# BAYESIAN SEAFLOOR CHARACTERIZATION FROM SAS IMAGERY

Bart Gips[a]

[a]*NATO Centre for Maritime Research and Experimentation (CMRE)*

Viale San Bartolomeo 400, La Spezia, Italy
email: Bart.Gips@cmre.nato.int

***Abstract:*** *Synthetic aperture sonar (SAS) allows for high-resolution imagery of the seafloor with quality that approaches that of an optical camera. From these images a human can identify what type of seabed is in view (e.g. smooth sand, sand ripples, sea grass, rocks). Segmentation into different classes is useful in a setting where autonomous underwater vehicles (AUVs) are used, since information on the bottom type should be taken into account when controlling the behaviour of the AUV. In particular, we focus on mine countermeasures (MCM) applications, where AUVs are generally tasked with autonomously scanning the seabed for mine-like objects on the seafloor. In this scenario, characterizing the type of seafloor informs us about the difficulty the AUV will have when using an automatic target recognition (ATR) algorithm to find mine-like targets and as such this information can be used in a planning and performance evaluation context. Previous work has shown that seabed characterization is possible using through-the sensor features. In this work we focus on predicting the presence of different bottom types based on multifractal analysis of SAS images. Using a Bayesian framework, and a Gaussian process classification (GPC) algorithm, we not only predict the seabed type, but also provide an uncertainty estimate. Secondly, the GPC has the advantage that it will lead to predictions with high uncertainty if unknown data are encountered, i.e. when the SAS images contains features that are statistically different from the training dataset. We compare the GPC with a deterministic classification algorithm and highlight the advantageous and disadvantages compared to some of the methodology proposed in earlier work.*

***Keywords:*** *Bottom type characterization, Gaussian process classification, Synthetic Aperture Sonar.*

## 1. INTRODUCTION

Sonar images contain a lot of information about the sea bed characteristics. This is especially true in the case of Synthetic aperture sonar (SAS), which allows for high-resolution imagery of the seafloor with almost photo-like quality. From these high-quality images, it is relatively easy to make out whether the seabed consists of sand ripples, is smooth, or is covered with sea grass or rocks.

Segmentation into different classes of sea floor is useful when controlling autonomous underwater vehicles (AUVs), since the bottom type is a possible important input for the autonomous control algorithms that affect the behaviour of the AUV. In mine countermeasures (MCM) applications in particular, AUVs are generally tasked with autonomously scanning the seabed for mine-like objects on the seafloor. This task varies in difficulty depending on the sea floor type. For example, the presence of vegetation such as posidonia will make it drastically more difficult to detect mine-like objects on the seabed, whereas sand ripples make it more difficult to detect targets when it is insonified from an angle perpendicular to the sand ripples, since the ripples create acoustic shadows that obscure targets. For this reason previous work has sought to automate sea bed characterization [10, 17, 3] with a specific focus on sand ripples in particular [5, 19]. In future work we aim to use the information on the bottom type in order to quantify the difficulty the AUV will have when using an automatic target recognition (ATR) algorithm that is designed to find mine-like targets. As such this information can be used in the context of planning and performance evaluation [7].

In this paper we use Gaussian processes prior in order to perform bottom type segmentation. Gaussian processes are a set of non-parametric Bayesian models [14]. As such this makes them more flexible than more commonly used parametric probabilistic models.

In the case of statistical models with a Gaussian likelihood function, as is the case in popular least square regression applications, Gaussian process priors are analytically tractable. However, in classification problems as in the context of bottom type characterization, the likelihood is no longer Gaussian. Therefore the resulting expressions contain intractable integrals that need to be evaluated numerically, e.g. by using Monte Carlo methods, or they can be approximated, e.g. by using a Laplace approximation, variational methods, or expectation propagation [13].

A secondary challenge comes in the form of scalability. In general, inference using Gaussian processes requires $O(N^3)$ computation, where $N$ is the number of data points in the training set. A way to decrease this computational complexity comes in the form of sparse approximations. Generally this amounts to defining a set of *M induced* points, that are not necessarily a subset of the original data points [15]. These induced points do not have a corresponding target value (i.e. the class index in the case of classification), but instead they serve as "funnel" that attempts to catch and compress all information from the full original training dataset.

In this work, we use the induced points as variational parameters [8], in order to further reduce the computational demand. This allows us to use stochastic gradient descent to learn the model parameters, making it possible to use very large datasets to learn model parameters.

## 2. METHODS

Before we describe our Gaussian process-based classification model, we start by describing the more traditional multinomial logistic regression model. We will use it to contrast it with the more elaborate Gaussian process model.

### 2.1. Multi-class classification

Classification models generally work by transforming an input variable $\mathbf{x}_i$ into activation values $\mathbf{f}_i$. These activation values are then projected through a link function to project them onto the interval $[0,1]$ such that they can be interpreted as probabilities. For our logistic regression implementation, we use the softmax link function to translate an activation vector $\mathbf{f}_i = [f_i^1, f_i^2, \ldots, f_i^C]^T$ to the probability of class label $c \in \{1,2,\ldots,C\}$ [2]:

$$p(y_i = c|\mathbf{f}_i) = \exp(f_i^c)/\sum_{c'} \exp(f_i^{c'}) \tag{1}$$

where we use subscript $i$ for indexing over data points, and superscript $c$ for indexing over classes. In a traditional multi-class regression the activation vectors $\mathbf{f}_i$ are a linear combination of the feature values:

$$f_i^c = (\mathbf{w}^c)^T \mathbf{x}_i \tag{2}$$

We can then use Bayes' theorem to express the posterior over the model parameters $\mathbf{W}$:

$$p(\mathbf{W}|\mathbf{X}) = \frac{p(\mathbf{y}|\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_N) p(\mathbf{W})}{p(\mathbf{y})}. \tag{3}$$

with $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^N$ being the input dataset, and $\mathbf{y} = [y_1, y_2, \ldots, y_N]$ the target values, i.e. class labels.

#### 2.1.1 Gaussian processes for classification

Instead of using a linear combination of feature values $(\mathbf{w}^c)^T \mathbf{x}_i$, we can use a Gaussian process (GP) prior [14], i.e. where all activation values $\mathbf{f} = [\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_N]$ are jointly Gaussian:

$$p(\mathbf{f}|\mathbf{X}) = \mathcal{N}(\mathbf{f}|\mathbf{0}, \mathbf{K}) \tag{4}$$

The activation values are now no longer an explicit function of the input $\mathbf{X}$, but are the result of an *a priori* unknown non-linear transformation of $X$ ($\mathbb{R}^D \to \mathbb{R}^C$), where $D$ is the dimensionality of the input data. Eq. (4) tells us how smooth we expect this transformation to be over the input domain through the covariance function. We use the squared exponential function to model the covariance of the feature values belonging to class $c$:

$$K_{ij}^c = K^c(\mathbf{x}_i, \mathbf{x}_j|\boldsymbol{\theta}) = \exp(-\frac{1}{2}\sum_{d=1}^{D} \theta_d^c (x_i^d - x_j^d)^2) \tag{5}$$

where $K_{ij}$ are entries of matrix $\mathbf{K}$ in Eq. (4), and $x_i^d$ is the $d$-th component of feature vector $\mathbf{x}_i$.

This leads to a posterior directly over the activation values $\mathbf{f}$, rather than model parameters:

$$p(\mathbf{f}|\mathbf{y}, \mathbf{X}) = \frac{p(\mathbf{y}|\mathbf{f}) p(\mathbf{f}|\mathbf{X})}{p(\mathbf{y}|\mathbf{X})}. \tag{6}$$

Prediction of activation values $\mathbf{f}_*$ belonging to a new data point $\mathbf{x}_*$ is performed by conditioning on the activation values of the reference dataset $\mathbf{f}$ and marginalizing over the posterior:

$$p(\mathbf{f}_*|\mathbf{y}, \mathbf{X}, \mathbf{x}_*) = \int p(\mathbf{f}_*|\mathbf{f}, \mathbf{x}_*) p(\mathbf{f}|\mathbf{y}, \mathbf{X}) \mathrm{d}\mathbf{f}. \tag{7}$$

**Link function** For the linear regression case, the softmax link function is attractive, since it is easy to differentiate with respect to the weight parameters $\mathbf{W}$. As in [11], we use the robust max function in the case of our GP classification:

$$p(y_i = c|\mathbf{f}_i) = \begin{cases} 1-\varepsilon & \text{if } c = \text{argmax}(\mathbf{f}_i), \\ \varepsilon/(C-1) & \text{otherwise.} \end{cases} \tag{8}$$

**Sparse approximation** In order to deal with with large datasets we introduce a pseudo data set of size $M < N$ [15] to find a sparse approximation to the full GP. This pseudo dataset $\bar{\mathbf{X}} = \{\bar{\mathbf{x}}_i\}_{i=1}^{M}$ has corresponding activation values $\bar{\mathbf{f}} = [\bar{\mathbf{f}}_1, \bar{\mathbf{f}}_2, \ldots, \bar{\mathbf{f}}_M]$.

This means we will perform inference on the pseudo activations $\bar{\mathbf{f}}$, i.e. we alter Eq. (6):

$$p(\bar{\mathbf{f}}|\mathbf{y}) = \frac{\int p(\mathbf{y}|\mathbf{f}) p(\mathbf{f}|\bar{\mathbf{f}}) \mathrm{d}\mathbf{f}\; p(\bar{\mathbf{f}}|\bar{\mathbf{X}})}{p(\mathbf{y})} \tag{9}$$

where $p(\mathbf{f}|\bar{\mathbf{f}})$ and $p(\bar{\mathbf{f}}|\bar{\mathbf{X}})$ follow our original GP prior (Eq. (4)). We use a variational approximation to this posterior, where the induced points are considered to be variational parameters as in [8].

**Implementation** For our control methodology of (linear) multinomial logistic regression, we use Newton's method to find the maximum a posteriori solution for weight parameters $\mathbf{W}$ in Eq. (3). The prior on the weight values is set to a univariate normal distribution with a mean of zero and a standard deviation of 10, i.e. $p(w_i^c) = \mathcal{N}(w_i^c|0, 10^2)$.

We implemented the variational multi-class GPC following the approach found in [8], using the GPflow python module [12]. We used a the robust max link function Eq. (8), with $\varepsilon = 10^{-3}$. The covariance functions were set to squared exponential (see Eq. (5)). Since we did not *a priori* expect different bottom type classes to have the same length scales in the space spanned by our predictors $\mathbf{x}_i$, we used class-specific hyper-parameters $\boldsymbol{\theta}_c$. These hyper-parameters were determined by maximization of the lower bound to the marginal likelihood (i.e. minimisation the variational free energy). The $M$ pseudo inputs $\bar{\mathbf{X}}$ were initially selected at random from the training data set. Subsequently the variational free energy was minimized using the Adam method for stochastic optimisation [9].

### 2.2. Sonar Data

In total we had 254 SAS images in which a human had judged the sea bottom type by drawing polygons over the original image. The images were recorded of the coast of Bonassola, Italy by the SAS on board the CMRE MUSCLE AUV [1]. Subareas within each image were labelled by hand to belong to one of 5 classes: ripples, posidonia, posidonia on ripples, rocks, or smooth sand. These SAS images had a resolution of 2001 by 7333 pixels with a corresponding resolution of 0.025 and 0.015 m/pixel respectively in the along-track an across track direction.

**Preprocessing** Before extracting any features from a SAS image, the image was normalized in a manner similar to the method reported in [18]. Briefly, the modulus of the complex SAS image was log-transformed, after which the image was normalized row- and column-wise.

In order to facilitate data handling, we subsampled the images after the normalization step by a factor of 2 and 3 to achieve a more square resolution of 0.05 and 0.045 m/pixel.

For training and testing of our classification algorithms we extracted 500 samples at random from every SAS image to put in a training data set, and we extracted another 100 different random data points to use in a test set. In total, this initially resulted in a training set with 127000 data points and a test set containing 25400 points. To train the classification models in

an unbiased manner, the training set was further downsampled by randomly drawing without replacement where necessary such that every class had equal number of datapoints. This lead to a total of $4968 \times 5 = 24840$ points in the final training data set.

### 2.2.1 Predictive features

For either of our classification algorithms to function we need features to populate the vectors $\mathbf{x}_i$ (e.g. see Eq. (2)).

**Ping-to-ping correlation** The on-board processing of the SAS images on the MUSCLE provides us with a ping-to-ping correlation metric [4]. The ping echo time series were cut up into 10 time series of equal length centred on time samples corresponding to different ranges, giving us correlation coefficients for 10 different range bins. These values were then mapped to the image coordinates by averaging the correlation coefficients of all ping pairs that contributed to an image pixel.

**Lacunarity** Lacunarity has previously been shown to be a good metric for sea floor characterization in sonar imagery [17]. Lacunarity was estimated by a box counting method and is equal to the square of the coefficient of variation, i.e. the ratio between the sample variance and the squared mean.

**Fractal dimensionality metrics** We followed the procedure suggested in [3], and consider both the asymmetric and the directional fractional dimension, such that for every length scale we will have 4 metrics. We considered length scale ratios of $1:2$, $2:4$, and $4:8$ pixels, this lead us to a total of 12 different fractal dimension estimates. Finally we smoothed these estimates using a $50 \times 50$ pixel rectangular normalized filter kernel to reduce high-frequency noise. We assumed the relevant changes in these metrics to be the result of differences in bottom type, therefore the relevant changes should vary smoothly across the image and will not be present in high-frequency components.

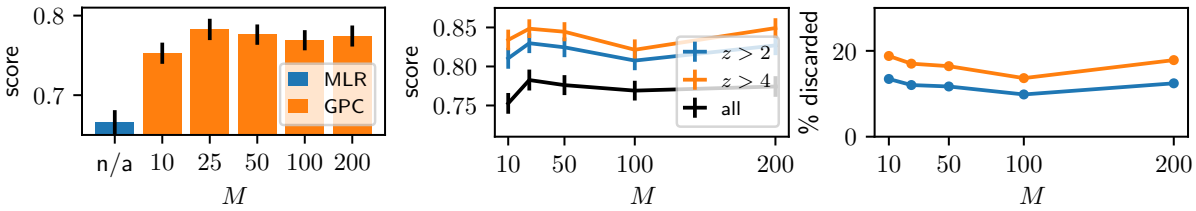
## 3. RESULTS

As mentioned in section 2.2, we used a training set consisting of 127000 data points and a test set of 25400 data points. We first applied the simpler multinomial logistic regression by calculating the maximum a posteriori estimate for the weight parameters based on the complete dataset. After having learned the weight parameters, we used them to predict the bottom type labels in our test set. To get an estimate on the standard error of the prediction performance we applied a bootstrapping procedure where we generated 1000 bootstrapped test datasets through resampling with replacement. The mean and standard deviation of the MLR prediction performance across these 1000 resampled test sets are shown with blue bars in Fig. 1.

Similarly we train the Gaussian process classifier (GPC) using different number of induced points $M$ and predicted the class labels of the test set. The results are show with orange bars in Fig. 1. Note that all GP classifiers outperformed the MLR model on average as measured by the classifier accuracy scores in Fig. 1a, with $M = 25$ the best performing model among our tests, increasing the number of induced points beyond 25 seemingly lead to over-fitting.

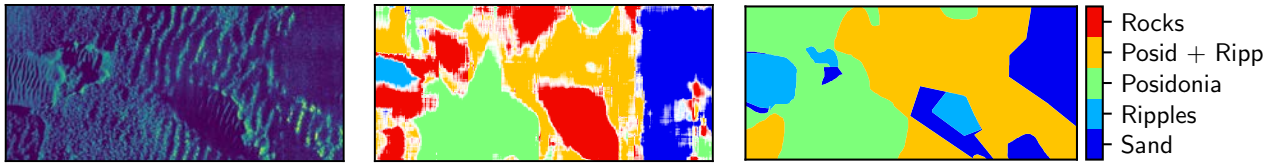
Note that classifying by chance would lead to a score of 20% and the classification accuracy of $> 66\%$ for the MLR model is well above that. This suggests that given the predictive features we use (section 2.2.1), the 5 bottom type classes are fairly separable using a linear model.

Our GPC models provide us with an uncertainty estimate on our predictions. To illustrate its usefulness, we repeated the accuracy analysis, but now we leave out predictions that are not




(a) (b) (c)

*Fig. 1: (a) Prediction accuracy score (percentage correctly classified) for the MLR model (blue bar) and for the Gaussian process classifies using different number of induced points M (orange bars). (b) Prediction accuracy score as in 1a, with and without discarding samples for which prediction was uncertain. (c) Number of samples discarded to generate curves in (b). Bars and curves in (a) and (b) depict mean values across 1000 resampled test sets (bootstraps), error bars depict 5 times the standard deviation across these bootstrap distributions.*




(a) (b) (c)

*Fig. 2: (a) example SAS image. (b) GPC prediction using the model with $M = 25$, transparency level reflects uncertainty ($z$ in Eq. (10)). (c) Manually labelled bottom types.*

certain. To roughly quantify certainty of a prediction we calculate a "z-score" quantifying the statistical distance to the nearest alternative class:

$$z = \min_{k \neq i} \left[ (\mu_i - \mu_k) / (\sqrt{\sigma_i^2 + \sigma_k^2}) \right] \tag{10}$$

where $\mu_k$ and $\sigma_k^2$ are the mean and variance of the predicted probability that the test point belongs to class $k$. We use index $i$ to indicate the class with the most probability mass (i.e. $\mu_i > \mu_k,\ \forall k \neq i$). The higher the value of $z$, the more certain our prediction, with $z = 0$ when two or more "winning" classes are equally likely.

We performed two new analyses where we either discarded all test data for which $z \leq 2$ (blue curve), or $z \leq 4$ (orange curve). The results are shown in Fig. 1b, showing that indeed, the GPC model's predictions with lower uncertainty have higher accuracy, as seen by the accuracy increasing to approximately 85% when around 17% of the test points were discarded (Fig. 1c) for the cases of $M = 25$ and $M = 200$.

As an illustration, Fig. 2 shows a difficult example SAS image with manually segmented bottom types (Fig.2c), together with the most likely bottom type as predicted by the GPC model with $M = 25$ (Fig.2b). Note that the uncertainty is highest around the borders between bottom type classes, and that the "rocks" class seems overrepresented.

## 4. DISCUSSION

In this work, we applied Gaussian process classification to automated bottom type characterisation based on sonar imagery. Their non-parametric nature makes GP models able to perform well for data sets with non-linear class boundaries. The highest performance in simi-

lar classification problems is generally achieved by convolutional neural networks (CNN) [10]. The advantage of CNNs lies in the fact that they automate feature extraction and can find a non-linear transformation that separates the classes. However, their performance comes with a considerably higher computational cost, and since the final step in most classifier CNNs is a softmax function, they generally do not provide uncertainty estimates. In addition, an important feature of GPs that we have glossed over so far, is the fact that GP models do not work with a class boundary captured in model parameters (e.g. the weights in Eq. (2)), but with a covariance function that typically depends on the distance to the data points in the training set. This means that after learning, when we encounter a new data point that is not at all similar to our training data set (e.g. a new bottom type, or very noisy or distorted data), the GP will give a non-informative result. That is, the classes will be equally probable admitting maximum uncertainty, a feature neither the MLR model nor a sofmax-based CNN can exhibit. Even if an uncertainty estimate were introduced (e.g. by using the full posterior, rather than the maximum a posteriori estimate [2] of the MLR weights $\mathbf{W}$ in Eq. (3), or through dropout in the CNN [6]), uncertainty estimates on the class probabilities will go to zero far away from class boundaries, this could cause the classifier to yield a very confident false result for a new point far way from the training data set.

While the stochastic optimisation approach to learning the induced points $\bar{\mathbf{X}}$ and optimising the GP hyper-parameters allows us to deal with large datasets, it comes with considerably longer training times compared to other sparse methods that use the full (smaller) data set during every iteration. However, this training is still very feasible, training our GPC model with M=25 induced points took less than 30 minutes on a quad-core 3.6 GHz desktop computer. Besides, all the training can be done offline, since once these parameters have been optimized, prediction amounts to calculating the mean and variance of the activation value $\mathbf{f}_*$ in $O(CM^2)$ [8], where $C$ is the number of classes, and $M \ll N$ is the number of inducing points. Note that since we are using a sparse approximation rather than a full Gaussian process, we are able to "compress" the 24840 training points to only 25 induced points, significantly decreasing the computational load for prediction. Assuming feature extraction can be done sufficiently fast, the GP model gives the opportunity for significantly faster inference than to a similar CNN approach. To improve the performance of GPC in the context of bottom type segmentation, more investigation in optimizing the set of predictive features is warranted, e.g. fractal dimensionality of different length scales can be considered. Since bottom types tend to vary smoothly across space, spatial covariance could be taken into account in our model, although care has to be taken such that the covariance will not go to zero as spatial distance increases. To achieve this, the spatial covariance can be modelled with its own covariance function, such that sonar-based and space-based covariance are additive. Altering the covariance function itself could also improve performance, e.g. adding a white noise term, and using a Matérn kernel [14] instead of the squared exponential function Eq. (5), as the latter is considered by some to be too smooth for modelling physical data [16]. Taken together, Gaussian process classification is a promising statistical method for bottom type characterisation based on sonar imagery.

## ACKNOWLEDGEMENTS

The author would like to thank the CMRE's ANMCM team for labelling the training dataset, and Thibaud Berthomier in particular for creating the software tool that made this possible.

## REFERENCES


[1] F. Baralli, M. Couillard, J. Ortiz, and D. G. Caldwell. Gpu-based real-time synthetic aperture sonar processing on-board autonomous underwater vehicles. In *OCEANS-Bergen, 2013 MTS/IEEE*, pages 1–8. IEEE, 2013.

[2] C. M. Bishop. *Pattern recognition and machine learning*. springer, 2006.

[3] D. Carmichael, L. Linnett, S. Clarke, and B. Calder. Seabed classification through multifractal analysis of sidescan sonar imagery. *IEE Proceedings-Radar, Sonar and Navigation*, 143(3):140–148, 1996.

[4] M. Couillard, J. Groen, and W. L. Fox. Performance assessment of the muscle synthetic aperture sonar. In *Proceedings of Meetings on Acoustics ECUA2012*, volume 17, page 070102. ASA, 2012.

[5] A. Crawford and A. Skarke. Automatic detection of sand ripple features in sidescan sonar imagery. In *2014 Oceans - St. John's*, pages 1–5, 2014.

[6] Y. Gal and Z. Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059, 2016.

[7] B. Gips, C. Strode, and S. Dugelay. Residual risk maps for performance assessment of autonomous mine countermeasures using synthetic aperture sonar. volume 40, pages 47–56. Institute of Acoustics, 2018.

[8] J. Hensman, A. Matthews, and Z. Ghahramani. Scalable variational gaussian process classification. 2015.

[9] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[10] D. Kohntopp, B. Lehmann, D. Kraus, and A. Birk. Seafloor classification for mine countermeasures operations using synthetic aperture sonar images. In *OCEANS 2017-Aberdeen*, pages 1–5. IEEE, 2017.

[11] A. G. d. G. Matthews. *Scalable Gaussian process inference using variational methods*. PhD thesis, University of Cambridge, 2017.

[12] D. G. Matthews, G. Alexander, M. Van Der Wilk, T. Nickson, K. Fujii, A. Boukouvalas, P. León-Villagrá, Z. Ghahramani, and J. Hensman. Gpflow: A gaussian process library using tensorflow. *The Journal of Machine Learning Research*, 18(1):1299–1304, 2017.

[13] H. Nickisch and C. E. Rasmussen. Approximations for binary gaussian process classification. *Journal of Machine Learning Research*, 9(Oct):2035–2078, 2008.

[14] C. Rasmussen and C. Williams. *Gaussian Processes for Machine Learning*. the MIT Press, 2006.

[15] E. Snelson and Z. Ghahramani. Sparse gaussian processes using pseudo-inputs. In *Advances in neural information processing systems*, pages 1257–1264, 2006.

[16] M. L. Stein. *Interpolation of Spatial Data: Some Theory for Kriging*. Springer Science & Business Media, 1999.

[17] D. P. Williams. Fast unsupervised seafloor characterization in sonar imagery using lacunarity. *IEEE transactions on Geoscience and Remote Sensing*, 53(11):6022–6034, 2015.

[18] D. P. Williams. The mondrian detection algorithm for sonar imagery. *IEEE Transactions on Geoscience and Remote Sensing*, 56(2):1091–1102, Feb 2018.

[19] D. P. Williams and E. Coiras. On sand ripple detection in synthetic aperture sonar imagery. In *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on*, pages 1074–1077. IEEE, 2010.