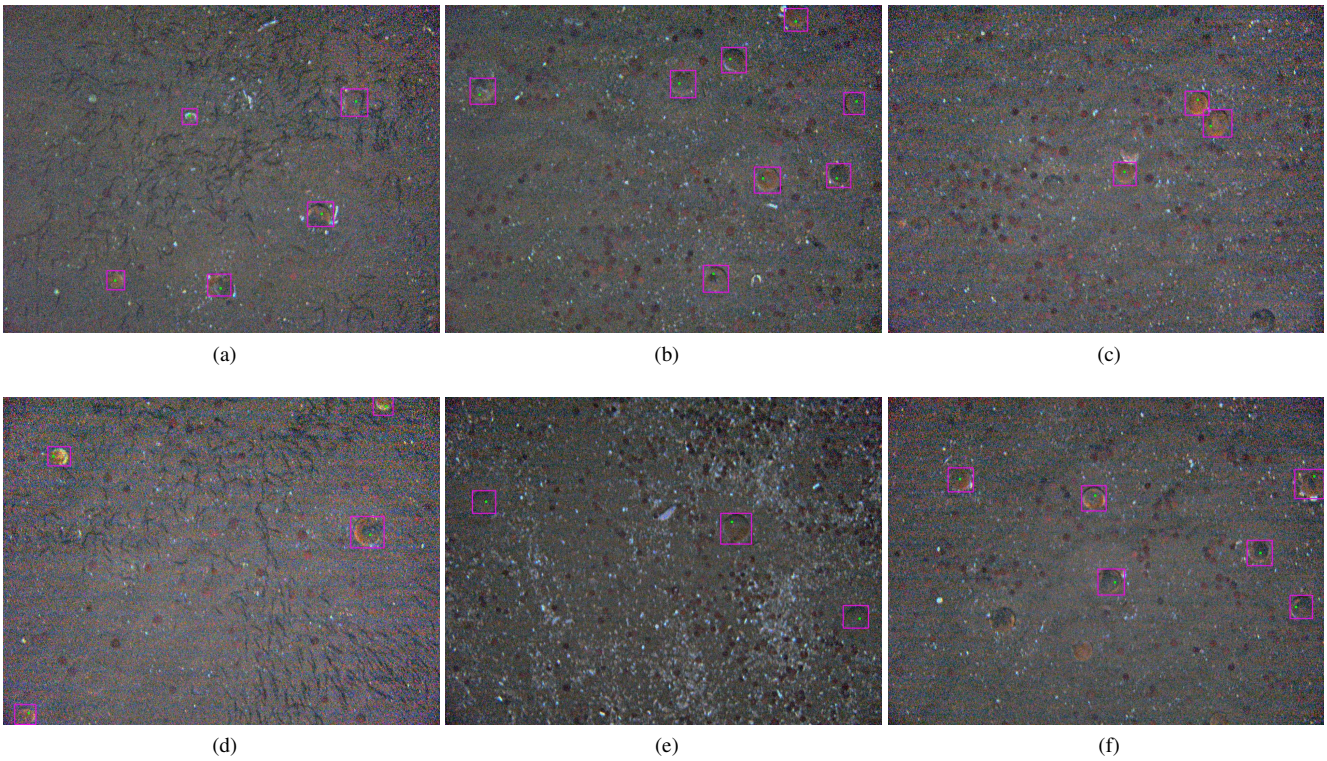
HighRes In order to better resolve small scallops, we doubled the width and height of the network's input layer to 832×832 to preserve image detail. This allows the network to "see" approximately 61% of the original AUV image information instead of only 14% as in the first two experiments. The total number of images and train/test split was the same as for the **Edges** network. The network was trained for 20K iterations.

AugmentedHighRes The `M49` manual bounding box annotations used to train **HighRes** were augmented with bounding boxes generated automatically by the heatmap network by upgrading all rough position annotations from `M6` and `M46`. This amounted to a total of 13,498 scallops from 7,061 images. YOLOv2 training was performed as fine-tuning from the **HighRes** 4K weights, continuing for another 10K iterations. This additional data constitutes more than $4\times$ the size of the **HighRes** training set alone.

6. Results

6.1. Heatmap

We evaluate the heatmap network's ability to accurately upgrade rough positions in a few ways. At the end of training, the mean IoU between the predicted and ground truth



(a) (b) (c)

(d) (e) (f)

Figure 4: Example annotation upgrade results from the heatmap network on M6 images. Green dots indicate the rough position annotations, pink boxes are the predicted bounding boxes. Because the original annotators did not mark them, the seemingly missed scallops in (f) are not upgraded.

bounding boxes on the M49 test set is 0.80, the median is 0.82, the min is 0.35, the max is 0.99, and 97.4% of upgrades are "correct" using the standard threshold of IoU ≥ 0.5. For comparison, on the training set itself the mean IoU is 0.90, median is 0.92, min is 0.36, max is 1.0, and 99.6% of upgrades are correct.

The predicted bounding boxes are robust to different backgrounds from other missions not seen during training. In particular, M6 was run in a completely different area of the Atlantic, and many of the images have features not seen in M46/49. A sample of the bounding boxes created for this mission are shown in Fig. 4. Another piece of evidence is that when the YOLOv2 detector is trained with these automatically-created bounding boxes in **AugmentedHighRes**, performance as measured by average precision improves modestly (see below).

6.2. Detection

Compiled with GPU acceleration and running on a machine with an NVIDIA Titan X GPU, each AUV image took from 70 to 100 ms to process for detections by the YOLOv2 network. Precision and recall for all detection experiments are plotted in Fig. 5.

The **NoEdges** network exhibits an average precision (AP) of **0.782**. To help assess how "good" this is, for comparison the easiest category for PASCAL VOC 2012 [5] according to the public leaderboard is *airplane*, for which YOLOv2 demonstrates the highest AP of 0.695. This makes our result quite competitive. However, it is harder to train on multiple classes rather than only one class as we do. Furthermore, when allowed to train with PASCAL as well as COCO [12] data, the best algorithm (not YOLOv2) reaches 0.95 on *airplane*. Also, it is clear that even though scallops near the image edges were not included in **NoEdges** training, the network is still capable of finding them. However, they are not in the test set and thus are counted as false positives. Including these objects during training and testing lifts the results for the **Edges** network, resulting in its AP reaching a maximum of **0.818** after 10K iterations.

For the **HighRes** network, the AP peaks after 4K iterations at **0.836**, a notable improvement over the lower-resolution versions of YOLOv2. The measured AP drops later in the training cycle, possibly due to overfitting. **AugmentedHighRes** lifts this score slightly, achieving an AP of **0.847** after 9K training iterations. We believe that there is considerable headroom to boost this number by including

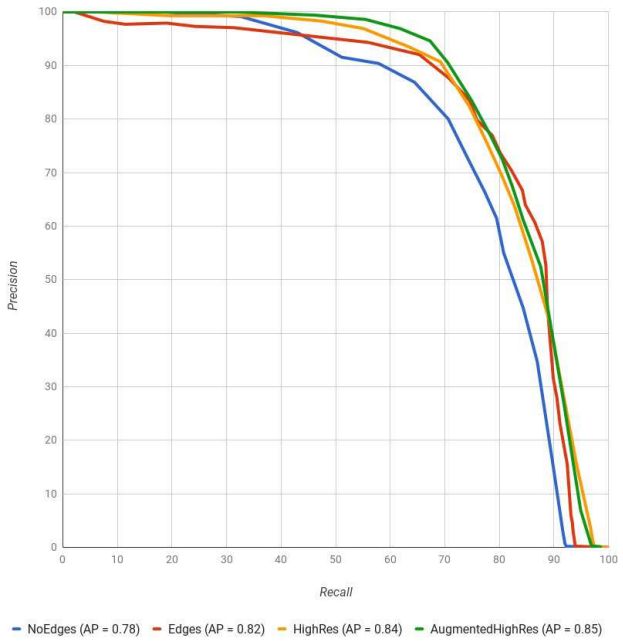


Figure 5: Precision-Recall (PR) curves of different YOLOv2 detection approaches

substantially more missions in the training set.

Fig. 6 shows some example detection results on the test set of the **HighRes** network. For these images, a confidence threshold of 0.2 was chosen, corresponding to a precision of 86% and recall of 54%. Green boxes are the ground truth, pink boxes denote correct detections by the network, and purple boxes are false positives. Numbers written next to each pink or purple box indicate how confident the network is that the object inside the box is indeed a healthy scallop.

After reviewing many of the detection images such as those shown in Fig. 6, we believe that the PR curves in Fig. 5 may be under-reporting performance due to some erroneous annotations. Some of the false positives may actually be healthy scallops that were overlooked in the initial annotation process–for example, the objects with a confidence score of 0.78 in Fig. 6(a) and a confidence score of 0.34 in Fig. 6(b). Conversely, in Fig. 6 (c) it is hard to argue that the small green box which counts as a false negative actually contains a scallop. Also causing false negatives, some scallops may be falsely marked as healthy when they are in fact compromised, as the green boxes in Figs. 6(e) and (f) would seem to show. Errors in the training set are not so critical, but certainly the test set should be cleaned carefully to better assess algorithms.

Validation on a separate mission We implemented a separate testing experiment for **HighRes**, which was trained on `M49`, on `M46` data. Images in this dataset are only annotated with rough position information, so we cannot compute IoU's in the standard manner. Instead, we see if the annotated rough position is simply *inside* a predicted scallop bounding box as the new criterion of correct detection. To do this, we changed the IoU threshold to 0: if the rough position is inside the prediction box, the IoU would be greater then zero. Otherwise, the IoU would equal to zero. Following this protocol, the network achieves an AP of 0.926, which is encouraging, but really just an upper bound on true performance.

Comparison to detectors from the literature [3, 10] used different datasets, so it is difficult to make a fair comparison with their results. [10] only gives two precision and recall data points which indicate a high number of false positives for their chosen parameters: on their "Dataset 1", they report 1% precision with a recall of 73%, and on "Dataset 2" they report 3% precision with a recall of 63%. From a glance at Fig. 5, our **HighRes** network reaches 85% and 93% precision at those recall levels, respectively.

[3] reports considerably more success, getting precisions ranging from 28% to 99% and recalls ranging from 69% to 94% with their "general" detector, depending on which subsets of data they tested on. Their results were improved slightly when first categorizing the substrate of the entire image and then using a specialized detector, which we do not do. Using a multicore processor, they are able to process the images at a rate of 10 Hz, just a little slower than this system. Despite the smaller size of scallops (due to the camera intrinsics) and lower contrast of our images, our results are still competitive.

7. Conclusion

In this report, we presented a deep convolutional neural network architecture for scallop detection. We used YOLOv2, a state-of-the-art convolutional neural network, for detecting small objects that only occupy a fraction of an entire image in very low contrast and cluttered underwater scenes. The system achieves high accuracy while running fast enough to keep up with the AUV's live image capture rate. Operating at this rate affords the prospect of motion control of the AUV in response to visually-detected changes in scallop densities, rather than following pre-programmed paths which are inefficient in time and sensor usage.

Our system can be used to improve the existing manual image annotation system by presenting "suggestions" to a human annotator in the form of detections at preselected confidence thresholds such that only a few scallops must be manually added per batch, or a few false positives must be removed. Furthermore, besides allowing automatic augmentation of the detector training set, the heatmap network can be used interactively by a human annotator to instantly

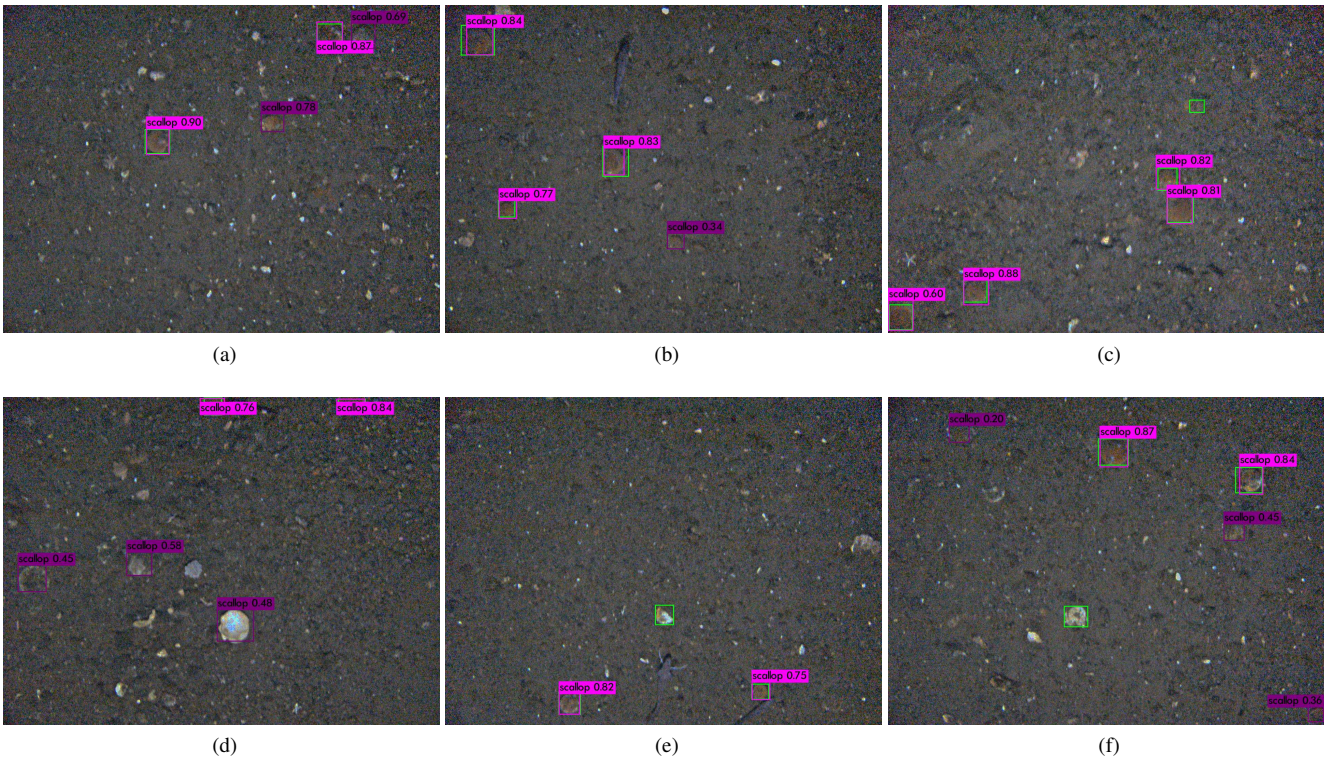


Figure 6: Example detection results from the **HighRes** network on the `M49` test set with detection confidence indicated (threshold = 0.2, corresponding to a precision of 54% and recall of 86%). Green boxes indicate ground truth, pink boxes are correct predictions, and purple boxes are false positives.

upgrade rough position clicks to bounding boxes as they work.

For future work, besides training the network on images for more missions, we plan to investigate adding multiple categories starting with compromised scallops, and moving on to other relevant creatures such as starfish, which prey on scallops. Following [3], it would likely be beneficial to train an ensemble of detectors for different benthic substrates, and for this we will need a whole-image terrain classifier for which the side-scan sonar could provide an additional input to the network. Furthermore, it is possible that the network could learn to perform contrast enhancement itself on the raw images.

Finally, besides the importance of detection accuracy on individual images, to achieve an accurate scallop count over a sequence of images taken from an AUV it is necessary to perform robust registration. This is necessary to avoid double-counting the same scallop if it appears in overlapping areas of successive images. We have preliminary results on this which we plan to incorporate into the final paper if accepted.

References

[1] J. Aguzzi, C. Costa, K. Robert, M. Matabos, F. Antonucci, S. Juniper, and P. Menesatti. Automated image analysis for the detection of benthic crustaceans and bacterial mat coverage using the venus undersea cabled network. *Sensors*, 11(11), 2011.

[2] M. Banko and E. Brill. Scaling to very very large corpora for natural language disambiguation. In *Proceedings of Association for Computational Linguistics*, 2001.

[3] M. Dawkins, C. Stewart, S. Gallager, and A. York. Automatic scallop detection in benthic environments. In *IEEE Workshop on Applications of Computer Vision*, 2013.

[4] K. Enomoto, M. Toda, and Y. Kuwahara. Scallop detection from sand-seabed images for fishery investigation. In *International Congress on Image and Signal Processing*, 2009.

[5] M. Everingham, L. V. Gool, C. Williams, J. Winn, and A. Zisserman. The PASCAL visual object classes (VOC) challenge. *International Journal of Computer Vision*, 2010.

[6] D. Ferraro. Estimating sea scallop incidental mortality from photogrammetric before-after-control-impact surveys. Master's thesis, University of Delaware, 2016.

[7] Y. Freund and R. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In *Computational Learning Theory*, 1995.

[8] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016.

[9] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[10] P. Kannappan, J. Walker, A. Trembanis, and H. Tanner. Identifying sea scallops from benthic camera images. *Journal of Limnology and Oceanography*, 2014.

[11] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Conference on Neural Information Processing Systems*, 2012.

[12] T. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollar, and C. Zitnick. Microsoft COCO: Common objects in context. In *European Conference on Computer Vision*, 2014.

[13] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C. Fu, and A. Berg. SSD: Single shot multibox detector. In *European Conference on Computer Vision*, 2016.

[14] F. Pereira, P. Norvig, and A. Halev. The unreasonable effectiveness of data. In *IEEE Intelligent Systems*, 2009.

[15] T. Pfister, J. Charles, and A. Zisserman. Flowing convnets for human pose estimation in videos. In *IEEE International Conference on Computer Vision*, 2015.

[16] J. Redmon. Darknet neural network framework. Downloaded spring, 2017 from `https://github.com/pjreddie/darknet`.

[17] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[18] J. Redmon and A. Farhadi. YOLO9000: Better, Faster, Stronger. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[19] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Conference on Neural Information Processing Systems*, 2015.

[20] G. Rosenkranz, S. Gallager, R. Shepard, and M. Blakeslee. Development of a high-speed, megapixel benthic imaging system for coastal fisheries research in alaska. *Fisheries Research*, 92(2), 2008.

[21] R. E. Schapire and Y. Singer. Improved boosting algorithms using confidence-rated predictions. *Machine Learning*, 37(3), 1999.

[22] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. In *International Conference on Learning Representations*, 2014.

[23] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015.

[24] A. Stewart-Oaten, W. Murdoch, and K. Parker. Environmental impact assessment: "pseudoreplication" in time? *Ecology*, 67(4):929–940, 1986.

[25] C. Sun, A. Shrivastava, S. Singh, and A. Gupta. Revisiting unreasonable effectiveness of data in deep learning era. In *IEEE International Conference on Computer Vision*, 2017.

[26] R. Taylor, N. Vine, A. York, S. Lerner, D. Hart, J. Howland, L. Prasad, L. Mayer, and S. Gallager. Evolution of a benthic imaging system from a towed camera to an automated habitat characterization system. In *IEEE OCEANS Conference*, 2008.

[27] J. Tompson, A. Jain, Y. LeCun, and C. Bregler. Joint training of a convolutional network and a graphical model for human pose estimation. In *Conference on Neural Information Processing Systems*, 2014.

[28] J. Uijlings, K. van de Sande, T. Gevers, and A. Smeulders. Selective search for object recognition. *International Journal of Computer Vision*, 104(2):154–171, 2013.

[29] C. Zitnick and P. Dollár. Edge boxes: Locating object proposals from edges. In *European Conference on Computer Vision (ECCV)*, pages 391–405. Springer, 2014.