

Automatic Scallop Detection in Benthic Environments

Matthew Dawkins Charles Stewart
Dept. of Computer Science, RPI
Troy, NY 12180 USA

matthew.d.dawkins@gmail.com, stewart@cs.rpi.edu

Scott Gallager Amber York
Woods Hole Oceanographic Institution
Woods Hole, MA 02543 USA

sgallager@whoi.edu, adyork@whoi.edu

Abstract

As a multi-billion dollar industry, scallop fisheries world-wide rely on maintaining healthy off-shore populations. Recent developments in the collection of optical images from extended areas of the ocean floor has opened the possibility of assessing scallop populations from imagery. The sheer volume of data — upwards of 20,000 images per hour — implies that automatic image analysis is necessary. This paper presents a computer vision software system to identify and count scallops. For each image, the system generates initial candidate regions of potential scallops, extracts image features in the candidate regions, and then applies one of several different trained AdaBoost classifiers to determine the strength of each region as a scallop. In making the final classification decision, the strength of the scallop classifier output is compared to the output of other classifiers trained to detect sand dollars, clams and other “distractors”.

1. Introduction

Recent advances in hardware, software and camera systems, together with the growing need to protect indigenous species, have stimulated development of new methods for environmental monitoring via imagery and video [10, 11, 17]. The protection of scallop populations is of particular importance to regional economies around the world, including parts of Japan, China, the United States, and Canada. In order to prevent over-harvesting, fisheries may be restricted to certain quotas or confined to operating in certain regions. The traditional method of sampling the scallop biomass and thereby deciding which areas should be open for harvest is based on dredging. The recent and growing availability of optical images of the ocean floor raises the possibility of replacing dredging with computer analysis of optical imagery to detect and count scallops, and determine their biomass. Developing the core scallop detection method is the goal of this paper.

Images are collected by the Habitat Mapping Camera

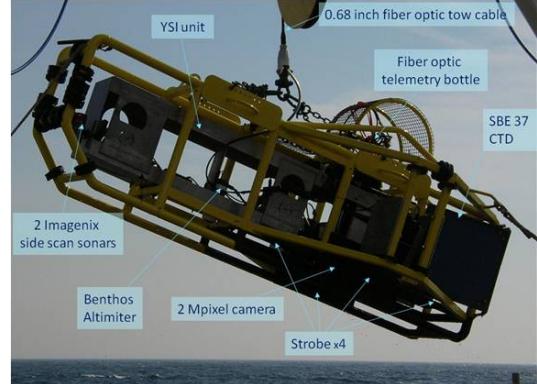


Figure 1. The Habitat Mapping Camera System (HabCam) II.

System, HabCam II, developed and operated by the Woods Hole Oceanographic Institution and members of the Hab-Cam Group (Figure 1), which is towed by a fishing trawler at heights of 1-3m above the ocean floor. The optical sub-system of the HabCam II includes a digital camera taking 1280 by 1024 resolution images at 5-6 frames per second. The camera is surrounded by 4 strobe lights to provide artificial lighting as the probe frequently operates at depths exceeding 50 meters.

1.1. Challenges for Scallop Detection

A successful scallop detector must address several challenges, as illustrated in Figures 2 and 3. In particular:

- The strobe lighting system produces images having a non-uniform illumination, while the attenuation of light in the seawater results in weakly-colored images. Moreover, water conditions may sometimes reduce the image contrast to a level where individual scallops lack clearly defined boundaries.
- Scallops may be white, brown or some combination thereof, and they have a wide range of textures. Less commonly, their shells can take on other hues.
- Scallops may be partially buried by sediment or occluded by other scallops or different organisms. Typ-

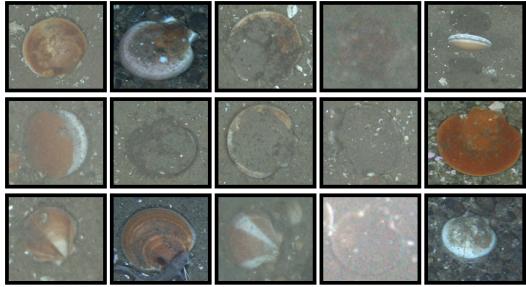


Figure 2. Examples of live scallop appearances. As shown, scallops can appear with a wide variety of color and texture patterns. They are often highly occluded by other organisms and sediment.



Figure 3. Related objects with similar appearances and sizes when compared to scallops. From left to right, a sand dollar, 2 clams, an urchin, and a brown rock.

ically, a buried scallop will still produce a visible boundary because its filtration will reduce the amount of sediment from the top and the sides of its shell.

- While the typical scallop shape is circular with a hinge on one side, swimming scallops are often seen edge-on, and partially buried scallops may show no hinge.
- Shells of dead scallops look like live scallops. Dead scallops may be identifiable by cracks in their shell, by atypical coloring, or by sediment appearing in the inner shell.
- Many objects have similar appearance to scallops, including sand dollars, clams, urchins and even rocks (Figure 3). Sand dollars and urchins often have radial symmetric textured patterns, while scallops have ring patterns originating at the hinge.
- Scallops commonly appear in a variety of benthic environments, including sand, rock and mud.

1.2. Prior Work

Several previous papers have described methods for scallop detection and counting. The published techniques are generally tailored for constrained sets of data. For example, Enomoto, et al. [6] extract features to describe the fluted pattern particular to bay scallops. Most of our data shows Atlantic sea scallops, and we must address a wider range of appearance variations. The work of [7] is specialized to detecting scallops in a sandy environment. The authors use a Hough transform to identify ellipses in grayscale images, which are then filtered based on nearby grayscale intensities close to the detected ellipses. The architecture of

this approach is similar to that of ours, but the challenges imposed by our data imply that more sophisticated interest point detection and image feature measurements are necessary. Gabor wavelets, texture energy gradients, and an SVM classifier are used in [9, 17] to segment and identify a variety of organisms, including scallops, but these studies were troubled by high false positive identifications for scallops.

The vast computer vision literature on object recognition can be roughly divided into the problems of (a) instance recognition, where the same object (e.g. a building) is recognized in multiple images [1, 13, 14], (b) category recognition, where the image is labeled according to its dominant category (horse, car, wagon, etc.) [3, 20], and (c) object detection, where examples of a certain object, often a face, are found by scanning the image and testing each location for the presence of the object of interest [19, 4]. Our work sits solidly in this latter category.

1.3. System Overview

Although our approach is primarily based on the object identification paradigm, like a number of recent segmentation algorithms (e.g. [2]), our technique generates initial candidate identifications of potential scallops (and distractor objects) in each image, and then tests these candidates using a sophisticated set of image measurements and trained classifiers. Our system includes the following components:

1. *Illumination and color correction* is applied to each image. A substrate category (sand, gravel, mud, etc.) is also optionally selected by the user to aid in classifier selection.
2. *Initial image filtering and histogram formation*: Several transformations are applied to each input image, some as simple as gray scale or color space mappings. More involved transformations use gathered statistics on color frequencies on a per image basis and on a per object (white scallop, brown scallop, sand dollar, etc) basis. These are used to generate new images that reflect the likelihood of a particular color occurring within an image and to generate empirical probabilities of a particular color arising from a particular object.
3. *Candidate region detection*: Four different interest operators are applied to the filtered images to generate candidate locations and associated regions that might be a scallop or might be one of the distractors. Each detector is introduced to handle particularly challenging cases of scallop appearance.
4. *Feature extraction*: A variety of color, texture and edge features are extracted for each candidate region.
5. *Classifiers*: A series of cascaded Adaboost classifiers are applied to the feature vector extracted for a can-

dicate region. Each classifier is specialized to a particular object — white scallop, brown scallop, buried scallop, dead scallop, sand dollar, clam, etc. — and sometimes to a particular substrate. The outputs of the classifiers are combined into a final classifier that determines the final label for the region.

These methods are described in the following sections.

2. Illumination and Color Correction

Each image is corrected on a pixel-by-pixel basis by dividing each intensity by the learned long-term average for that particular pixel location, color channel, and HabCam sled height (obtained from metadata).¹ Interpolation is applied between heights to obtain accuracy. All subsequent processing is applied to these corrected images.

3. Initial Image Transformations

Multiple transformations are applied to the illumination and color-corrected input RGB image in order to create a series of images used by one or more later operations. This series includes:

- The RGB image itself, I_{rgb} , together with its grayscale version, I_{gs} , and its mapping to L*a*b, I_{L*a*b} .
- The magnitude, I_{mag} , and direction, I_{dir} of the gradients of I_{gs} , computed using the Sobel operator.
- A “color commonality” image, I_{cc} , where each pixel’s color is replaced by the likelihood of the color across the entire image.
- A conditional probability image, $I_{cp;o}$, for each object of interest o (brown scallop, white scallop, sand dollar, etc.) giving the probability of a pixel’s color conditioned on the pixel being from object o .

Color commonality image, I_{cc} , is formed to address problems in contrast in underwater images (see Figure 4). An $N \times N \times N$ color histogram (typically, we use $N = 64$ for the 12-bit Habcam images), H_{rgb} , is formed from image I_{rgb} and then smoothed with a 3-dimensional Gaussian ($\sigma = 1.0$ bin). Pixel value $I_{cc}(\mathbf{x})$ is entry $H_{rgb}(I_{rgb}(\mathbf{x}))$ in this histogram. Note that this is applied to each pixel separately. Despite the non-uniformity of its space, RGB works as well as L*a*b, so we use it for simplicity.

One instance of conditional probability image, $I_{cp;o}$, is computed separately for each object, o , to be detected. A color histogram, H_o , is initialized from the pixels in the regions corresponding to object o in the manually-labeled training images. This histogram is updated on-line based

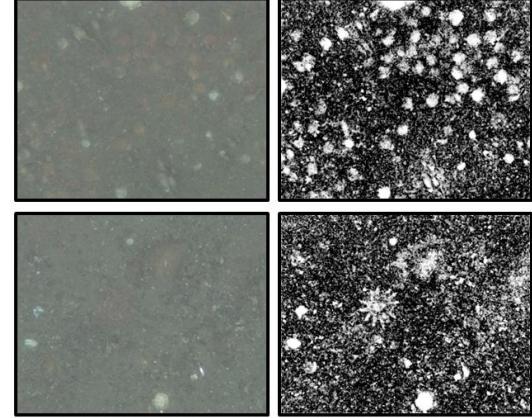


Figure 4. Example color commonality images (I_{cc}) are shown on the right. In this diagram, brighter intensities in I_{cc} correspond to less frequent colors in the original image. Because the color of the background is relatively consistent, foreground objects become more salient.

on detections gathered when running the software on a Hab-Cam image sequence. For the update, if at least one instance of object, o , (a white scallop, for example) is detected in a new image, then that image’s pixels from all detected white scallops are used to form a new $N \times N \times N$ histogram, which is then merged into the existing H_o with a given weight (0.03). The resultant histogram is renormalized to have a cumulative weight of 1. A histogram, H_{bg} is formed similarly for the background from the training data and on-line image regions where none of the objects are detected. The value given to each individual pixel in $I_{cp;o}(\mathbf{x})$ is the difference between the object and background histograms, i.e.

$$I_{cp;o}(\mathbf{x}) = H_o(I_{rgb}(\mathbf{x})) - H_{bg}(I_{rgb}(\mathbf{x}))$$

Note that when this value is 0 a pixel is equally likely, based on its color alone, to be object (e.g. white scallop) or background. Finally, we aggregate over all objects of consideration o to generate the final image:

$$I_{cp}(\mathbf{x}) = \max_o I_{cp;o}(\mathbf{x})$$

4. Candidate Detection

After computing the initial image transformations, four different techniques are applied to detect candidate object regions. Each candidate is represented by a 5-parameter ellipse specifying the image location, orientation and axes. While these techniques seem somewhat redundant, the experimental results presented below clearly show the improved results obtained when using all four.

4.1. Difference of Gaussian Peaks

The first detector extracts difference-of-Gaussian peaks in image position and in scale space, following the tech-

¹We thank Jason Rock and Peter Honig for their design, implementation and refinement of this algorithm.

nique described in the original SIFT paper [12]. If σ is the detection scale, then the initial ellipse for this candidate is a circle of radius $\sqrt{2}\sigma$. The range of scales to search is established by physical considerations using known scallop sizes, camera calibration parameters, and metadata that records the height of the camera above the ocean floor. Early in a HabCam collection sequence, this “blob detection” is applied to the generic color commonality image I_{cc} , but once I_{cp} has been become sufficiently stable — e.g. after $n = 10$ images with detections of any objects of interest — the computation switches to use I_{cp} . The blob detection method does well at detecting the standard general case of objects (scallops) where the entire shell is showing, but also does well with cases that may lack clearly defined edges due to water turbidity. Example results are shown in Figure 5.

4.2. Adaptive Thresholding

Adaptive thresholding is similarly applied to I_{cp} , starting with a threshold of 0 — as negative values indicate a pixel is more likely to be background. An ellipse is fit to each connected region in the thresholded image, and any ellipse whose dimensions fall within the acceptable ranges established by physical considerations is added to the list of candidates. If there are too many small candidates, the threshold is lowered by a certain percentile and the process repeated. Conversely, if any candidate is significantly larger than our upper object scanning size, the threshold is raised. Example results are shown in Figure 6.

4.3. Template Matching

At each pixel location, \mathbf{x} , a set of 16 one-dimensional derivatives is computed at locations $\mathbf{x} + R(\cos \theta_i, \sin \theta_i)^\top$, for directions $\theta_i = 2\pi i/16$, $i \in \{0, \dots, 15\}$ and circle radius R (Fig. 7). The direction of the i^{th} derivative is $(\cos \theta_i, \sin \theta_i)^\top$. A derivative is computed for each channel of I_{L*a*b} and the results are combined at each location

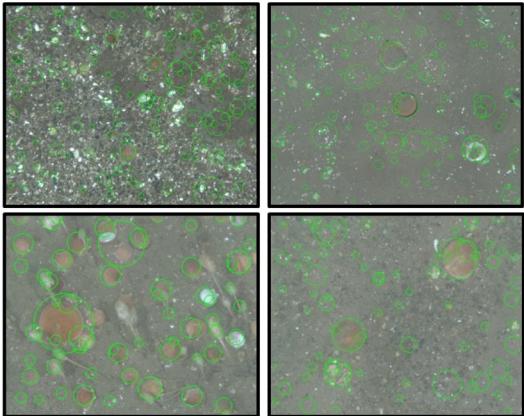


Figure 5. Difference of Gaussian candidate points on I_{cp} .

by taking the L_2 norm. In turn, these are combined at the central pixel \mathbf{x} by averaging all 16 responses and dividing by the maximum, which emphasizes consistency and ensures that one derivative does not dominate. This is applied at a fine sampling of R values — seven per octave — but the computation is fast because we can precompute 16 1D derivative images. As before, physical parameters determine the range of scales. Only a limited number of extrema are reported as candidate points, by performing a windowing across the image and taking the top 5 candidates in each window (see Fig. 8).

4.4. Edge Detection

Lastly, to provide another layer of redundancy and to better handle cases near image borders, a Canny edge detector is applied (Fig. 9), connected chains are extracted, and an ellipse is fit to a downsampled version of each chain. Results are scored by the mean-square error of the chain edge pixels to the ellipse. The top 40 results in the image are reported as potential candidates.

4.5. Consolidation and Prioritization

These methods all have the potential to detect the same object multiple times, so we need to apply a consolidation step. Candidates are merged when their centers are within 10% of the average of their major axis lengths, when their axis lengths are both within a 20% margin of each other, and when their orientation difference is less than 30° . Candidates are then ranked in a single priority queue based first on whether or not the candidate was detected by multiple detectors, followed by interleaving the top results from each detector based on their respective detection magnitudes. This ordering allows us to set a limit on total processing time by controlling the final number of candidates.

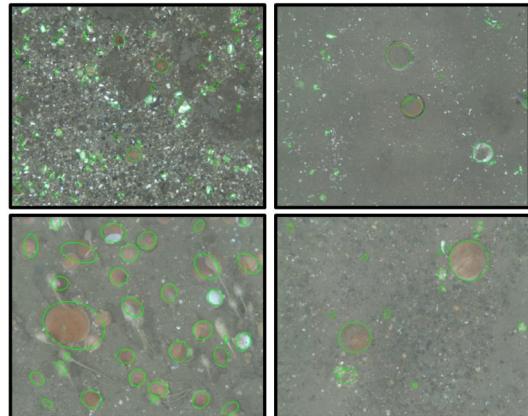


Figure 6. Adaptive thresholding candidate points.

Feature Set	Number of Features	Description
Edge-Based Properties	137	Color gradients and measures of smoothness around an estimated candidate contour
Candidate Point Properties	9	Ellipticity, size of axis in real units, ratio of axis lengths.
Histogram of Oriented Gradients	3528	HoG descriptors calculated on I_{gs} and “saliency” image I_{cc} around each candidate
Raw Color-Based Features	122	Properties derived from L*a*b, RGB and I_{cp} extracted from 32 different regions around candidates
Gabor Filtering	30	Smoothed output of various Gabor filters applied to different regions around candidates

Table 1. Overview of extracted features. Not every feature is always used in the resultant learned classifiers.

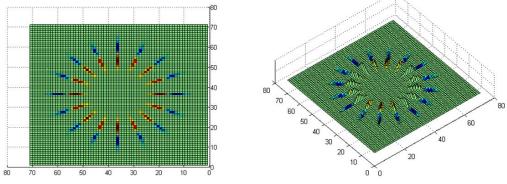


Figure 7. The 16 derivative locations and directions for template matching.

5. Features Extracted for Candidate Points

The rank ordered candidate points must be classified as one of several types of scallops (brown, white, buried, dead), distractors (sand dollars, clams, etc.), or background. In order to do this, a vector of features is computed for each region as input to a set of classifiers. These features are generic in the sense that they are not specific to a particular object type, making the feature computation — the most expensive operation overall — independent of the number of classifiers. The features forming this 3832 component vector are summarized in Table 1. They are designed to capture shape, color and texture properties, and are computed as follows.

The first set of measures is computed from a piecewise — and potentially open — contour extracted near the region boundary. To compute this contour, each pixel within a small distance of the elliptical candidate region boundary is assigned a weight based on (a) the distance to the bound-

ary, (b) the angle between the intensity gradient and the direction to the region center, and (c) the gradient magnitude. Local maxima with the highest weights in each radial direction from the region center are then selected, and the resulting “edge” points are linked into one or more chains. Each chain is weighted as the sum of the edge weights that form it. A partial contour is created for the candidate by selecting the linked chains with the highest weights to be a part of this contour, stopping the selection process when there is at least 1 edge pixel in each of 4 quadrants surrounding the candidate point (Figure 10).

This partial contour is used as the basis for extracting the following boundary contour features: (a) the mean-squared error between the contour points and an ellipse fit to the points, (b) the average color in the region near the contour boundary, and (c) the average color first and second derivatives in the direction from the two highest-weighted edgel chains toward the center of the candidate region. To compute (c), each pixel in the entire contour is shifted a distance of 1 pixel towards the center of the candidate, the average L*a*b color for the shifted contour is calculated, and the process repeated for several iterations. Sampled fairly densely, the difference between the average colors (computed at each iteration) are reported as the primary color gradient features.

The second feature set includes nine properties of the elliptical region, converted to physical units (meters) using the camera and sled metadata. This includes the axis lengths and their ratio, the ellipse perimeter and area, and

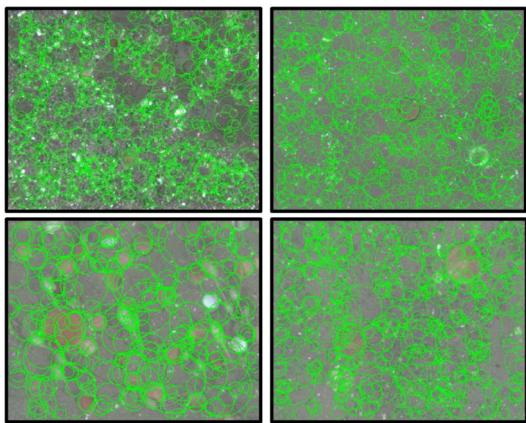


Figure 8. Template approximation candidate points from I_{L*a*b} .

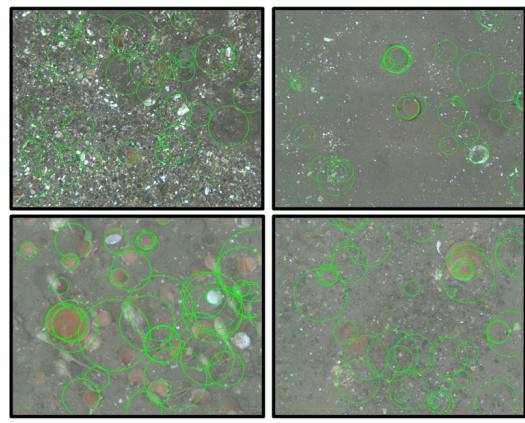


Figure 9. Edgel candidate points from I_{gs} .

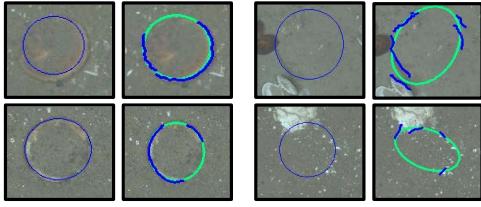


Figure 10. Examples of extracted edges for valid scallop candidate points (left) and non-candidate points (right). The left sub-columns show the original candidate point, and the right the estimated contour (blue) and estimated ellipse (green)

the percentage change between the initial candidate point axis lengths and the revised estimation from the prior step.

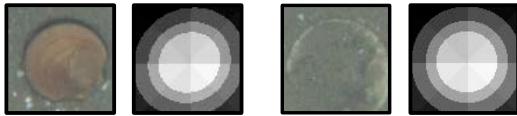


Figure 11. The 32 regions (16 inner, 16 outer) used for the raw color descriptor centered on each candidate.

Third, a 32-bin color descriptor is formulated as shown in Figure 11 comprised of 16 inner bins (within the candidate boundary) and 16 outer bins. The inner and outer groups each contain two concentric rings of eight bins, and four values are computed for each bin: the three-component average L^*a^*b value and the percentage of pixels with a positive I_{cp} value. 64 of these values (the L^*a^*b and I_{cp} averages of the inner bins) are reported directly as part of the size 122 color descriptor. The remaining 58 features are computed from various aggregates across multiple bins including: the difference in values of all of the inner boundary regions with those of their direct (outer) neighbors, averages of the values of all inner regions combined, and the difference in values of the combined inner region average with a combined average over all outer regions.

Fourth, in order to extract both textural and shape information, two Histogram of Oriented Gradient descriptors are created in the style of [4], centered on each individual candidate. The 1764-bin descriptor is computed once for the grayscale input I_{gs} and again for the color commonality image I_{cc} . The latter is most important for input images with very low contrast. Overall, the HoG descriptors aid in classifying the scallop circular shape and in differentiating between the texture patterns on different organisms (e.g. scallops vs. sand dollars).

Finally, to complement the textural cues of the HoG descriptors, the responses of an array of multiple scale-normalized Gabor filters are measured at different points around the center of each candidate at 5 locations (at the

candidate center and the 4 corners of an oriented cross pattern extending halfway towards the candidate boundary from the center of the candidate). Only the real magnitude of the Gabor filter outputs were used as features. The exact selection of Gabor filters was performed via manual tuning.

6. Classifiers and Training

Following computation of the features in a candidate region, a decision is made about whether the region is a scallop (white, brown, dead), a sand dollar, a clam, another distractor object, or nothing at all. This decision is made by combining the results of a set of AdaBoost classifiers [8, 15] running on two levels. The first level, designed to eliminate as many false positives as possible without introducing many false negatives, contains a one-versus-all classifier for each object of interest. Candidates that have a positive response for one of the object classifiers are fed into the second stage. The second stage contains additional classifiers, including both one-vs-all classifiers and one-vs-one classifiers for comparing particular objects against each other (e.g. white scallop vs. sand dollar). The strongest object response among these is converted to a probability and then thresholded.

Each individual classifier in the entire system was trained via Real AdaBoosting [16, 18], for up to a set number of iterations on top of decision stumps or short decision trees of fixed length. In order to focus the training, the output of the candidate detectors was manually annotated as to both object category and localization of the region on the object. Poorly-localized object regions were not used as positive instances during training.

Following the two level classification, if any remaining candidates overlap in area by more than 60%, the one with the lowest classification value is eliminated. Additionally, to leverage the fact that sand dollars and scallops often occur in clusters, if we last detected an image that contained

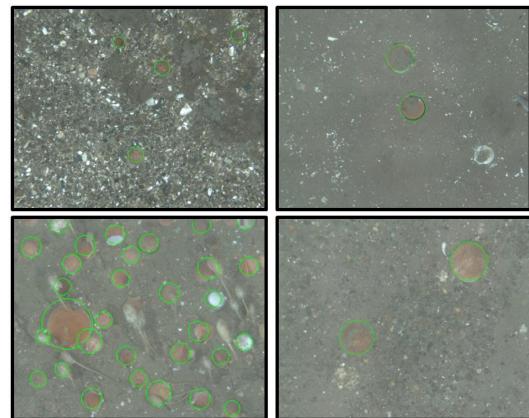


Figure 12. Examples of final detections, after all filtering.

Data Set	Classifier Type	Detection Rate	Precision	Counting Error
20080628_0310	General	94.15	98.77	-4.68%
	Specialized	94.19	94.74	-0.58%
20090629_0000	General	69.14	96.28	-28.19%
	Specialized	88.40	99.07	-9.12%
20090629_0130	General	87.07	92.42	-5.78%
	Specialized	92.04	90.29	+2.34%
20090316_1650	General	78.26	90.00	-13.04%
	Specialized	88.29	98.80	-10.64%
20081111_2040	General	84.21	28.07	+200.00%
	Specialized	60.87	66.67	-8.70%
20100821_0220	General	85.71	88.00	-2.60%
	Specialized	94.64	86.89	+8.93%

Table 2. Results on a number of different cruises with a general-purpose scallop detector and a scallop detector specialized to the benthic substrate that is pre-dominant in an image.

Candidate Point Detectors	Detection Rate	Precision	Counting Error
Difference of Gaussian (DoG)	68.68	98.43	-30.60%
Filtered Canny Edges (FCE)	37.36	98.55	-62.09%
Template Approximation (TA)	73.22	98.53	-25.68%
Adaptive Threshold (AT)	60.65	99.10	-38.80%
DoG + TA	80.43	99.33	-18.49%
FCE + AT	62.50	98.29	-36.41%
DoG + FCE + TA + AT	85.56	98.71	-15.21%

Table 3. Combined results on 20080628_0310 and 20090629_0000 with varied candidate point detectors enabled.

a large number of sand dollars, then the sand dollar classification value is scaled by a fixed factor. Differentiating between buried sand dollars and buried scallops is one of the most difficult sub-problems of scallop detection, and is why special emphasis is placed on detecting sand dollars.

7. Experimental Results and Discussion

The detection algorithm was applied to image sets collected from HabCam cruises over 2008-2010. These were in different locations up and down the Atlantic seaboard. In order to explore several aspects of this design, we have gathered a variety of experimental results from these data sets. We focus here primarily on the effectiveness of the algorithm on scallops. For more details see [5].

The overall results are summarized in Table 2. Each individual image set, represented by two rows in the table, is labeled by its date of collection. Each typically contains several thousand images (for these experiments). Rows labeled “General” show the detection rate (P_d), precision, and the percentage counting error for the described algorithm across the entire data set. Rows labeled “Specialized” show the results when the described algorithm is further specialized, in both training and testing, to the particular substrate seen in each image — rocks, sand, mud, gravel, boulder, in addition to image quality (sharp vs cloudy). Currently this specialized labeling is done manually, but soon it will be done automatically. All numbers are obtained by comparing the results to those of a human annotator.

The results vary dramatically across the different cruises. This occurs for a variety of reasons. Data set

20080628_0310 shows excellent results for both the general and specialized algorithms, while 20090629_0130 and 20100821_0220 show good results in general and improvements into the mid 90’s for the specialized algorithm. On the other hand, data set 20081111_2040 exhibited a very high false positive rate. The images in the data set include very few scallops and a high number of distractor objects such as rocks and shells. In addition, the image quality was very low. Images from set 20090629_0000 were taken at a much higher altitude — typically 3-4m about the ocean floor rather than 1-2m, producing much smaller scallop regions in the images (see Figure 13). It also has many occlusions and a very high density of scallops.

Tables 3 and 4 explore various features of the algorithm design. In particular, Table 3 shows the results on two image sets with different candidate detectors enabled, while keeping everything else in the system constant. In this experiment the general classifiers were used. Clearly, the combination of the four detectors makes a significant difference. We attribute this to both the low contrast of scallops and their appearance variations across images. Table 4 details the false positive rate per meters squared on a dataset rich with distractor objects. Results are shown with classifiers trained on both each individual feature set, and their union. Their aggregate showed the highest discriminatory power of the resultant classifiers.

Finally, we briefly consider the computation time involved in our scallop detection algorithm. Not surprisingly the feature extraction step requires more than 50% of the total time. Since, on a Intel Quad-Core i7 processor with 4 Gb of RAM, the entire computation requires about 800 milliseconds per image on a single core, when using all cores,

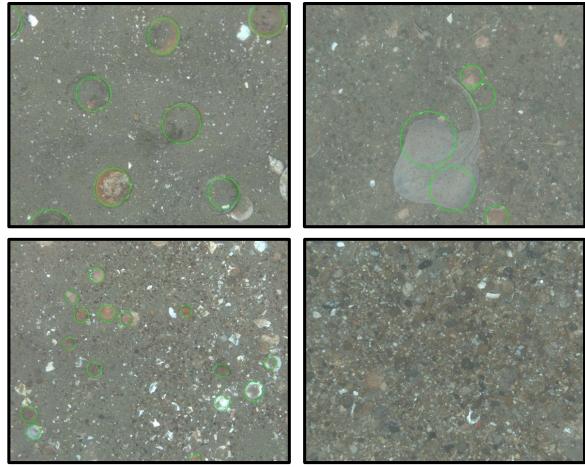


Figure 13. Mixed good (left) and poor (right) results from multiple environments. Shown from left to right, top to bottom are gravel (20090316_1650), mud (20100821_0220), shell (20090629_0000), and rocky (20081111_2040) environments.

Feature Set	Detection Rate	Precision	False Positives (per m^2)
Edge-Based Features	77.89	61.16	2.887
Candidate Point Properties	61.54	62.50	1.474
Dual HoGs	76.71	75.68	1.105
Color-Based Features	75.75	40.32	2.273
Gabor Filtering	24.18	73.34	0.491
All Features	85.56	98.71	0.122

Table 4. False positive rates on 20080628_0310 and 20090629_0000, with classifiers trained with different feature sets enabled.

we obtain 10Hz throughput, allowing scallop detection to keep up with the HabCam data collection rate.

8. Conclusions and Future Directions

We have presented a scallop detection system that is based on a pipeline of initial image analysis, candidate extraction, feature computation, and a two-level final classifier. Effectively running at frame rate, it has shown strong initial results on very challenging imagery. On reasonably good quality data, its detection rate is in the mid 90s with few false detections. Its performance drops on poorer quality data and data with very few scallops, where we often run into previously unknown distractor categories.

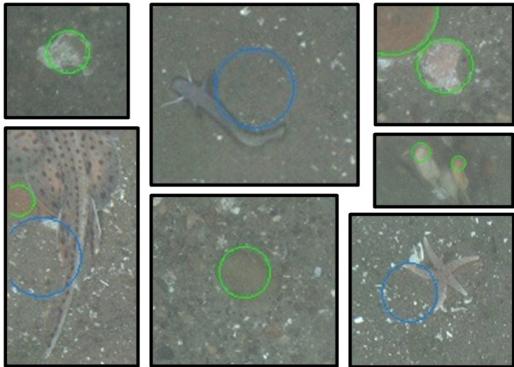


Figure 14. Examples of false positives where the boundary of the “scallop” is formed by other objects.

There are four major directions for further work. First is a more detailed analysis of the algorithm configuration and role of each processing step. Second is integration with automatic substrate detection, allowing application of specially-trained classifiers following the initial candidate detection stage. Third is to remove false positives through more comprehensive understanding of each image, eliminating errors caused in part by the appearance of nearby objects (see Figure 14). This is quite important because both the boundaries and interiors of buried scallops can be subtle. Fourth is the evaluation of the system in the broader context of trying to make fisheries policy decisions more effectively and less invasive.

References

- [1] R. Arandjelovic and A. Zisserman. Three things everyone should know to improve object retrieval. In *CVPR*, 2012.
- [2] J. Carreira, F. Li, and C. Sminchisescu. Object recognition by sequential figure-ground ranking. *IJCV*, 98(3), 2012.
- [3] G. Csurka, C. R. Dance, L. Fan, J. Willamowski, and C. Bray. Visual categorization with bags of keypoints. In *ECCV Workshop on Statistical Learning*, 2004.
- [4] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005.
- [5] M. Dawkins and C. Stewart. Scallop detection in multiple maritime environments. *Rensselaer Polytechnic Institute, Digital Collections*, 2011.
- [6] K. Enomoto, T. Masashi, Y. Kuwahara, H. Fisheries, W. Masaaki, and K. Hatanaka. Scallop detection from gravel-seabed images for fishery investigation. In *MVA, IAPR*, 2009.
- [7] K. Enomoto, M. Toda, and Y. Kuwahara. Scallop detection from sand-seabed images for fishery investigation. In *2nd Int. Congress on Image and Signal Processing*, pages 1–5. IEEE, 2009.
- [8] Y. Freund and R. Schapire. A desicion-theoretic generalization of on-line learning and an application to boosting. In *Computational Learning Theory*, pages 23–37, 1995.
- [9] S. Gallager et al. High resolution underwater imaging and image processing for identifying essential fish habitat. In *Report of the National Marine Fisheries Service Workshop on Underwater Video Analysis*, pages 44–54, 2005.
- [10] R. Holman, J. Stanley, and T. Ozkan-Haller. Applying video sensor networks to nearshore environment monitoring. *IEEE Pervasive Computing*, 2(4):14–21, 2003.
- [11] D. King, R. DeGraaf, P. Champlin, and T. Champlin. A new method for wireless video monitoring of bird nests. *Wildlife Society Bulletin*, pages 349–353, 2001.
- [12] D. Lowe. Distinctive image features from scale-invariant keypoints. volume 60, 2004.
- [13] D. Nistr and H. Stewnius. Scalable recognition with a vocabulary tree. In *CVPR*, pages 2161–2168, 2006.
- [14] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *CVPR*, 2007.
- [15] G. Ridgeway. The state of boosting. *Computing Science and Statistics*, pages 172–181, 1999.
- [16] R. Schapire and Y. Singer. Improved boosting algorithms using confidence-rated predictions. *Machine Learning*, 37(3):297–336, 1999.
- [17] R. Taylor et al. Evolution of a benthic imaging system from a towed camera to an automated habitat characterization system. In *OCEANS*. IEEE, 2008.
- [18] A. Vezhnevets. GML Adaboost matlab toolbox. *Technique Manual, Graphics and Media Laboratory, Computer Science Department, Moscow State University, Moscow, Russian Federation*, 2006.
- [19] P. Viola and M. Jones. Robust real-time face detection. *IJCV*, 57(2), 2004.
- [20] J. Winn, A. Criminisi, and T. Minka. Object categorization by learned universal visual dictionary. In *ICCV*, 2005.