An Automated Method for Semantic Classification of Regions in Coastal Images

Bas Hoonhout[a,b,*], Max Radermacher[a], Fedor Baart[a,c], Laurens van der Maaten[d]

[a] *Department of Hydraulic Engineering, Faculty of Civil Engineering and Geosciences, Delft University of Technology, Stevinweg 1, 2628CN Delft, Netherlands*
[b] *Unit of Hydraulic Engineering, Deltares, Boussinesqweg 1, 2629CN Delft, Netherlands*
[c] *Deltares Software Centre, Deltares, Boussinesqweg 1, 2629CN Delft, Netherlands*
[d] *Department of Intelligent Systems, Faculty of Electrical Engineering, Mathematics and Computer Science, Delft University of Technology, Mekelweg 4, 2628CD Delft, Netherlands*

Abstract

Large, long-term coastal imagery datasets are nowadays a low-cost source of information for various coastal research disciplines. However, the applicability of many existing algorithms for coastal image analysis is limited for these large datasets due to a lack of automation and robustness. Therefore manual quality control and site- and time-dependent calibration is often required. In this paper we present a fully automated algorithm that classifies each pixel in an image given a pre-defined set of classes. The necessary robustness is obtained by distinguishing one class of pixels from another based on more than a thousand pixel features and relations between neighboring pixels rather than a handful of color intensities.

Using a manually annotated dataset of 192 coastal images, a Structured Support Vector Machine (SSVM) is trained and tested to distinguish between the classes *water*, *sand*, *vegetation*, *sky* and *object*. The resulting model correctly classifies 93.0% of all pixels in a previously unseen image. Two case studies are used to show how the algorithm extracts beach widths and water lines from a coastal camera station.

Both the annotated dataset and the software developed to perform the model training and prediction are provided as free and open-source data sources.

Keywords: coastal, images, semantic classification, support vector machine, features

*Corresponding author
Email addresses: bas.hoonhout@deltares.nl (Bas Hoonhout), b.m.hoonhout@tudelft.nl (Bas Hoonhout)

1. Introduction

Coastal imagery nowadays is a valuable and low-cost source of information for coastal research in a variety of disciplines. Characteristics such as beach width, water line dynamics, wave breaking and runup, vegetation cover, aeolian dynamics and beach usage are visible to the naked eye from a simple coastal image (e.g. Figure 1, A). Further analysis of the acquired images can provide us with derived information like bathymetries, flow patterns and sediment transport trajectories. Coastal image analysis is not restricted to ordinary visible light imagery, but can be applied to (near-)infrared, multi- or hyperspectral imagery and video as well, increasing the number of coastal features that can be distinguished.

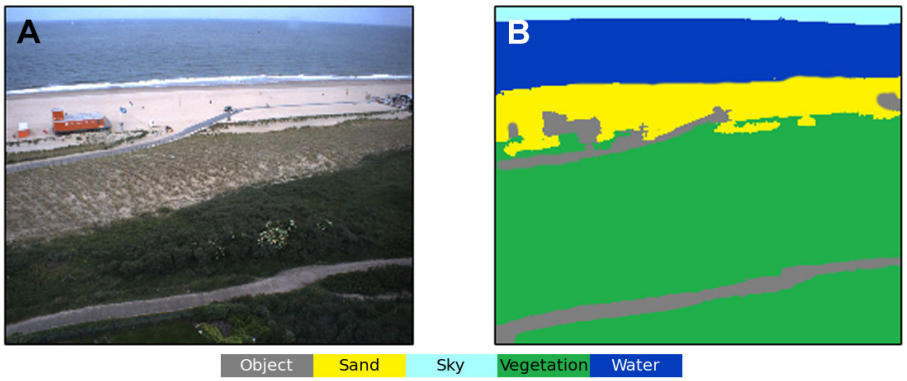

Figure 1: Example of a coastal image taken on July 1st, 2013 in Kijkduin, The Netherlands (A) and the corresponding manual annotation (B).

Since investments to install a coastal camera station and corresponding data storage are low compared to most other monitoring alternatives, the amount of coastal camera stations worldwide is increasing steadily. With the increasing amount of coastal imagery data, an increasing number of coastal image analysis algorithms is being developed with a variety of applications. Pioneering work on swash runup estimates from coastal images was done by *Holland and Holman* (1991). The extraction of runup lines inspired many authors to map intertidal bathymetries from series of runup lines obtained from a series of coastal images (e.g. *Plant and Holman*, 1997; *Aarninkhof et al.*, 2003; *Quartel et al.*, 2006; *Plant et al.*, 2007; *Uunk et al.*, 2010; *Osorio et al.*, 2012). Many shoreline extraction algorithms are available, including those that use multispectral images for increased precision (e.g. *Sekovski et al.*, 2014). Subsequently, coastal images were used to estimate surfzone currents (e.g. *Holland et al.*, 2001; *Chickadel et al.*, 2003) and later subtidal bathymetries (e.g *Aarninkhof et al.*, 2005; *van Dongeren et al.*, 2008; *Holman et al.*, 2013). The global presence of coastal camera stations makes it possible to monitor long-term coastal behavior (*Smit et al.*, 2007) and sparked several applications for coastal zone managers, like

estimating coastal state indicators (*Davidson et al.*, 2007), deriving beach usage statistics (*Jimenez et al.*, 2007; *Guillén et al.*, 2008), tracking sandbar positions (*Lippmann and Holman*, 1989; *van Enckevort and Ruessink*, 2001; *Price and Ruessink*, 2011) and rip current detection systems (*Bogle et al.*, 2000; *Gallop et al.*, 2009).

The size of the coastal imagery archive grows rapidly. New camera stations are deployed every year, adding to the diversity of the total dataset. The data intake per station is increasing, which makes the total dataset harder to analyze. The applicability and performance of these algorithms on the large coastal imagery datasets that are presently available depends on two characteristics in particular: automation and distinctiveness of relevant pixels. Many algorithms need some kind of manual pre-selection of relevant pixels (the region of interest) or manual quality control, which often hampers analysis of large, long-term coastal imagery datasets (e.g. *Holland and Holman*, 1991; *Aarninkhof et al.*, 2003; *Jimenez et al.*, 2007). Algorithms that do not rely on a manual quality control need to distinguish between classes of pixels based on a limited number of intrinsic pixel features, usually the color channels Red, Green and Blue (RGB) or Hue, Saturation and Value (HSV) (e.g. *Aarninkhof et al.*, 2003; *Quartel et al.*, 2006). Consequently, feature characteristics for different classes are very likely to overlap and automated classification of large, long-term coastal imagery datasets becomes unreliable if not unfeasible without site- and time-dependent calibration, limiting the applicability of the algorithm.

To the best of our knowledge, this paper presents the first fully automatic, high-quality algorithm for the supervised segmentation of coastal images into image regions containing major classes such as *water*, *sand*, *vegetation*, *sky*, and *objects* (Figure 1, B). In contrast to prior work, this algorithm does not rely on a few color features with limited discriminability between classes, but aggregates over more than a thousand features that contain information on color, texture, and visual appearance. In addition, the algorithm uses a machine learning framework that allows us to leverage thousands of features for high-accuracy classification and enables the use of structured learning where relations between neighboring pixels are taken into account. The algorithm is not tailored to any specific classification task, but is merely a general classification framework that can be applied on large, long-term coastal imagery datasets or on parts of individual images. Nonetheless, it produces substantially better results than obtained by tailored algorithms that rely on color features alone and omit the use of structured learning. In addition, we present a manually annotated dataset of coastal images that can be used for training and testing of machine-learning based systems such as ours, and we present an open-source Python toolbox for performing coastal image analysis tasks.

2. Methodology

Automated classification of regions in coastal images is done using a classification model. Classifying image regions into various meaningful classes based on a set of properties (features) shows similarities to regression models (e.g.

linear regression). Regression models are used to predict the value of a target parameter based on some input samples. In principle, classification models are regression models, which use a threshold value for the target parameter to distinguish between a set of discrete classes.

Supervised classification models (as opposed to unsupervised models, which are not treated here) require training. During training the optimal threshold values are determined based on an annotated dataset. Optimization is done by minimizing a cost function. The definition of this cost function is the main factor that distinguishes between various model types. For example, a linear regression model usually minimizes the mean squared error of the predicted target parameter over all training samples. Artificial neural networks, which are basically a network of regression models, are occasionally used for classification purposes as well (e.g. *Kingston*, 2003; *Verhaeghe et al.*, 2008). In this study a method closely related to a Logistic Regressor (LR) is used (e.g. *Vincent*, 1986; *Dusek and Seim*, 2013). A LR is, although the name suggests regression, a classification method that optimizes the logistic loss function.

The workflow adopted in this study consists of four steps: 1. a manually annotated dataset of coastal images is oversegmented into *superpixels*; 2. for all images in the dataset an extensive set of features is extracted; 3. a suitable classification model is trained using the manually annotated data; 4. this model can be used to automatically classify future images. The workflow is visually presented in Figure 2. In this section these four steps are described. In the next section the performance as well as a first application of the algorithm is presented.

2.1. Dataset

The ArgusNL dataset with manually annotated coastal images consists of 192 images obtained from 4 coastal camera stations located along the Dutch coast (Egmond, Jan van Speijk, Kijkduin and Sand Motor), each containing 5 to 8 cameras.

Each camera that is part of the coastal camera stations used in this study takes a snapshot image twice an hour. Apart from snapshot images, also 10 minutes mean, variance, min and max images are stored, but these are not used for classification. Also, any images that were obscured and thus did not contain any valuable data are discarded. Images can be obscured either because the image was taken before sunrise or after sunset or because of the presence of rain, fog, sun glare in the water or dirt on the camera lens.

From all suitable snapshot images taken during the summer of 2013 by these cameras 192 images are randomly selected. The images are automatically oversegmented (see subsection 2.2) and one of the following classes is assigned manually to each segment: 1. sky; 2. water (sea); 3. water (pool); 4. sand (beach); 5. sand (dune); 6. vegetation; 7. object (sea); 8. object (beach); 9. object (dune). In this study the classes are aggregated to the most relevant ones, being: 1. sky; 2. water; 3. sand; 4. vegetation; 5. object.

The ArgusNL dataset, including the images, oversegmentation and annotation, was published by *Hoonhout and Radermacher* (2014).

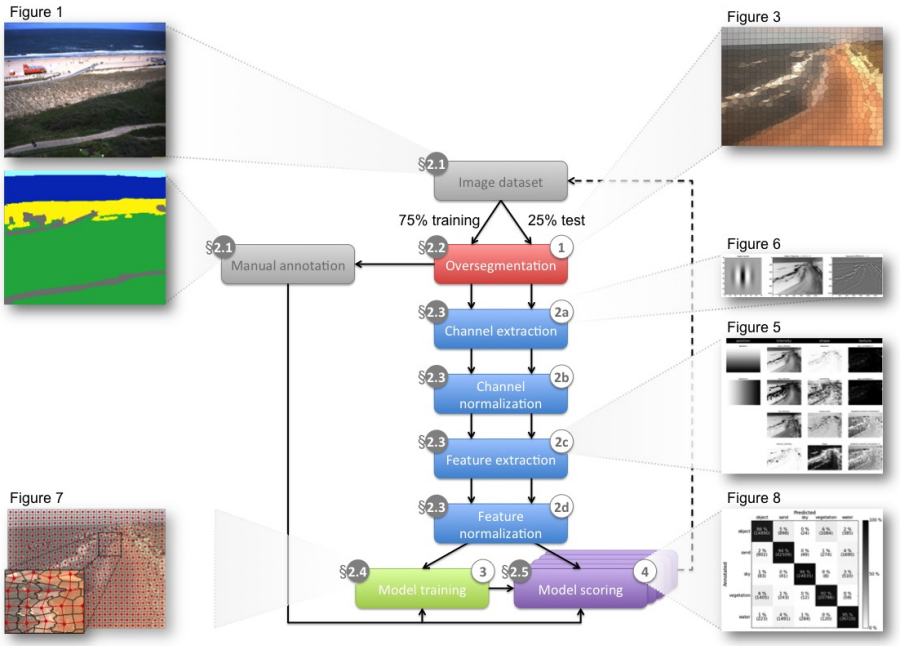

Figure 2: General workflow of classification algorithm with the four main steps as explained in section 2. Splitting the dataset in a training and test set is repeated multiple times to limit the dependence of the actual split and results in a set of model scores. Each step in the algorithm is numbered for referencing in the main text. The thumbnails provide visual references to the figures presented in sections 1 and 2.

2.2. Oversegmentation

Segmentation is the process of subdividing the image in functional segments, like dunes, beach and sea. Oversegmentation is the process of subdividing the image in smaller segments of similar pixels, which are called *superpixels*, that do not necessarily constitute a functional part of the image[1]. Oversegmentation into superpixels enables us to boost the number of features that can be used for classification (Figure 3 and Figure 2, step 1). The automated segmentation algorithm SLIC (*Achanta et al.*, 2012) is adopted for this purpose.

The SLIC algorithm is a computationally inexpensive algorithm that projects each pixel in an image to be segmented into a 5-dimensional space based on 2 image coordinates (U and V) and 3 intensity or color dimensions (usually using the perceptually linear CIELAB colorspace rather than RGB). Subsequently, a predefined number of cluster centers is chosen in this space. Each cluster center will form a superpixel. Initially, the cluster center locations are chosen as an equidistant regular grid over the image. 600 superpixels are used for this

[1]The terms segmentation and oversegmentation are used indiscriminately in the remainder of this paper. In both cases the process of oversegmentation is referred to.

Figure 3: Original image (left) and oversegmented image (right).

dataset, although the exact number may vary per image depending on its dimensions since a regular grid is to be assured. Each pixel is assigned to the closest cluster center. Subsequently, like in K-means clustering (*Lloyd*, 1982), the RMS distance in the 5-dimensional space from each pixel to its corresponding cluster center is then iteratively minimized by moving the cluster centers each iteration step to the centroids of the current cluster and updating the cluster assignments.

The SLIC algorithm deviates from the standard K-means algorithm in the treatment of the spatial dimensions U and V. First, the area in an image that can be covered by a single superpixel is limited. A superpixel can therefore not stretch from one corner of the image to another. Second, the importance of the spatial dimensions is weighed compared to the intensity dimensions using a user-defined compactness parameter. A high compactness results in relatively square and heterogeneous superpixels, while a low compactness results in homogeneous, but scattered superpixels. A compactness of 20 is used throughout this study.

The SLIC algorithm does not prevent a superpixel from comprising distinct image structures. Such scattered superpixels may have undesirable effects. For example, white-capping of waves may become part of a single superpixel together with white sand at the beach. Along the color dimensions these pixels are very close and possibly equal. Along the spatial dimensions these pixels may be very close as well, but not connected. In order to prevent these ambiguous superpixels from appearing, the superpixels are forced to be connected by running a region growing algorithm to post-process the segmentation result. The region growing algorithm simply determines if all pixels within a superpixel are interconnected. If not, the largest cluster within the superpixel is preserved and the other, smaller clusters are assigned to the surrounding superpixels. If more superpixels surround a cluster, the one that shares the largest boundary is chosen.

2.3. Feature extraction

The success of classification depends on the distinctive quality of the different features. The color features that are in common use (*Quartel et al.*, 2006; *Aarninkhof et al.*, 2003) are not sufficient to separate the different classes that occur. Features are not very distinct when only looking at intrinsic features, like Red, Green and Blue (RGB) or Hue, Saturation and Value (HSV), alone.

Figure 4 shows an image in which three superpixels are selected from the classes *water*, *sand* and *object*. The intrinsic features for these superpixels show very limited variability over the classes, which makes reliable classification difficult. A key property of the proposed algorithm is the use of many derived features that can distinguish classes over a range of different conditions.

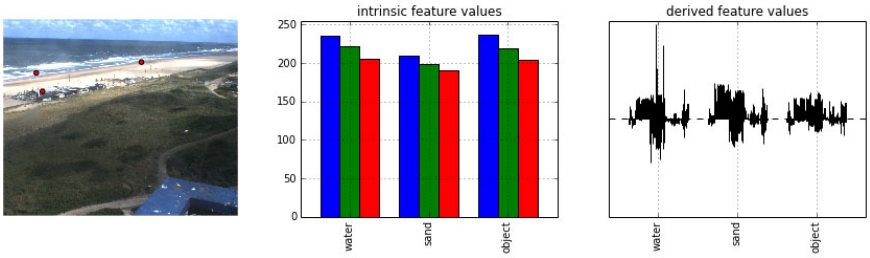

Figure 4: Coastal image from Kijkduin, The Netherlands, with three selected superpixels from the classes *water*, *sand* and *object* (left, red dots). Color intensities alone show limited variability over the classes (middle), while derived features provide much more distinctiveness (right).

After transformation of the image from pixels with only color to superpixels, these additional features become available. Superpixels have variance, patterns, texture, color, shape and topological relations that can be transformed into features. This allows to generate more than a thousand of different new features (Figure 2, step 2a). The challenge is to find a set of features that describe the classes that are of interest uniquely.

By using location features obvious information is learned, for instance sky is generally above the sea and sea is often above the beach. Shape features can for example help separate breaking waves as these tend to be eccentric and have highly irregular superpixel shapes. Some areas require texture features that describe the spatial variation and structure of colors and intensities. Vegetation can be inferred from a Grey Level Co-occurrence Matrix (GLCM, *Haralick et al.*, 1973) applied to the green channel (*Lu*, 2005).

The proposed classification algorithm uses 1727 features from the categories 1. position 2. intensity 3. shape 4. texture . Figure 5 shows a selection of features from these categories. The features of position and intensity are computed using trivial functions. The shape and texture definitions represent more complex characteristics. For example, the *holeyness* is defined as the superpixel convex hull area divided by the total pixel area. The *eccentricity* is the ratio of the minor and major axis of an ellipse fitted to the shape of the superpixel and *shape* is defined as the area divided by the squared perimeter. The *gray correlation* correspond to the correlation with a Grey Level Co-occurrence Matrix with interval of 5 and 17 pixels. In the full feature set the gray patterns are varied by angle. Figure 5 shows that shape features distinguish breakers and vegetation from beach, sky and deep sea, while texture features mainly separate beach from sea. The whole set of features combined provides information on how to

distinguish classes individually.

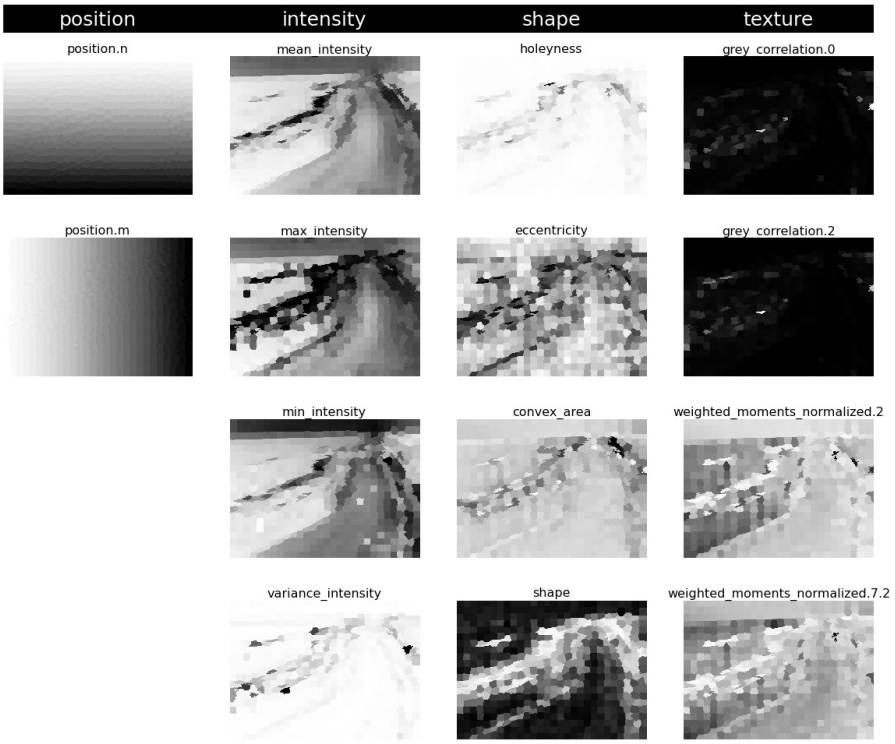

Figure 5: A selection of features from the four main categories used for classification extracted from Figure 3. Low values are light, high values are dark.

One of the categories that is hard to separate is water from wet or dry sand. This can be done more accurately using additional channels, like near-infrared (*Hoonhout et al.*, 2014), but additional channels can also be added artificially (Figure 2, step 2c). Here extra channels are created based on difference of Gaussian filtering (*Crowley and Parker*, 1984) and Gabor filtering (*Gabor*, 1946). Difference of Gaussian filtering enhances areas in an image with large pixel intensity gradients, for example areas which are darker (or lighter) than their surroundings, such as wet sand. Gabor filtering is a method to enhance image texture at various scales and orientations. Figure 6 shows an impression of these extra artificial channels. Each of these channels is added to the original, segmented image and for each of these channels features are extracted.

Both channels and features are postprocessed to make sure that features are invariant for scale of the image and/or superpixel. For example, the *area* of a superpixel can be a feature, but is expressed in number of pixels. Since the average number of pixels in a superpixel varies depending on the image size, the feature value depends on the image size as well. Such a feature value is divided

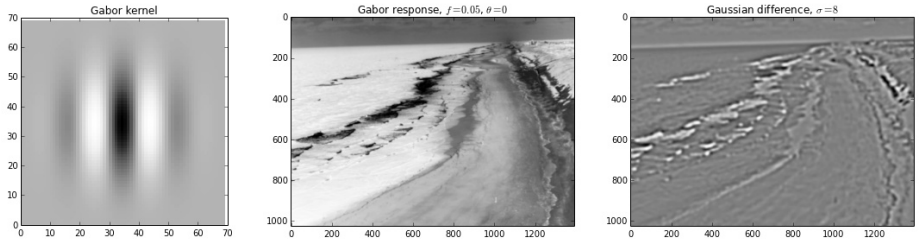

Figure 6: A selection of artificial channels extracted from Figure 3: Gabor kernel (left) with corresponding filter response (middle) used to detect textures that are shore-parallel and Gaussian difference response (right) used to detect relatively dark and lighter areas.

by the total number of pixels in the image to make it invariant for the scale of the image. Moreover, some features may produce much bigger values than others due to the nature of the feature. This may cause a preference for large features by the classification model. To prevent this, all features are scaled to a standard-normal distribution based on the range of feature values found in the training dataset (Figure 2, steps 2b and 2d).

2.4. Model definition

The actual classification of regions in coastal images is done using a classification model (Figure 2, step 3). As discussed in the introduction of this section, various model types are suitable for this learning task (*Koller and Friedman*, 2009). In this study a model closely related to a Logistic Regressor (LR) is used. A LR is an example of an unstructured model that performs classification of individual superpixels without looking at the rest of the image. A Support Vector Machine (SVM) is very similar to a LR, but optimizes the hinge loss function rather than the logistic loss function. The difference between the two cost functions is actually very minor. Alternatively, a structured model that performs classification of individual superpixels by taking their spatial context into account can be used. A Conditional Random Field (CRF) and a Structured Support Vector Machine (SSVM) are both examples of structured models. The former optimizes the logistic loss function, whereas the latter optimizes the hinge loss function. Generally, unstructured models are less complex, but structured models perform better on image classification tasks. In this study, a SSVM is used, because of the increased performance compared to unstructured models and the availability in the PYSTRUCT toolbox (*Mueller and Behnke*, 2014). The description of the model as presented below can be found in *Nowozin and Lampert* (2011).

For any of these models, the class of a superpixel is inferred by choosing the class such that the prediction function is maximized, see Equation 1.

$$\underset{y \in \mathcal{Y}}{\operatorname{argmax}} \quad \boldsymbol{\omega}^T \Psi(\mathbf{X}, y) \tag{1}$$

Here, y is the field of predicted class labels for all superpixels in the image, $\mathcal{Y}$ is the output space consisting of all valid class labels, $\boldsymbol{\omega}$ is a learned parameter

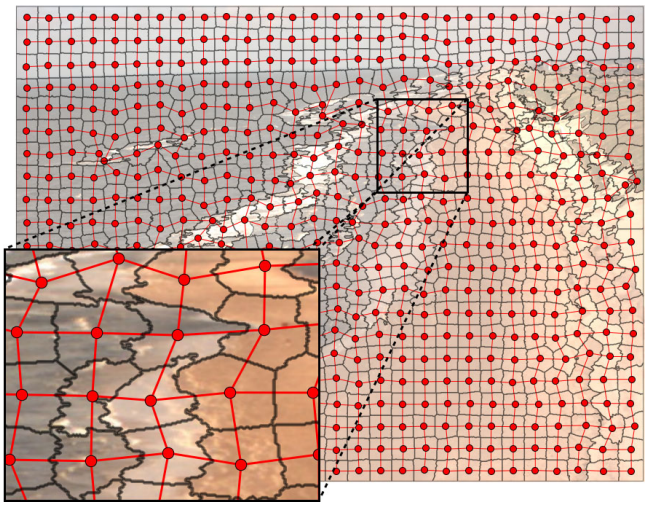

Figure 7: Structure of the model used for coastal image classification in this study. Every superpixel is associated with a set of features through unary potentials (red dots) and to its neighboring superpixels through pairwise potentials (red lines).

vector, T denotes the transpose and $\mathbf{X}$ is the $n \times m$ matrix containing m feature values for all n superpixels in the image. The joint feature function Ψ depends on the specific type of model that is applied. In case of the SSVM used in this study, the product $\boldsymbol{\omega}^T \Psi(\mathbf{X}, y)$ is given by Equation 2.

$$\boldsymbol{\omega}^T \Psi(\mathbf{X}, y) = \sum_{i=1}^{n} \omega_{y_i}^{(b)} + \sum_{i=1}^{n} \sum_{j=1}^{m} \omega_{j,y_i}^{(u)} x_{i,j} + \sum_{i=1}^{n} \sum_{j \in \mathcal{N}_i}^{m} \omega_{y_i,y_j}^{(p)} \tag{2}$$

From this equation, it follows that the vector $\boldsymbol{\omega}$ consists of three different types of parameters: 1. a bias $\omega^{(b)}$ for every valid class label, which expresses the prevalence of a class over the other classes without looking at a specific image or superpixel 2. a unary potential $\omega^{(u)}$ for every possible combination of class and feature, expressing the degree to which a feature characterizes a certain class 3. a pairwise potential $\omega^{(p)}$ for every possible combination of two class labels. The latter is evaluated over each superpixel's connectivity neighborhood $\mathcal{N}$ and expresses the credibility of two classes being assigned to neighboring superpixels. In the case of the coastal imagery dataset, this will for example yield that the class *vegetation* is less likely to border *sea* than *sand*. The structure of the SSVM used in this study is visualized in Figure 7. In this study, the connectivity neighborhood of a superpixel consists of the four superpixels it shares direct borders with, but can generally take any shape.

The parameter-vector $\boldsymbol{\omega}$ has to be trained using a set of ℓ training examples. To this end, the training dataset of annotated coastal images is used (*Hoonhout and Radermacher*, 2014). The actual learning is done by minimizing the regu-

larized cost function given in Equation 3.

$$\min_{\boldsymbol{\omega}} \left(\frac{1}{2}\|\boldsymbol{\omega}\|^2 + \frac{C}{\ell} \sum_{k=1}^{\ell} \max_{y \in \mathcal{Y}} \left\{ \Delta(y_k, y) + \boldsymbol{\omega}^T \Psi(\mathbf{X}, y) - \boldsymbol{\omega}^T \Psi(\mathbf{X}, y_k) \right\} \right) \tag{3}$$

Note that y denotes the trained field of class assignments to each superpixel, whereas y_k refers to the annotated field of class assignments to each superpixel in the k^{th} image of the training set. The function $\Delta(y_k, y)$ computes the zero-one loss, which evaluates to zero when y equals y_k and one otherwise. The regularization parameter C can be chosen freely and goes to reduce the risk of overfitting the training dataset. It controls the balance between the two terms in Equation 3. A lower value of C assigns a relatively larger penalty to the sum of $\boldsymbol{\omega}$ (the first term), which reduces the range of values in $\boldsymbol{\omega}$ and reduces the difference in importance of individual features ($\omega^{(u)}$) and class combinations ($\omega^{(p)}$) accordingly. Hence a lower value of C means more regularization and, following a similar reasoning, a higher value of C means less regularization. As Equation 3 is not differentiable everywhere, the one-slack formulation of the regularized cost function as implemented by *Mueller and Behnke* (2014) is used. Finally, the optimization is performed using the cutting plane algorithm. More details can be found in *Nowozin and Lampert* (2011).

2.5. Model evaluation

The performance of the model can be expressed as a percentage of superpixel classes that have been predicted correctly by the trained SSVM with respect to the manual annotation. However, doing these predictions on images that have been part of the model training dataset would result in a too high percentage of correctly predicted superpixels, as these images have actually been used to fit $\boldsymbol{\omega}$. The key to model evaluation is determining to what extent the model generalizes to unseen images. Hence, the full dataset of 192 annotated coastal images is split into a part that is used for training the model (75% of all images) and a part that is used for testing (25%) as done in many machine learning applications (*Koller and Friedman*, 2009). As the performance of the trained model on predicting the validation set might depend on the specific train-test partition that is chosen, five different partitions have been applied in this study. This provides a way of expressing the model performance as a statistical parameter with a mean and standard deviation depending on the exact partitioning (Figure 2, step 4).

2.6. Software

The methodology used is implemented in an open-source Python toolbox called FLAMINGO (*Hoonhout and Radermacher*, 2015). The toolbox provides tools for (over)segmentation, classification and rectification (projection in real-world coordinate system, see subsection 3.2) of images. It also provides a file and configuration structure for storing datasets and corresponding analyses and a graphical user interface for manual annotation of images. Among others, the toolbox relies on the PYSTRUCT toolbox (*Mueller and Behnke*, 2014), the

SCIKIT-IMAGE toolbox (*van der Walt et al.*, 2014) for image classification and the OPENCV toolbox (*Bradski*, 2000) for image rectification. The toolbox is also published through the OpenEarth initiative (`http://www.openearth.eu`, *van Koningsveld et al.*, 2010).

3. Results

In this section, the performance of the model described in section 2 is evaluated based on the 192 image ArgusNL dataset (*Hoonhout and Radermacher*, 2014). Furthermore, two case studies are assessed as to demonstrate the capabilities and applications of the trained model.

3.1. Testing

A common way to summarize the results of a classification task is the confusion matrix, which presents counts of all testing instances (i.e. individual superpixels in all testing images) based on their actual class (ground truth labeling by manual annotation) and the class predicted by the model. Summed over all partitions, this yields approximately 140,000 testing instances as shown in Figure 8. The percentages in the figure are normalized per row. All entries on the diagonal of the confusion matrix represent correct predictions by the model, accounting for an average accuracy of 93.0% with a standard deviation of 0.7%.

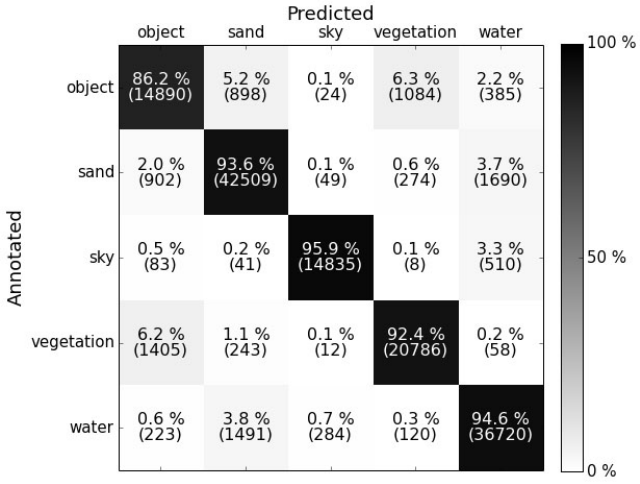

Figure 8: Aggregated confusion matrix over all partitions. The percentages represent the sensitivity of the model. The values between parentheses are absolute superpixel counts.

These percentages can be elaborated further to evaluate the models ability to correctly predict specific class labels. To this end, various measures are used, among others precision and sensitivity. The precision of a model with respect to class X is the percentage of superpixels, out of all superpixels predicted as X,

Table 1: Model precision, sensitivity, F1-scores and occurrence. Values are averaged over all partitions. Standard deviations are between parentheses.

	Prec.	**Sens.**	**F1**	**Occ.**
Object	85.1 (2.0)	85.6 (3.1)	85.3 (3.2)	12.4 (2.3)
Sand	94.1 (0.8)	93.6 (0.6)	93.8 (0.6)	32.5 (3.9)
Sky	97.7 (1.2)	95.7 (1.3)	96.7 (0.5)	11.1 (1.0)
Vegetation	93.4 (1.9)	92.3 (0.6)	92.8 (0.92)	16.2 (3.4)
Water	93.5 (1.0)	94.3 (1.8)	93.9 (1.3)	27.8 (5.0)

that actually has the ground truth label X. It can be computed by normalizing every entry in the diagonal of the confusion matrix with the sum of its column. The sensitivity of a model (also known as recall) with respect to class X is the percentage of superpixels, out of all superpixels having ground truth label X, that have been predicted as X. To compute sensitivity, every entry in the diagonal of the confusion matrix should be normalized with the sum of its row (as was already done for the percentages given in Figure 8).

In the first two columns of Table 1, precision and sensitivity of all 5 classes are presented, averaged over the 5 partitions. The standard deviation is given between parentheses. It is noted that evaluating these parameters for every partition and presenting the average value leads to slightly different values than the sensitivity as evaluated for the cumulative confusion matrix in Figure 8. In order to combine precision and sensitivity into a single performance metric, the F1-score was proposed by *van Rijsbergen* (1979). Simply taking the average of both values would mask cases where one is very high and the other is very low, e.g. in a model that monotonously predicts every superpixel as *sand*, the class *sand* has 100% sensitivity, but very low precision. Instead, the harmonic mean is taken according to Equation 4, where P is precision and S is sensitivity.

$$F1 = 2\frac{PS}{P+S} \tag{4}$$

The third column in Table 1 lists the average F1-scores for all classes, including their standard deviation between parentheses. The occurrence (i.e. the percentage of the ground truth data having class label X) is added to characterize the composition of the testing dataset. Altogether, the model is specifically good at recognizing *sky*, *water*, *sand* and *vegetation* and has most difficulties in recognizing *objects*. The confusion matrix shows that, at most occasions, *vegetation* or *sand* were erroneously recognized as *objects* and vice versa. None of the classes exhibit discrepancy between precision and sensitivity values, which implies that the model is well-balanced.

Table 2: Top 5 features of class *water*. Values are averaged over all partitions. Standard deviations are between parentheses.

Feature	$\overline{\omega^{(u)}}$	
mean intensity 1	-3.82	(0.58)
min intensity 3	-3.20	(0.40)
max intensity	2.52	(0.52)
weighted moments normalized 7.2	2.48	(0.51)
mean intensity 3	2.47	(0.28)

The trained unary potentials ω^u give information about the strength of every feature in distinguishing every class from the other classes. A full analysis of feature strength would involve too much detail to treat within the scope of this study, but as an example the top 5 features of the class *water* are given in Table 2. Features are ranked by absolute value of the unary potential. The overbar denotes averaging over all 5 partitions. The associated standard deviations over the partitions are shown between parentheses. The top feature, *mean_intensity.1*, represents the superpixel-averaged intensity of the first image channel, being the red channel. The strong negative unary potential indicates that superpixels of class *water* tend to have low redness, a property that was employed by *Turner and Leyden* (2000) for their shoreline detection algorithm. Furthermore, the fifth ranked feature reflects a property of *water* that corresponds most with the human perception: water is blue. However, the negative potential of the second feature indicates that uniformly blue pixels are not likely to be of class *water*. The third feature, the maximum brightness in a superpixel, averaged over the three intrinsic color channels, might refer to the presence of bright foam patches associated with wave breaking. The fourth ranked feature describes a far more complex property that is less intuitive to humans, but, judging from its strong unary potential, is of great help in distinguishing *water* from other classes.

In addition, the pairwise potentials can be studied to obtain information about the likelihood of class-pairs being assigned to neighboring superpixels. The upper half of the symmetric pairwise potential matrix is given in Table 3. Several intuitive inter-class relationships can be derived from the matrix, like the fact that *sand* is likely to border *water*, *vegetation* and *objects*, but not likely to border *sky*. However, some of the pairwise potentials require a more complicated interpretation, that goes to show the sophistication of a structured classification model like a SSVM. For example, the class *sky* only has negative pairwise potentials, implying that predicting *sky* next to any other superpixel is penalized by the model. Apparently, the model is very capable of predicting *sky* solely based on the unary potentials, such that it can afford the penalty from the pairwise potentials. By setting a penalty on the pairwise potential, the model prevents *sky* from being erroneously predicted at a location where the superpixel features are only vaguely representing *sky*, but where the classes of surrounding superpixels make the prediction of *sky* fairly likely.

Table 3: Pairwise potentials

	Object	*Sand*	*Sky*	*Vegetation*	*Water*
Object	3.32	2.16	-2.63	2.72	-0.74
Sand		3.19	-4.60	2.12	1.60
Sky			-3.56	-3.58	-3.77
Vegetation				2.99	-1.70
Water					2.50

To illustrate the performance of the model, two examples of predicted images are given in Figure 9. The top row represents an image that was well predicted by the model, having 97.6% of its class labels predicted correctly. The snapshot was taken on a bright day with a very calm sea, resulting in unambiguous and low noise feature values. The bottom row illustrates what happens if the model makes predictions on a snapshot with more difficult weather conditions. Raindrops and dirt on the camera lens obscure the image, leading to significantly different feature values than those that dominate the training dataset. Although the main outline of the predicted image is still correct, the model performance breaks down around class interfaces and clusters of erroneous predictions appear in the multicolored residential area.

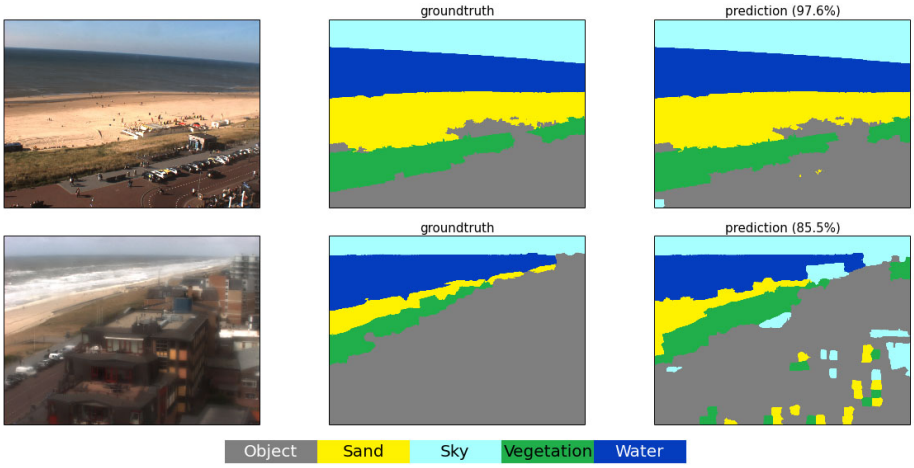

Figure 9: Examples of good (top) and bad (bottom) model performance. The original image is shown in the leftmost column, together with the ground truth labeling (middle column) and model prediction (rightmost column).

As mentioned in subsection 2.4 the model can be fine-tuned using a regularization parameter C. In this study a moderate regularization of $C = 1$ is used without further optimizing this parameter. Optimization of the regularization parameter would require, next to the training and test dataset currently used, an additional validation dataset with manually annotated images. The test set could be split into a validation and test set, but it was chosen not to do so to

prevent the sets from becoming too small. Training the model using different values for C, however, shows that the chosen value of 1 is not far from the optimal value and that the model does not suffer much from under- or overfitting (Figure 10). It should be noted that applying the algorithm on a different dataset using images from different stations and/or with different classes may cause problems with under- or overfitting and hence may require optimization of the regularization parameter.

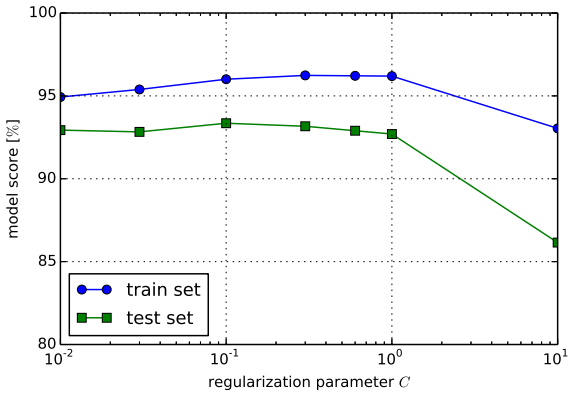

Figure 10: Model performance over a single partition depending on the value of the regularization parameter C.

3.2. Case study: Beach width and waterline

The range of possible applications of semantic classification of regions in coastal images is very wide. Two case studies are treated in this section to illustrate the application of the algorithm. In the next section alternative applications are discussed.

3.2.1. Beach width

Beach width is regarded one of the most important parameters for coastal managers (*Kroon et al.*, 2007). Eroding beaches and, thus, decreasing beach width raise concerns about coastal safety against flooding at locations all over the world. Furthermore, many coastal regions owe the biggest part of their revenue to recreational tourism at their beaches. Generally a wider beach provides higher safety and can accommodate a larger number of visitors.

For this case study, a two year dataset of fortnightly images with water levels around mean sea level (MSL) was composed, using images from the Kijkduin Argus station. The trained SSVM presented in subsection 3.1 was used to classify these images. Using camera physics, the classified images were projected onto a horizontal x-y plane at MSL (*Holland et al.*, 2001), as depicted in Figure 11. Beach width was determined in every transect of the projected image by taking the cross-shore distance spanned by superpixels of class *sand* (constrained to

the central part of the projection, where the full cross-shore extent of the subaerial beach is covered in the projected image). The predicted beach width was reduced to a single value per image by taking the median over all transects, so as to reduce scatter by erroneously classified superpixels. The resulting evolution of beach width over time is presented in Figure 12. Without any quality control, a linear trend is fitted through the data points using least squares (solid line in the figure). The trend line coincides with the main body of the data and indicates a slightly eroding trend (8.4 m/year). Groundtruth data determined from RTK-DGPS measurements with an all-terrain vehicle were added to Figure 12 for reference. To avoid randomness of the exact cross-shore transect location, 7 in-situ transects coinciding with the image view were selected. Beach width was extracted from these data by taking the distance between the C.D. +4m and 0m marks. The mean beach width over 7 transects is shown in the figure, including error bars of 2 times the standard deviation on either side. The beach width derived from coastal imagery coincides fairly well with the in-situ data, although the linear eroding trend of the latter (dashed line) is twice as strong (16.9 m/year).

Scatter in the extracted beach widths is caused by several error sources. First, erroneous class assignments, which either predict non-*sand* superpixels as *sand* (overprediction of beach width) or vice-versa (underprediction of beach width), cause a limited number of distinct outliers up to $O(100\text{m})$. Second, by using (generic) snapshot images the image data include variations in hydrodynamic conditions. Variations in water line position due to the tidal phase are canceled out by using only images with water levels around MSL. Variations in water line position due to wave run-up are limited to $O(10\text{m})$ during the observed period based on an empirical run-up formulation (*Hunt*, 1959) and measured wave heights and periods. Third, inaccurate segmentation may cause the algorithm to fail providing pixel precise results, which introduces errors of $O(1\text{m})$ considering that the image resolution around the water line is $O(0.1\text{m})$.

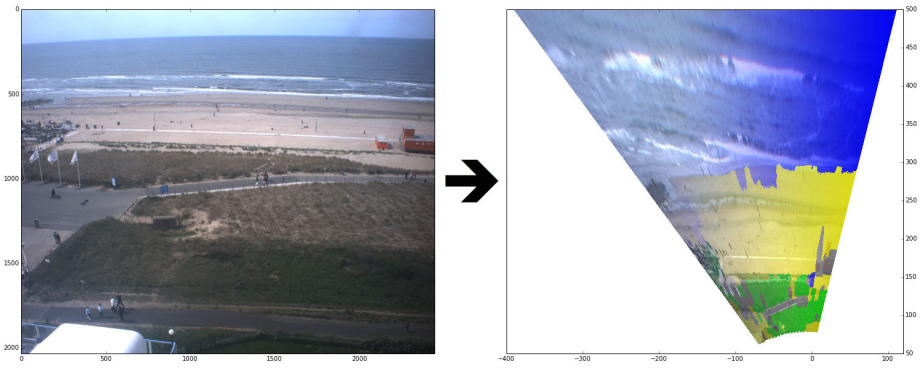

Figure 11: Original coastal image (left) and rectified coastal image that is projected on a horizontal x-y plane (right), including classification result (shaded colors).

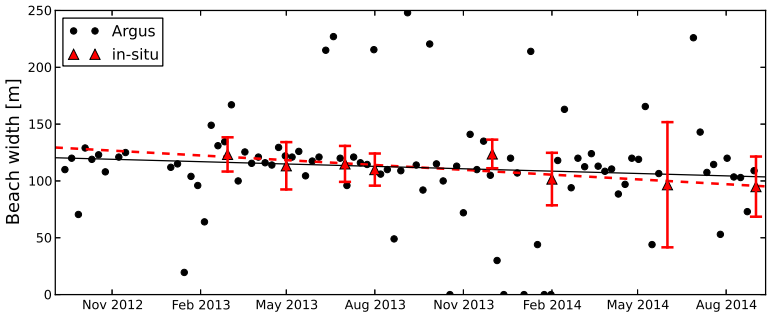

Figure 12: Trend in beach width determined using image classification (black dots and line), compared to in-situ measurements (red triangles and line) in Kijkduin, The Netherlands. Red error bars indicate a range of plus or minus one standard deviation of the in-situ beach width found over the stretch of beach in the camera view.

3.2.2. Waterline

Although the primary product of coastal image classification is the separation of different *areas*, the border between two areas is often a useful product as well. The instantaneous waterline, being the border between the *sea* and *beach* area, is an important indicator for coastal science. By tracking the waterline over a sequence of images the width of the intertidal area can be determined. Also, relating the instantaneous position of the waterline to the tidal elevation provides a three-dimensional bathymetry of this area (e.g. *Aarninkhof et al.*, 2003; *Quartel et al.*, 2006; *Uunk et al.*, 2010; *Osorio et al.*, 2012).

A series of images from the Kijkduin Argus station at May 4th, 2014 is used for this case study. The Kijkduin Argus station consists of 6 cameras, 4 out of which have the intertidal zone in view with sufficient resolution to extract instantaneous water lines. The trained SSVM presented in subsection 3.1 was used to classify these images. The images are projected into a horizontal x-y plane at MSL (Figure 11) and the waterline is simply determined by finding the first beach pixel seen from offshore. For each of these pixels the bed level is determined based on RTK-DGPS measurements obtained frequently in this area. Consequently a series of points is obtained in three-dimensional space that are supposedly located on the instantaneous waterline. From these points the median value is taken in order to exclude outliers, caused by erroneously classified pixels, to obtain an estimate of the instantaneous water elevation at the water line.

Figure 13 shows the astronomical tide measured in Scheveningen (5km north of Kijkduin) and the waterline estimates based on a 24-hour sequence of coastal images from the Kijkduin Argus station. The course of the astronomical tide is captured reasonably well, but also some scatter is observed. The scatter can partially be explained by physical processes that decouple the instantaneous water level from the position of the water line. Wave setup and runup cause an

additional elevation of the water line compared to the still water level. Local differences in morphology influence the wave propagation and hence the alongshore and temporal differences in water line location. Also, the classification procedure provides us with a location of the water line in a horizontal plane. In order to compare this location to the vertical tide the bed level along this line is determined based on a measured bathymetry. This bathymetry has a much coarser resolution (20m alongshore) than most of the image pixels. Therefore this conversion is an approximation and a source of errors.

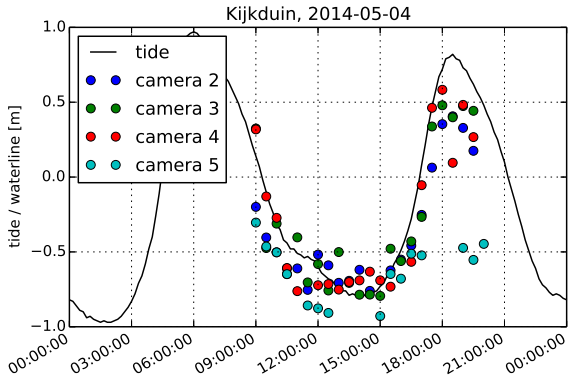

Figure 13: Waterline estimates based on a 24-hour sequence of coastal images in Kijkduin, The Netherlands

Despite the very generic approach, without any focus on particular classes or regions of interest, and without any form of quality control, the proposed classification technique already provides sensible and useful data from raw coastal images. Yet several improvements, that are outside the scope of this study, can be made. For example, both case studies were based on raw snapshot images that contain considerable noise, like variations in hydrodynamic conditions and anthropogenic activities. By using derived image products, like time-averaged images much noise can be removed and more homogeneous classes can be learned improving the classification performance.

Another example is that the superpixels used in these case studies are large compared to the width of the intertidal beach area and may cover 10 meter or more. Using a finer segmentation is expected to improve the performance considerably. Superpixels that constitute the border of the waterline might even be classified on a pixel-by-pixel basis to overcome initial errors made in the segmentation step. Moreover, the segmentation step itself may be improved as well by using the fact that a series of images with temporal correlations between them is observed, to ensure that water line movement is continuous in

time. Supervoxels[2] can be used for this purpose rather than superpixels. All of these improvements can be applied within the proposed algorithm.

4. Discussion

The development of the algorithm initially started to track beach widths over time and improve the robustness of algorithms for waterline detection and intertidal bathymetry determination. The genericity of the algorithm, however, sparked many possible additional applications within the very same framework. Among these are vegetation coverage determination, people counters, aeolian feature detection, digital elevation map correction, quality control, et cetera.

Obviously no algorithm is perfect and also this algorithm has some pitfalls. First the segmentation step is recognized as a weak link in the procedure. The segmentation step determines in an early stage what pixels will be assigned the same class. An error made in this step can cause a wrong classification on pixel level easily. For example, when a white patch of foam is clustered together with a white portion of the beach, it will be impossible to accurately detect the instantaneous waterline. Currently, segmentation is only done based on the intrinsic image features (RGB, HSV, etc.) and is therefore susceptible to errors. Another pitfall is that the algorithm is particularly good at classifying areas. Due to the structured learning (the inclusion of pairwise potentials) it is unlikely that classification errors are made in the center of such an area, for example, that a white patch in the middle of the sea is predicted to be beach. Hence, most errors are made in the vicinity of the borders, which is important to realize when, for example, applying the algorithm to waterline detection (see also Figure 9). For applications like waterline detection one can think of performing a-posteriori reclassification. After a superpixel boundary has been classified as a sand-water boundary, binary classification (sand-water) can be applied to smaller superpixels or individual pixels surrounding the boundary.

Despite these pitfalls that still exist, the algorithm is proven to be generic and robust and shows unprecedented capability of classifying arbitrary pixels in coastal imagery. Moreover, most pitfalls have still potential for improvement. Segmentation is now based on intrinsic image features, but the filter channels, including information on edges and textures, can also be used in the segmentation procedure. Similarly, superpixels bordering an area may be classified on a pixel-by-pixel basis using the very same algorithm to obtain pixel precise classification.

The genericity of the algorithm is also apparent from the fact that very similar algorithms are used in other fields of research that cope with large amounts of image data, like remote sensing (*Mountrakis et al.*, 2011). Satellite images are automatically classified according to land use or vegetation type (e.g. Figure 14) and in medical sciences, automated classification of histology imagery

[2] A supervoxel is a segmentation of the space-time domain, while a superpixel is a segmentation of the space domain alone

is used to find malignant tumors in an early stage (e.g. *Belsare and Mushrif*, 2012).

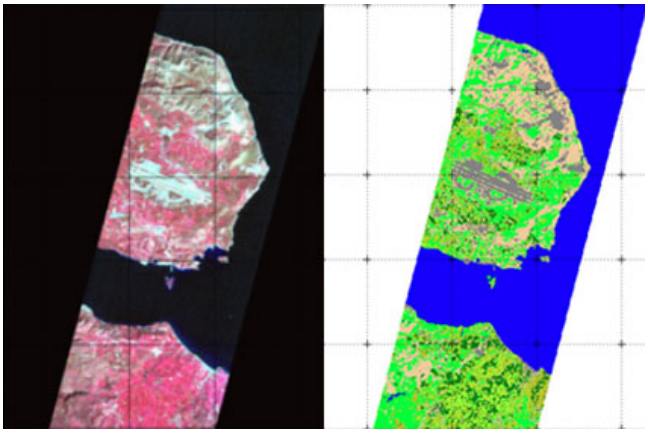

Figure 14: Satellite image (left) and Land-use classification (right) of prefecture of Chania on the island of Crete, in Greece (*Petropoulos et al.*, 2012)

Compared to existing coastal image analysis techniques the proposed algorithm uses an abundance of pixel features and relations between neighboring superpixels to distinguish one class of pixels from another. Figure 15 shows a simplified representation of the algorithms evolution with respect to its percentage of correctly predicted pixels in the testing dataset, starting from classification using intrinsic pixel features only. After segmentation the number of features is boosted, increasing the overall model performance considerably. The introduction of complex texture and filter features, filter channels and structured learning made the algorithm perform on the 93.0% accuracy level it is now.

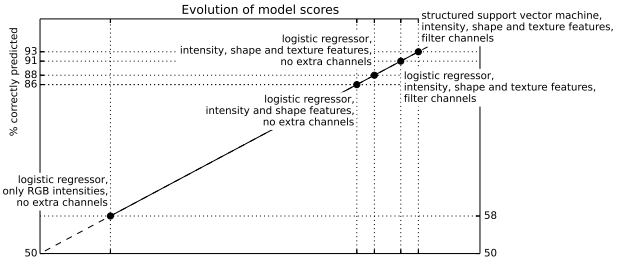

Figure 15: Evolution of model scores averaged over all 5 partitions after increasing the number of features, filter channels and introducing structured learning. Note that the actual effectiveness cannot be determined from this graph since the order of improvements matters.

Another major difference between the proposed algorithm and existing algorithms for coastal image analysis is the fact that it fully operates in pixel space,

rather than in real-world coordinates. For many coastal science and management applications it is useful to map an image to real-world coordinates, but from the algorithms perspective it is not necessary. This enables additional applications that do not require image rectification, like people counting, automated cropping and quality control. Moreover, objects and regions in an image are deformed differently during rectification depending on their position in the camera view due to varying real-world pixel resolution throughout the image. This effect might deteriorate the performance of classification algorithms that are based on rectified images.

5. Conclusions

The applicability of many existing algorithms for coastal image analysis is limited for the large, long-term coastal imagery datasets that are presently available for coastal research, since they generally lack automation and robustness. Site- and time-dependent calibration is often required. In this paper a fully automated algorithm is presented that can cope with these large coastal imagery datasets and classifies each pixel in a given image into a pre-defined set of classes.

The key to success of the approach presented here is the abundance of image information that is used for discrimination of pixel classes. This increase is achieved by combining oversegmentation of the images with extensive feature extraction. Instead of relying solely on intrinsic pixel information, a set of 1727 pixel features is extracted for every pixel in the dataset and used to distinguish between five classes of pixels: *object*, *sand*, *sky*, *vegetation* and *water*. Furthermore, a structured machine learning approach was adopted in order to take relations between neighboring pixels into account. Using a manually annotated dataset of 192 coastal images, a Structured Support Vector Machine (SSVM) is trained and tested to cast predictions on the class of every pixel in the dataset. The resulting model correctly classifies 93.0% of all pixels in an unseen image on average with a standard deviation of 0.7% computed over 5 train/test partitions. The model performs well over all classes, which is indicated by the F1-score of 93% or more for all classes, except for the versatile class *objects* that is still reasonably accurate with 85%.

By analyzing the trained feature weights it was shown that the model uses class characteristics that are very similar to human intuition, like *water* is blue and heterogeneous, but also relies on more sophisticated features that are less obvious for humans to interpret.

The application of the algorithm was illustrated using two case studies that showed how this generic algorithm can be applied to extract useful data like beach widths or water lines from a coastal camera station without any form of manual quality control. This opens up new possibilities to analyze large, long-term datasets of coastal imagery and to apply these algorithms to different types of coastal images, including webcams and mobile cameras. Both the annotated dataset and the software developed to perform the model training and prediction are provided open-source and free of charge.

Acknowledgments

For their work discussed in this paper Hoonhout and Baart are supported by the ERC-Advanced Grant 291206 – Nearshore Monitoring and Modeling (NEMO). Radermacher is supported by STW Grant 12686 – Nature-driven Nourishments of Coastal Systems (NatureCoast), S1: Coastal safety.

Aarninkhof, S. G., I. L. Turner, T. D. Dronkers, M. Caljouw, and L. Nipius (2003), A video-based technique for mapping intertidal beach bathymetry, *Coastal Engineering*, *49*(4), 275–289, doi:10.1016/S0378-3839(03)00064-4.

Aarninkhof, S. G. J., B. G. Ruessink, and J. A. Roelvink (2005), Nearshore subtidal bathymetry from time-exposure video images, *Journal of Geophysical Research*, *110*(C6), C06,011, doi:10.1029/2004JC002791.

Achanta, R., A. Shaji, K. Smith, and A. Lucchi (2012), SLIC Superpixels Compared to State-of-the-Art Superpixel Methods, *Transactions on pattern analysis and machine intelligence*, *34*(11), 2274–2281.

Belsare, A. D., and M. M. Mushrif (2012), Histopathological Image Analysis Using Image Processing Techniques: An Overview, *Signal & Image Processing : An International Journal*, *3*(4), doi:doi:10.5121/sipij.2012.340323.

Bogle, J. A., K. R. Bryan, K. P. Black, T. M. Hume, and T. R. Healy (2000), Video observations of rip formation and evolution, *Journal of coastal research*, *34*, 117–127.

Bradski, G. (2000), The opencv library, *Dr. Dobb's Journal of Software Tools*.

Chickadel, C. C., R. A. Holman, and M. H. Freilich (2003), An optical technique for the measurement of longshore currents, *Journal of Geophysical Research*, *108*(C11), 17, doi:10.1029/2003JC001774.

Crowley, J. L., and A. C. Parker (1984), A representation for shape based on peaks and ridges in the difference of low-pass transform, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *PAMI-6*(2), 156–170.

Davidson, M., M. Van Koningsveld, A. De Kruif, J. Rawson, R. A. Holman, A. Lamberti, R. Medina, A. Kroon, and S. G. J. Aarninkhof (2007), The CoastView project: developing video-derived Coastal State Indicators in support of coastal zone management, *Coastal Engineering*, *54*, 463–475.

Dusek, G., and H. Seim (2013), A probabilistic rip current forecast model, *Journal of Coastal Research*, *29*(4), 909–925.

Gabor, D. (1946), Theory of communication. part 1: The analysis of information, *Journal of the Institution of Electrical Engineers - Part III: Radio and Communication Engineering*, *93*, 429–441(12).

Gallop, S. L., K. R. Bryan, and G. Coco (2009), Video observations of rip currents on an embayed beach, *Journal of Coastal Research*, *56*, 49–53.

Guillén, J., A. García-Olivares, E. Ojeda, A. Osorio, O. Chic, and R. González (2008), Long-Term Quantification of Beach Users Using Video Monitoring, *Journal of Coastal Research*, *24*(6), 1612–1619, doi:10.2112/07-0886.1.

Haralick, R., K. Shanmugam, and I. Dinstein (1973), Textural features for image classification, *Systems, Man and Cybernetics, IEEE Transactions on*, *SMC-3*(6), 610–621, doi:10.1109/TSMC.1973.4309314.

Holland, K., and R. Holman (1991), Measuring run-up on a natural beach, *EOS Trans Am Geophys Union*, *72*, 254.

Holland, K., J. Puleo, and T. Kooney (2001), Quantification of swash flows using video-based particle image velocimetry, *Coastal Engineering*, *44*(2), 65–77, doi:10.1016/S0378-3839(01)00022-9.

Holman, R., N. Plant, and T. Holland (2013), cBathy: A robust algorithm for estimating nearshore bathymetry, *Journal of Geophysical Research: Oceans*, *118*(5), 2595–2609, doi:10.1002/jgrc.20199.

Hoonhout, B. M., and M. Radermacher (2014), Annotated images of the dutch coast, FTP server, doi:10.4121/uuid:08400507-4731-4cb2-a7ec-9ed2937db119.

Hoonhout, B. M., and M. Radermacher (2015), Flamingo: a coastal image analysis toolbox, GIT repository, doi:10.5281/zenodo.14596.

Hoonhout, B. M., F. Baart, and J. S. M. Van Thiel de Vries (2014), Intertidal beach classification in infrared images, *Journal of Coastal Research*, *Special Issue No. 66*, in: Green, A.N. and Cooper, J.A.G. (eds.), Proceedings 13th International Coastal Symposium (Durban, South Africa) ISSN 0749-0208.

Hunt, I. A. (1959), Design of seawalls and breakwaters, *Journal of Waterways and Harbours Division*, *85*(WW3), 123–152.

Jimenez, J. A., A. Osorio, I. Marino-Tapia, M. Davidson, R. Medina, A. Kroon, R. Archetti, P. Ciavola, and S. G. J. Aarninkhof (2007), Beach recreation planning using video-derived coastal state indicators, *Coastal Engineering*, *54*, 507–521.

Kingston, K. S. (2003), Applications of complex adaptive systems approaches to coastal systems, Ph.D. thesis, University of Plymouth.

Koller, D., and N. Friedman (2009), *Probabilistic graphical models*, 1280 pp., MIT Press, Cambridge, MS, USA.

Kroon, A., M. A. Davidson, S. G. J. Aarninkhof, R. Archetti, C. Armaroli, M. Gonzalez, S. Medri, A. Osorio, T. Aagaard, R. A. Holman, and R. Spanhoff (2007), Application of remote sensing video systems to coastline management problems, *Coastal Engineering*, *54*, 493–505.

Lippmann, T. C., and R. A. Holman (1989), Quantification of sand bar morphology: a video technique based on wave dissipation, *Journal of Geophysical Research*, *94*, 995–1011.

Lloyd, S. P. (1982), Least squares quantization in pcm, *IEEE Transactions on Information Theory*, *28*(2), 129–137, doi:doi:10.1109/TIT.1982.1056489.

Lu, D. (2005), Aboveground biomass estimation using landsat tm data in the brazilian amazon, *International Journal of Remote Sensing*, *26*(12), 2509–2525, doi:10.1080/01431160500142145.

Mountrakis, G., J. Im, and C. Ogole (2011), Support vector machines in remote sensing: A review, *ISPRS Journal of Photogrammetry and Remote Sensing*, *66*, 247–259, doi:doi:10.1016/j.isprsjprs.2010.11.001.

Mueller, A., and S. Behnke (2014), Pystruct - learning structured prediction in python, *Journal of Machine Learning Research*, *15*, 2055–2060.

Nowozin, S., and C. Lampert (2011), Structured learning and prediction in computer vision, *Foundations and trends in computer graphics and vision*, *6*, 185–365.

Osorio, A., R. Medina, and M. Gonzalez (2012), An algorithm for the measurement of shoreline and intertidal beach profiles using video imagery: PSDM, *Computers & Geosciences*, *46*, 196–207, doi:10.1016/j.cageo.2011.12.008.

Petropoulos, G. P., C. Kalaitzidis, and K. P. Vadrevu (2012), Support vector machines and object-based classification for obtaining land-use/cover cartography from Hyperion hyperspectral imagery, *Computers & Geosciences*, *41*, 99–107, doi:doi:10.1016/j.cageo.2011.08.019.

Plant, N. G., and R. A. Holman (1997), Intertidal beach profile estimation using video images, *Marine Geology*, *140*, 1–24.

Plant, N. G., S. G. J. Aarninkhof, I. L. Turner, and K. S. Kingston (2007), The performance of shoreline detection models applied to video imagery, *Journal of Coastal Research*, *223*, 658–670.

Price, T. D., and B. G. Ruessink (2011), State dynamics of a double sandbar system, *Continental Shelf Research*, *31*(6), 659–674, doi:10.1016/j.csr.2010.12.018.

Quartel, S., E. Addink, and B. Ruessink (2006), Object-oriented extraction of beach morphology from video images, *International Journal of Applied Earth Observation and Geoinformation*, *8*(4), 256–269, doi:10.1016/j.jag.2006.01.002.

Sekovski, I., F. Stecchi, F. Mancini, and L. Del Rio (2014), Image classification methods applied to shoreline extraction on very high-resolution multispectral imagery, *International Journal of Remote Sensing*, *35*(10), 3556–3578, doi:10.1080/01431161.2014.907939.

Smit, M. W. J., S. G. J. Aarninkhof, K. M. Wijnberg, M. Gonzalez, K. S. Kingston, H. N. Southgate, B. G. Ruessink, R. A. Holman, E. Siegle, M. Davidson, and R. Medina (2007), The role of video imagery in predicting daily to monthly coastal evolution, *Coastal Engineering*, *54*, 539–553.

Turner, I. L., and V. M. Leyden (2000), System description and analysis of shoreline change: August 1999 - february 2000; report 1: Northern gold coast coastal imaging system, *Tech. rep.*, University of New South Wales, Manly Vale (NSW), Australia.

Uunk, L., K. M. Wijnberg, and R. Morelissen (2010), Automated mapping of the intertidal beach bathymetry from video images, *Coastal Engineering*, *57*(4), 461–469, doi:10.1016/j.coastaleng.2009.12.002.

van der Walt, S., J. L. Schönberger, J. Nunez-Iglesias, F. Boulogne, J. D. Warner, N. Yager, E. Gouillart, T. Yu, and the scikit-image contributors (2014), scikit-image: image processing in Python, *PeerJ*, *2*, e453, doi: 10.7717/peerj.453.

van Dongeren, A., N. Plant, A. Cohen, D. Roelvink, M. C. Haller, and P. Catalán (2008), Beach wizard: Nearshore bathymetry estimation through assimilation of model computations and remote observations, *Coastal Engineering*, *55*, 1016–1027.

van Enckevort, I. M. J., and B. G. Ruessink (2001), Effect of hydrodynamics and bathymetry on video estimates of nearshore sandbar position, *Journal of Geophysical Research*, *106*(C8), 16,969–16,979.

van Koningsveld, M., G. J. de Boer, F. Baart, T. Damsma, C. den Heijer, P. van Geer, and B. de Sonneville (2010), Openearth - inter-company management of: Data, models, tools and knowledge, in *Proceedings WODCON XIX Conference : Dredging Makes the World a Better Place*, Beijing, China.

van Rijsbergen, C. J. (1979), *Information retrieval*, 2 ed., Butterworths.

Verhaeghe, H., J. De Rouck, and J. Van der Meer (2008), Combined classifier-quantifier model: A 2-phases neural model for prediction of wave overtopping at coastal structures, *Coastal Engineering*, *55*, 357–374.

Vincent, P. (1986), Differentiation of modern beach and coastal dune ssand - a logistic regression approach using the parameters of the hyperbolic function, *Sedimentary Geology*, *49*, 167–176.

Acronyms

LR Logistic Regressor

CRF Conditional Random Field

SVM Support Vector Machine

SSVM Structured Support Vector Machine

RGB Red, Green and Blue

HSV Hue, Saturation and Value

LAB Lightness, color-opponent A, color-opponent B

CIELAB Commission Internationale de l'Éclairage LAB