Automated shoreline extraction process for unmanned vehicles via U-net with heuristic algorithm

Article *in* Alexandria Engineering Journal · September 2024

DOI: 10.1016/j.aej.2024.05.104

CITATIONS
0

READS
27

6 authors, including:

Katarzyna Prokop
Silesian University of Technology
13 PUBLICATIONS **75** CITATIONS
SEE PROFILE

Dawid Połap
Silesian University of Technology
156 PUBLICATIONS **3,512** CITATIONS
SEE PROFILE

Marta Wlodarczyk-Sielicka
Maritime University of Szczecin
39 PUBLICATIONS **504** CITATIONS
SEE PROFILE

Andrzej Stateczny
Gdynia Maritime University
114 PUBLICATIONS **1,884** CITATIONS
SEE PROFILE

Alexandria Engineering Journal 102 (2024) 108–118

Contents lists available at ScienceDirect

Alexandria Engineering Journal

journal homepage: www.elsevier.com/locate/aej

Original article

Automated shoreline extraction process for unmanned vehicles via U-net with heuristic algorithm

Katarzyna Prokop [a], Dawid Połap [a,*], Marta Włodarczyk-Sielicka [b], Karolina Połap [c], Antoni Jaszcz [a,c], Andrzej Stateczny [d]

[a] Faculty of Applied Mathematics, Silesian University of Technology, Kaszubska 23, 44-100 Gliwice, Poland
[b] Faculty of Navigation, Maritime University of Szczecin, Waly Chrobrego 1-2, 70-500 Szczecin, Poland
[c] Marine Technology Ltd., Roszczynialskiego 4/6, 81-521 Gdynia, Poland
[d] Gdynia Maritime University, 81-87 Morska St. 81-225 Gdynia, Poland

ARTICLE INFO

Keywords:
Shoreline detection
U-net
Heuristic
Adaptive method
Automated solution

ABSTRACT

Detecting the shoreline is an important task for its potential use. The shoreline allows cropping of the image into two separate areas that present the water area and the shore. It is particularly interesting because the images can be used to analyze pollution, land development, or even waterfront erosion. Unfortunately, automatic shoreline detection is a complex problem due to numerous physical and atmospheric issues. In this paper, we present a solution based on a U-net convolutional network, that is trained to shoreline detection on a dedicated database. The database is automatically generated by applying image processing techniques and a heuristic algorithm. Using heuristics, optimal values of mask generation parameters are determined. Consequently, the solution allows for the automation of generating a set of masks by analyzing the boundary line and the efficiency of the segmentation network. The proposed solution allows for the analysis of the coastline, where potential obstacles and even occurring waves can be quickly detected. To evaluate the proposed solution, tests were carried out in real conditions, which showed the effectiveness of the model. In addition, tests were carried out on a publicly available database, which allowed for obtaining higher results than existing methods.

1. Introduction

Cameras are often used to analyze coastal and water areas [1]. It is a relatively cheap solution enabling the execution of precise graphic images in the form of several video frames. In the case of capturing an area from a boat, quite often the frame includes part of the ship, water, shore with objects located on it, and the sky. Quite often there is a need to separate redundant areas in a given image. In the case of coastal analysis, the water part, as well as the boat, are redundant. Hence, an important action is to extract only the area of interest.

Extracting the selected area is not the easiest task in terms of technical and implementation [2]. First of all, attention should be paid to the recorded image. Various weather conditions, or even the camera positioning, may cause the software to malfunction [3]. An example is too much reflection of the sun on the water's surface. The classic approach to image analysis is based on image processing through the use of graphic filters [4]. Graphic filters, such as edge detection or blur allow to obtain specific elements or even to reduce indeterminate (by blurring). However, the modification affects the entire image, not just a fragment. Consequently, this action contributes to image modification and not the final extraction. An alternative approach is to use machine learning methods. U-net networks are interesting because these neural network models process images and return their modified version [5]. This architecture allows for the creation of a tool that will process a given image in a specific way.

Shoreline detection plays a crucial role in addressing contemporary environmental and navigational challenges. Environmental monitoring, coastal management, disaster response and navigation safety all benefit from new solutions facilitating important tasks in those fields. Due to the effects of global warming, the reappearing droughts make the regressing shorelines evident problem [6]. The use of neural networks makes it possible to define the division of the land and water area, i.e. to draw a shoreline, on which the given image can then be trimmed. However, this is not an easy task due to the database. Neural networks are among the data-hungry algorithms that require a huge amount of

* Corresponding author.
E-mail addresses: katarzyna.prokop@polsl.pl (K. Prokop), dawid.polap@polsl.pl (D. Połap), m.wlodarczyk@pm.szczecin.pl (M. Włodarczyk-Sielicka), k.kesik@marinetechnology.pl (K. Połap), aj303181@student.polsl.pl (A. Jaszcz), a.stateczny@wn.umg.egu.pl (A. Stateczny).

https://doi.org/10.1016/j.aej.2024.05.104
Received 15 February 2024; Received in revised form 12 April 2024; Accepted 28 May 2024

data to train such an architecture. The last few years brought many new algorithms for prediction curves [7], analysis of the distribution of objects and plant formations on the coasts [8] and new datasets for improving machine learning models [9,10]. Hence, in this paper, we propose a solution based on the automatic creation of the database and, in the event of a network's low accuracy, its modification. This approach is based on a combination of the classical approach using image processing and machine learning algorithms. The proposed approach is composed of two stages linked together by the verification module. In the first step, classical graphics processing algorithms are used to define the shoreline. Then the entire database is modified and used to train the U-net. In the case of low learning outcomes, the parameters in the first stage are changed (with the help of a heuristic algorithm) and the whole thing is repeated. Additionally, it is possible to influence the user, who can interrupt the current state of operation of the method and evaluate the correctness of the shoreline curve. The main contributions of this research are:

- automation of database generation for a segmentation network,
- framework for the extraction of a selected area in the images related to the lake shore analysis,
- automatic knowledge construction (result images) based on classical image processing and heuristic approach,
- U-net model and verification methods of learning results,
- mechanism of obstacle detection by unmanned vehicle.

2. Related works

The problem of extracting a specific area in a 2D image comes down to the problem of segmentation. The most important element is determining where a specific object or area is located. To divide the image into the water and above water parts, there is a need to find the area that divides them — it can be curved. The process of image analysis based on segmentation finds huge application in the processing of camera, satellite, or even sonar images. An example of such research is optical satellite images [11]. In this research, the authors describe a method composed of two phases, the first of which uses classic image analysis solutions like geometric correction, and noise-removing techniques for image preparation to the final analysis. It was based on using an unsupervised classification method that uses a multi-spectral technique based on the analysis of infrared bands. The obtained research shows that a combination of such methods gives good results.

Image processing techniques are still used which can be seen in [12], where an automatic thresholding algorithm for shoreline detection was described. Again in [13], a geometric active contour model was applied. The idea of the proposal was based on improving the pressure function in that model with a smoothing area to change the curve. Another example of using the classical approach to modeling the shoreline detection algorithm is the combination of edge detection, thresholding and additional filters [14].

Deep learning is one of the most used tools in the analysis of images. In [15], the authors propose the use of U-net with developed loss functions. The U-net network is a decoder–encoder type network, build from convolutional blocks. The extraction of the features helps the dense layers learn the positional information and thus better recreate certain features during the up-scaling. The architecture was tested on a selected dataset and showed the superiority of proposed functions against other, known ones. A similar problem was discussed in terms of the fusion of two different images and then using it in the U-net model [16]. Improvements in U-net architecture are also made on the modeling phase of different layers. It was shown as introducing a pyramid pooling module and attention mechanism to standard architecture in [17]. The proposed architecture showed very high efficiency in the segmentation and detection of the shoreline. However, the authors of the method point out that there are major problems with image analysis in the case of reflected objects. In order to improve the methods, the research should pay attention to removing environmental disturbances. It is worth analyzing a new type of U-net that was dedicated to shoreline detection [18]. Its architecture is composed of MobileNetV2 and a symmetric decoder with focal loss. A detailed discussion and review of the existing literature are presented in [19]. The authors point out the need for research in the field of segmentation, and classification of shoreline detection. The reason is the possibility of analyzing changes taking place on the borders of land and water, or even paying attention to the degradation of the area. The analysis of the existing methods showed the need to improve the methods as well as automate their operation for the purposes of monitoring the coastline.

The presented analysis of the current research in the field of shoreline detection makes it possible to notice that classical solutions as well as neural networks are used and modified. It is worth noting that the methods dedicated to a given problem can be easily transferred to related matters. The shoreline detection algorithms can be used in road line detection or other segmentation problems. Of course, it works both ways. Hence, it is worth paying attention to research on similar topics to see how the same tools can be improved. An example is the use of a different type of network such as the generative adversarial network for road segmentation [20]. Detection of water bodies is also interesting and U-net can be used for it. It was shown in [21], where remote sensing images were analyzed by the new architecture of a neural network called NT-net, noise-canceling transformer network, which improves the extraction of key features in images. An interesting solution is to use tailored metric learning strategies [22], where learning occurs at the point level to form the contour between water and land areas.

3. Proposed model

In this section, we explain line detection based on image processing techniques and its limitations. Then, the U-net model is described with dedicated architecture. We focus on explaining the problem of creating a database for such a neural network. Next, we describe our proposition based on the mentioned tools that can easily change the database after training U-net by just a few iterations. This proposition is based on creating a database by the use of image processing tools, then training and evaluating U-net. In the case of low evaluation metrics, the parameters of the used methods are changed and the whole process is repeated.

3.1. Line detection by image processing techniques

The problem of shoreline detection can be partially solved by classical image processing methods such as edge detection algorithms. The lack of universality of these solutions concerns the necessity to select the appropriate initial parameters for a given image. In this section, a method to obtain a shoreline with a classical approach preceded by preliminary image processing is presented.

To prepare an image for further processing, it is necessary to simplify it. The motivation for it is to reduce the number of information in the image. The first step of this process is the exclusion of excessively bright areas. The reason for this operation is an attempt to prevent the influence of sunlight on the edge detection algorithm course. For this purpose, the shades of white are converted to a new color, for example, blue determined by a triplet of values (r_n, g_n, b_n), pixel by pixel. Naturally, the threshold (r_t, g_t, b_t) against which the pixels are compared must first be defined. This operation can be expressed using the following function $\xi(\cdot)$:

$$\xi((r_a, g_a, b_a)) = \begin{cases} (r_a, g_a, b_a) & \text{if } (r_a, g_a, b_a) \geq (r_t, g_t, b_t) \\ (r_n, g_n, b_n) & \text{if } (r_a, g_a, b_a) < (r_t, g_t, b_t), \end{cases} \quad (1)$$

where (r_a, g_a, b_a) stands for actual color values in the RGB model. Furthermore, the degree of image saturation is modified through the range of colors established for a given case. Firstly the conversion from

(a) (b) (c)

Fig. 1. An example of captured frames and detected shoreline by image processing techniques.

the RGB to the HSV model is needed [23]. Then the range of considered hues has to be defined. It can be represented by two triplets (h_l, s_l, v_l), (h_u, s_u, v_u) which are lower and upper limits respectively. The outcome of this operation is a much less complex image than the original one. Hues outside the range are replaced with black color. Additionally, the Gaussian blur with selected kernel $k \times k$ (should be positive and odd) is applied to further simplify the processed image.

At the end of color modifications, the image is converted to grayscale and again subtly blurred. This conversation can be expressed by the formula:

$$y = 0.299 \cdot r_a + 0.587 \cdot g_a + 0.114 \cdot b_a, \tag{2}$$

where y stands for the amount of light for each pixel. Indeed, grayscale images include only information about the intensity and represent it by shades of gray. Ultimately, an image that is ready for the edge detection algorithm performance is obtained — as the image has been stripped of any edges unnecessary in this process, such as clouds in the sky or waves in the water and the coast is smoothed out by blur. In general, it should be emphasized that all parameters related to colors (i.e. RGB threshold (r_t, g_t, b_t), RGB modifying values (r_n, g_n, b_n) and HSV limits (h_l, s_l, v_l), (h_u, s_u, v_u)) should be set manually, individually for a given scenery.

Then, the Canny [24] and Hough Transform [25] filters can be used. The Canny allows major edge extraction. In turn, the main application of Hough Transform is detecting straight lines which in combination makes it possible to determine the approximate course of the coastline. A computational approach to edge detection created by Canny is based on an observation that the requirements for the application on various vision systems are relatively similar. The structure of the algorithm can be presented by means of several separate stages. Calculations start with computing the first derivative in horizontal direction G_x and the same in vertical direction G_y at each pixel location. Then, for each pixel the gradient magnitude G is calculated with direction Θ_G according to the following equations:

$$G = \sqrt{G_x^2 + G_y^2}, \tag{3}$$

$$\Theta_G = \arctan\left(\frac{G_y}{G_x}\right). \tag{4}$$

Another step is called NMS—Non-Maximal Suppression [26]. This operation consists of searching the local maximum in each pixel's neighborhood in the gradient direction. The purpose of this step is to highlight the key points of too-wide edges. Points that are not local maximums are blanked out, so the result is a binary image. The last component of the algorithm assumes computing low and high thresholds to filter out pixels with low gradient values. Points with a value greater than a high threshold are considered as points that for sure belong to some edge. Low and high thresholds, t_l and t_h respectively, can be obtained according to the following formulas:

$$t_l = \max\{0, \lfloor(1 - \sigma) \cdot v\rfloor\}, \tag{5}$$

$$t_h = \min\{255, \lfloor(1 + \sigma) \cdot v\rfloor\}, \tag{6}$$

where v is a median of pixel values of a given image and σ constitutes an initially fixed parameter. This operation ensures that only the most important edges will be obtained as a result. Also, given that the algorithm is noise sensitive before it starts the 5 × 5 Gaussian blur should be run to reduce noises.

Having a binary image with edges, it can be possible to determine the shoreline location. For this purpose, Hough Transform can be used. Hough space is a two-dimensional plane represented by ρ (a vertical axis) and θ (a horizontal axis). These symbols come from the form of the normal line presented in the parametric equation as:

$$\rho = x \cos\theta + y \sin\theta, \tag{7}$$

where ρ stands for the length of the line and θ is the angle between the normal line and the X-axis. For any point (x, y) of the line, ρ and θ are constant. The algorithm maps the points belonging to the edge of Hough space. Thus in the task of line searching points coordinates (x, y) are known and ρ and θ are sought. Points (ρ, θ) which are defined by (x, y) (points in Cartesian space) map curves (in Hough space). In the case when two points lie on the same line in Cartesian space, the curves in Hough space intersect at the specified point. Because of that, the Hough Transform algorithm determines lines by finding values of (ρ, θ) with the number of intersections greater than a set threshold. After line detection, the last task to do is to extract the exact line of the shoreline. It can be performed for example by selecting the lowest located line or calculating the average line in the area of shoreline, depending on the scenery. An example of such line detection is shown in Fig. 1. The red line indicates the detected shoreline. As can be observed in (c), the image processing detection technique struggles with over-bright samples with visible reflections. The described algorithm with all steps for re-implementation purposes is shown in Alg. 1.

Algorithm 1: Line detection based on image processing techniques.

Input: thresholds (r_t, g_t, b_t), (h_l, s_l, v_l), (h_u, s_u, v_u), new color values (r_n, g_n, b_n), σ parameter, image p, Gaussian blur kernel $k \times k$

Output: image with plotted line p'

```
p' = p;
for every pixel values (r_a, g_a, b_a) on p do
    Eliminate bright shades by ξ((r_a, g_a, b_a)) according to Eg. (1)
      with threshold (r_t, g_t, b_t);
Convert p to HSV model;
Modify p saturation with limits (h_l, s_l, v_l), (h_u, s_u, v_u);
Do Gaussian blur on p with kernel k × k;
for every pixel values (r_a, g_a, b_a) on p do
    Convert (r_a, g_a, b_a) values to gray shade y according to
      Eq. (2) ;
Perform Canny edge detection on p;
Use Hough Transform to find lines on p;
Choose the right line and plot it on p';
return p'
```

3.2. U-net model

U-net represents one of the convolutional neural networks mainly used for processing biomedical images. The concept of this architecture was proposed in 2015 by Ronneberger, Fischer, and Brox [27] in a problem of biomedical image segmentation. The structure of this network is an extension of a fully convolutional network and consists of two components which are called encoder (also known as a contracting path) and decoder (another way expansive path) due to the operations performed by a given part. In contrast to neural networks used for classification where the output is only a determined class, U-net has to carry out the task of classifying each pixel of an image and then project the extracted features back to the image form. This type of problem is called semantic segmentation, the idea of which is to get an image segmented into separate classes.

Each of the paths pulls together a recurring series of smaller operations called blocks. All components resemble the letter "U". For this reason, architecture bears the name U-net. Starting with the encoder, from the original paper the following downsampling operations can be distinguished: two 3×3 unpadded convolutions followed by ReLU and 2×2 max pooling procedure (with downsampling stride equals 2). ReLU is an abbreviation of "rectified linear unit" which in the context of artificial neural networks means an activation function. The formula standing for ReLU is presented in equation:

$$f(x) = \max(0, x), \tag{8}$$

where x stands for an input value. All of these actions make up a block that repeats over the encoder. What is important, each block of the contracting path leads to a double feature channel number.

On the other hand, the decoder's task is to upsample a feature map. A 2×2 up-convolution follows this process, a concatenation with a proper feature map from the encoder and two 3×3 convolutions. Each convolution is succeeded by a rectifier (ReLU). What is more, 'up-convolution' reduces the number of feature channels by dividing it by two. As in the case of an encoder, these actions build every block in the decoder. After a series of blocks is performed, the final 1×1 convolution layer can be launched. It results in mapping obtained classes in an image.

The convolutions included in the encoder are intended to bring out features from the image. Each input image is characterized by three dimensions, which are height, width and depth. Features are detected by a filter that moves across the input image. This tool can be modeled as a two-dimensional array of weights. Assuming F is a feature map that stands for an output, a, b are its indexes of rows and columns, all elements of the resulting matrix can be calculated according to:

$$F[a,b] = (I * \omega)[a,b] = \sum_j \sum_k \omega[j,k] I[a-j, b-k], \tag{9}$$

where ω is a filter of size $m \times m$ and I symbolizes an input image, which size is $N \times N$. Filter size usually is fixed as 3×3, similar to the model description derived from the original paper. Therefore, the filter ω is moving over specified pixels and the process of recalculating this area performs. Unpadded convolution means that there is no application of an additional frame to the image which is built of zeros to prevent information lost during the operation of the filter. Possible losses are caused by the fact that the filter moves more times over the edges of the image than over the rest of the image. The movement of the filter is determined by an input parameter called stride. It can even be equal to 1 and shift pixel by pixel. The size of this step, however, affects what size of the convolution result. Denoting the stride as S, the size F_s of the output matrix can be expressed as:

$$F_s = \left\lfloor \frac{N-m}{S} + 1 \right\rfloor. \tag{10}$$

Another type of layer that a model performs is the max pool layer. Like during the convolution processing, this operation shifts a filter across the image. Thus, two initial parameters can be distinguished again, which are filter size and stride. A characteristic feature that distinguishes this layer from the convolution is the fact that the filter does not have any weights. The purpose of the max pool layer is to reduce a specified number of pixels (determined by the size of the filter) to one value using an aggregation function — *max* function. This means that if the filter is over a given area, it will reduce it to the highest value occurring in it.

On the other hand, up-convolution is a typical layer for a decoder. The task of it is to return (decode) the image to its original size. Up-convolution represents the type of transposed layer. It often gets confused with deconvolution which reverts the process of convolution. Despite this, deconvolution and up-convolution terms cannot be used interchangeably. By a transposed convolution, the same spatial resolution is generated as by a deconvolutional. The difference lies in its internal components. A transposed convolution performs a regular convolution but additionally carries out spatial transformation. However, the transposed convolution is not numerically equal to the original image. To restore the original image size from images after a convolutional operation, every single pixel should generate several values. The solution is to multiply its value by values from the filter and then sum up adequate numbers to receive transposed images in adequate size. For the shoreline detection, a dedicated architecture of U-net is shown in Fig. 2.

To train the neural network, one of the network training algorithms has to be used. The training process relies on minimizing the loss function (also known as the cost function). This function serves to evaluate how well the considered dataset is modeled by a created algorithm. Observation of the loss function value during the training of various model variants allows for extracting the best parameters for it. A high value immediately informs about an unsuccessful learning process. However, the loss function can be defined by various formulas. One of the most popular is sparse categorical cross-entropy defined as below:

$$L = -\sum_{i=1}^{c} y_i \cdot \log \hat{y}_i, \tag{11}$$

where c means the number of outputs (classes in one-hot encoding), y_i is the true label and $\hat{y}$ marks for the predicted one.

Adaptive Moment Estimation (ADAM) [28] is an example of an algorithm for network training. ADAM is based on the stochastic gradient descent which stands for an iterative way of optimizing an objective function along suitable smoothness properties. According to the name of the algorithm, estimation of the first and second moments of the gradient is needed. Moment m_n can be calculated as the expected value of a random variable X to the power of n. The adequate equation is:

$$m_n = E[X^n]. \tag{12}$$

The loss function gradient constitutes a random variable X. Commonly it is evaluated on mini-batches, in other words, small data parts. Assuming $h(\theta)$ is an objective function, the following steps proceed. During every iteration denoted by t, mean m_t (the first moment), variance v_t (the second moment) due to the values of distribution β_1, β_2 are computed. It is shown as:

$$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t, \tag{13}$$

$$v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2 \tag{14}$$

and g_t (a gradient) is calculated in the mini-batch under consideration in on-going iteration t, according to:

$$g_t = \nabla_\theta h_t(\theta). \tag{15}$$

Usually it is assumed that $m_0 = 0$, $v_0 = 0$ and default values of β_1, β_2 equal 0.9 and 0.999, respectively. Additionally, the variance is uncentered. It means that the mean is kept during the calculations and

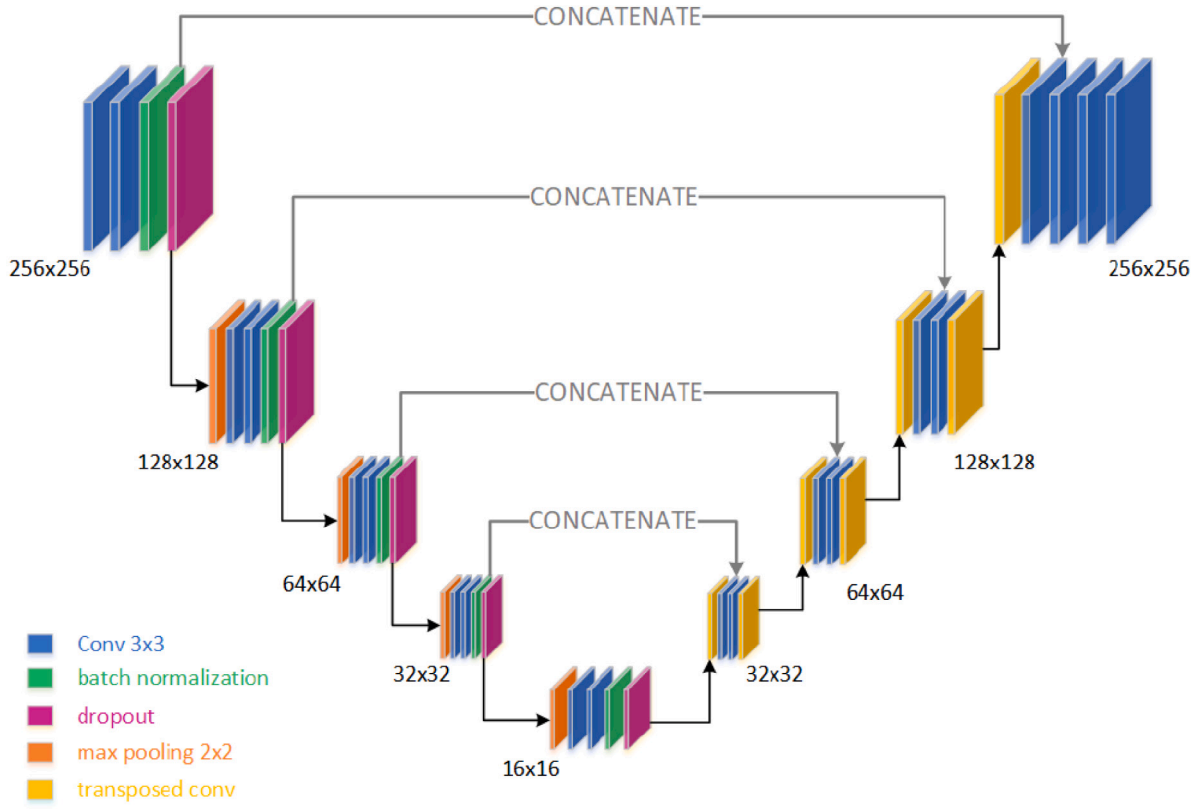

Fig. 2. An U-net architecture for shoreline detection.

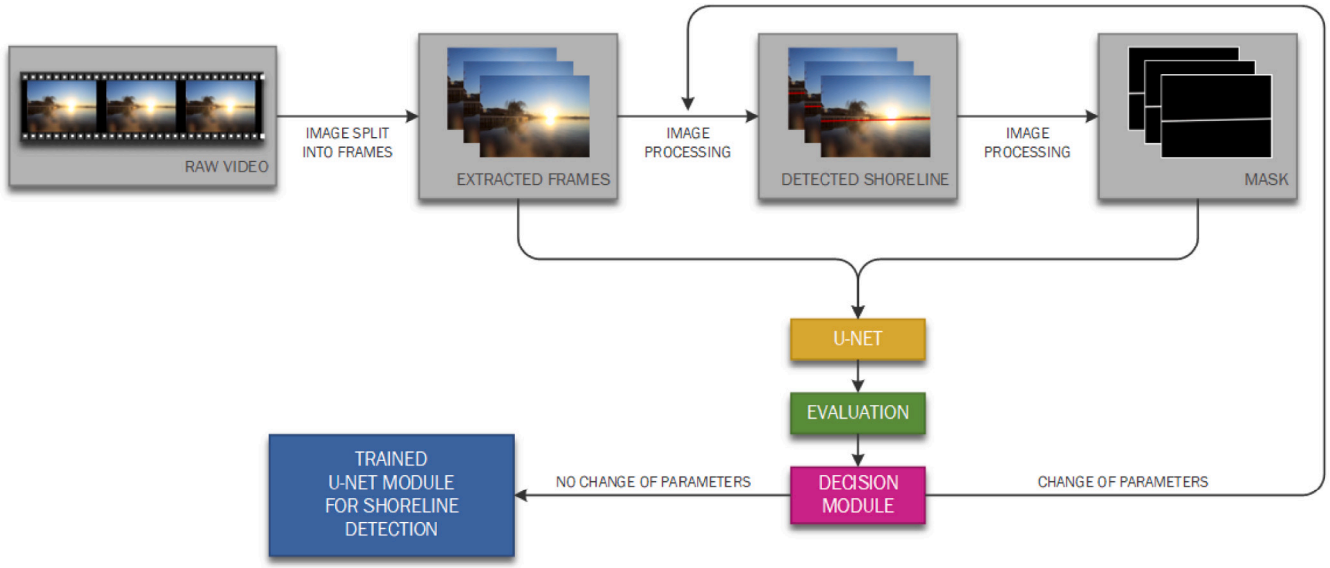

Fig. 3. Visualization of the proposed model. Video is split into a set of frames that are processed by image processing techniques for mask creation. Then the database is trained and evaluated. In the case of low metrics, the parameters are modified and the whole process is repeated.

is not subtracted. Another step is bias-correction of the first and second moments using the following formulas:

$$\hat{m}_t = \frac{m_t}{1-\beta_1^t}, \tag{16}$$

$$\hat{v}_t = \frac{v_t}{1-\beta_2^t}. \tag{17}$$

To scale the learning rate, computed moving averages are needed. The model weights w are updated according to:

$$w_t = w_{t-1} - \eta \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon}, \tag{18}$$

wherein ϵ is a small value that prevents dividing by zero and η constitutes the step size hyperparameter. The algorithm of using U-net is presented in Algorithm 2.

Algorithm 2: U-net training process.

Input: Small value ϵ, β_1 and β_2, training set D, max iteration number T

$m := 0$, $v := 0$, $t := 0$;
while $t < T$ **do**
  Create a mini-batch composed of 32 images from D;
  Calculate gradient using Eq. (15);
  Update first and second moment estimate by Eq. (13)-(14);
  Correct biases according to Eq. (16)-(17);
  Update weights using Eq. (18);
  $t++$;

3.3. Automatic U -net adaptation

U-net classifier uses a dataset composed of two subsets: original images and masks to know what the result should be. For automatization of the mask creation, we propose to use a proposed algorithm in Alg. 1. However, this method has many different parameters and selection can be difficult. The mentioned parameters are threshold values, σ and size of kernel k (see Alg. 1), which can be described as a vector of parameters. For this reason, the initial values of these parameters are randomly generated. A small set of randomly selected images is created (for instance 1% of all images). Then, using generated parameters, masks are created. Then, the average slope of the shoreline on all masks is calculated. This is a parameter that should ideally be 0 degrees to the horizontal line. However, random parameters may not be suitable. A heuristic optimizer can be used to find better parameters. For this purpose, the adaptation function will be defined as the slope angle calculated as follows:

$$R\left((x_1, y_1), (x_2, y_2)\right) = a = \tan\alpha = \frac{y_2 - y_1}{x_2 - x_1}, \tag{19}$$

wherein a is the slope of the straight line passing through the given two points of the shoreline i.e. (x_1, y_1), and (x_2, y_2). These points can be selected based on the points that are furthest from each other, i.e. in the first and last rows of the matrix representing the image.

As a heuristic algorithm, a simple simulated annealing [29] can be used. It is a mathematical model for the minimization of a given function and it is inspired by a metallurgical process. It is based on heating the metal to a high temperature and keeping it under certain conditions. Then, slow cooling of the metal takes place. Formally, this process means maintaining the thermodynamic equilibrium of a given metal. The assumption of the model is the initial high temperature, which is described by the parameter T_{SA}. And it is a parameter that will be changed to reach a lower value. The algorithm uses the simplified thermodynamic equation:

$$P(E) \approx \exp\left(-\frac{\delta C}{k \cdot T_{SA}}\right) \tag{20}$$

where E is analyzed system, k is the Boltzmann constant and δC means difference between two solutions (x_1', x_2') and (x_1, x_2). It can be calculated as:

$$\delta C = \lfloor R(x_1', x_2') - R(x_1, x_2) \rfloor. \tag{21}$$

The new solution is calculated if a random value is lower than the actual $P(E)$. The formula for the new solution is defined as:

$$(x_1', x_2') = (x_1 + \delta x, x_2 + \delta x), \tag{22}$$

where $\delta x = \frac{N(0,1)}{(i+1)\cdot 100}$ in ith iteration of the algorithm, $N(0,1)$ is randomly selected value in $\langle 0, 1\rangle$.

This algorithm is performed to find the average value of slopes (see Eq. (19)) in a created subsets of masks with J images which are closest to the ideal situation, i.e. it tends to 0:

$$\left(\sum_{i \in |J|} R((x_{1,i}, y_{1,i}), (x_{2,i}, y_{2,i}))\right) \to 0. \tag{23}$$

The returned parameters are used for creating a mask for all images in the dataset. The U-net model is trained by it just for a low number of iterations. Then, the final evaluation is performed as:

$$\begin{cases} \sum_{i \in |T_{val}|} R((x_{1,i}, y_{1,i}), (x_{2,i}, y_{2,i})) \in \langle -0.3, 0.3\rangle \text{ and } L < 0.1, \\ \text{actual parameters are good,} \\ \text{in other cases parameters should be modified,} \end{cases} \tag{24}$$

where is a test set with samples used for the evaluation of the trained model.

If the condition is not met, then the method is repeated until the formula is met: a heuristic algorithm modifies the parameters and the U-net is re-trained. In this case, when the parameters are found and the above condition is met, then the U-net is trained by a larger number of iterations to reach better segmentation results. The steps of this proposal are presented in Alg. 3. It should be noted that during analyzing each set of parameters, a small set of images is evaluated — which increases the computational complexity. Hence, it is suggested to create a set consisting of a small number of samples and, after finding the optimal set of parameters, perform full processing.

Algorithm 3: Automatic U-net adaptation to shoreline detection.

Input: dataset with selected images D_q, temperature T_{SA}, Boltzmann's constant k, iterations K

Generate an random values of input parameters $x = (x_1, x_2)$;
$i := 0$;
while $i < K$ **do**
  Calculate δx;
  Use Eq. (22) to prepare a possible new solution $x' = (x_1', x_2')$;
  Process dataset D_q using new solution;
  if $C(x) < C(x')$ *and random value is lower than* $P(E)$ **then**
    $x = x'$;
  else
    if $N(0,1) > \exp\left(-\frac{\delta C}{k \cdot T_{SA}}\right)$ **then**
      $x = x'$;
    Calculate δC according to Eq. (21) using D_q;
  $T_{SA} = T_{SA} \cdot 0.9$;
  $i++$;
Use found a solution to generating masks for all images in dataset;
Train U-net with prepared dataset;
Evaluate U-net;
if *Eq. (24) is met* **then**
  Return trained model;
else
  Repeat algorithm to modify found parameters;

3.4. Application of the proposal solution for unmanned vehicles

The proposed application allows for the automation of data analyses performed in real time (see Fig. 3). It is worth noting that the development of autonomous and unmanned vehicles is an important element in today's industry. An example is unmanned aerial vehicles [30,31], where the possibilities of analyzing camera data for further processing are presented. In the case of unmanned floating vehicles, analysis of camera data allows for a similar operation [32], where the authors indicate the possibilities of using collision-free segmentation using machine learning. In [33], the authors point to the high technological level of such devices, which allows for the prediction of extensive use in future years. As part of the analyses, it was noticed that machine

learning enables quick decision-making based on data obtained in real-time. This contributes to the automation and high precision of such systems.

Our proposal is an important element of the unmanned water vehicles system. The reason for this is the analysis of the water and surface environment through automatic trips and system control using data collected from sensors. However, analyzing the coastline allows for quick decision whether to slow down, change direction, or even return to the starting point. Analysis of camera data can be used for environmental analyses or the above-mentioned vehicle driving inspection, which was also discussed in [32]. The coastline in ideal vehicle flow conditions will be a straight line. If there are large jumps on the shoreline, this will mean some obstructions. If a particular frame shows some spikes or missing data in certain areas of the image, then there will be an object present or the waves will be too large. The ability to analyze such a system is based on frame comparison. If the vehicle has moved away from the previous measurement and the coastline facing the net is still not clear, then this will indicate waves are occurring. In the second situation, when the coastline is more accurate, then there was an object in the previous frame.

4. Experiments

The experimental part was divided into two stages. The first stage consisted of analyzing the proposed solution in laboratory conditions. In this context, actual data from the ongoing project and a public dataset were used to evaluate the framework's operation. The second stage involved conducting tests in real conditions using an unmanned vehicle.

4.1. Settings

The database was created as part of measurement tests in the Zawory town on the Klodno reservoir, Poland. The measurements consisted of recording the video from a camera mounted on an unmanned water vehicle (see Fig. 8). Extraction of individual frames from the video was performed every 3 s. Finally, the dataset was composed of 3969 samples. Then the masks were created by the proposed method and analyzed by us. The evaluation consisted of verifying the obtained segmentation results and their interpretation as correct, i.e. through manual comparison tests. The recorded video was 1280 × 720 pixels and 60 frames per second. The tests were carried out on a computer consisting of an Intel Core i9-10850K 3.60 GHz CPU, 32 GB RAM, and an NVIDIA GeForce RTX 3060 graphics card, and the implementation was made in Python 3.1.

4.2. Analysis of the proposed solution

In the conducted experiments, we used the U-net model shown in Fig. 2. The proposed method was trained using masks created by image processing techniques. As a result, the method parameters were initially randomly generated. The created database was used to train the U-net network with the data set divided into three subsets: training, validation and test in the ratio 70:20:10, which contributed to the following sample numbers: 2778, 794 and 397. The network was trained in 3 iterations with 32 batches. Then, the evaluation was performed and in the case of obtaining low results, the parameters were improved and the dataset was recreated.

Based on the obtained network results, we checked the influence of various parameters of the simulated annealing algorithm. First of all, it was checked whether the values of the adaptation function tended to be 0. Taking into account that the heuristic algorithms are prepared to perform mainly one task and here, if the values were negative, larger values were searched. Otherwise smaller ones. It was dependent on the ideal slope value equal to 0. This operation is possible only if the recording is made by a correctly positioned camera and no additional

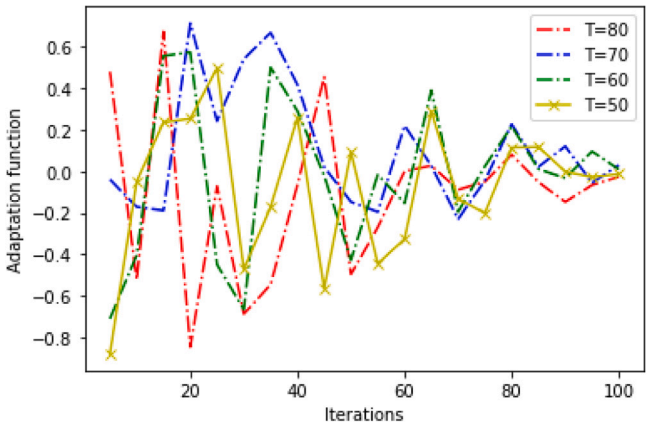

Fig. 4. The dependence of the number of iterations on the value of the adaptation function.

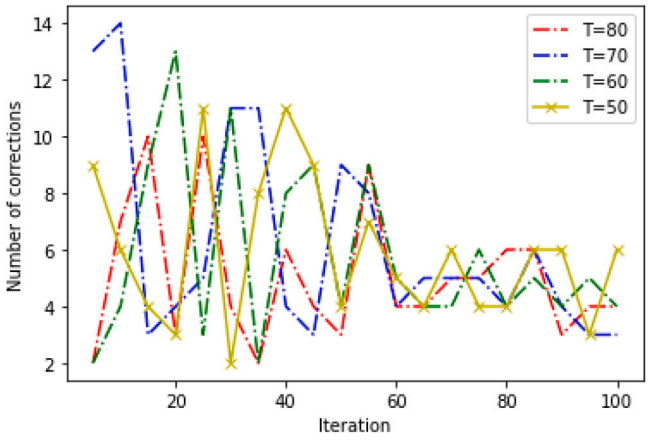

Fig. 5. The dependence of the number of iterations on the number of parameters corrections.

problems. Hence, in the equation Eq. (24) the interval that represents the measurement error due to various problems (like reflection) was included (the dataset used for evaluation was composed of 10 random samples). Tests were performed on four different temperature values like $\{50, 60, 70, 80\}$. The values were selected in a test manner. For values smaller or larger than those, the results were similar without any higher metrics. The obtained results are shown in Figs. 4 and 5. The found parameters are becoming more accurate in terms of their fitting to the shoreline in terms of a large number of iterations. It is easy to see that using 65 iterations returns the results that satisfy the condition in Eq. (24). Moreover, all tested initial temperature values indicated a similar convergence to 0 (see Fig. 4). As the number of iterations increases, the value of the adaptation function tends to 0, which is the assumption of the function — an indication of the slope of the shoreline parallel to the OX axis.

For all cases, an analysis of the number of necessary database modifications was carried out to meet the condition in Eq. (24). The results are shown in Fig. 5. When less than 60 iterations in the simulated annealing method are used, the number of database modifications varies greatly. Above 60 iterations, the required number of parameters needed to create the masks has been between two and six modifications. It is worth noting that with 65 iterations (which were the optimal choice based on Fig. 4), four/five parameter corrections depend on the selected initial temperature. Moreover, based on initial temperature $T = 60$ (for which the curve is most stable in Fig. 5) two tests were performed with an additional 10 training iterations in the end. Of course, the training process was performed after meeting the

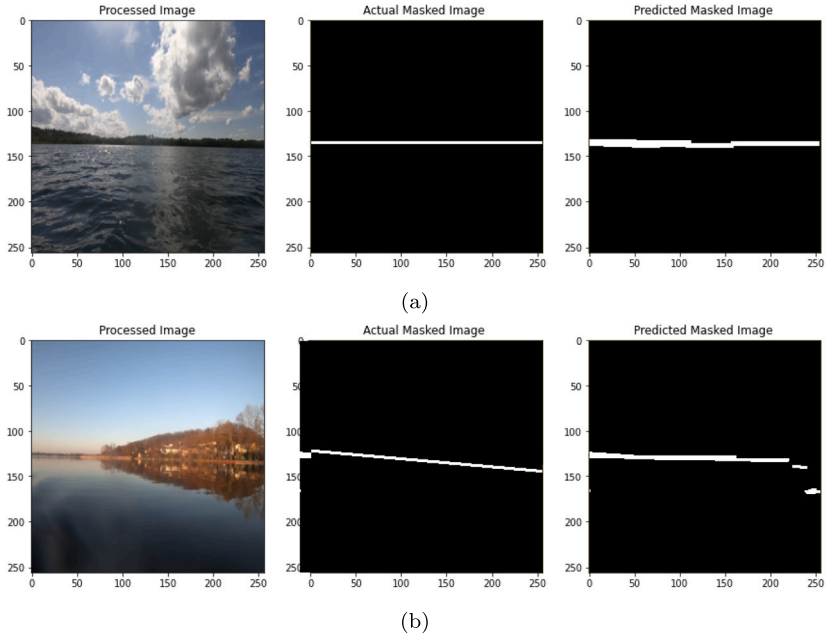

Fig. 6. Examples of images and their processing after 3 corrections of the parameters in the process of creating masks.

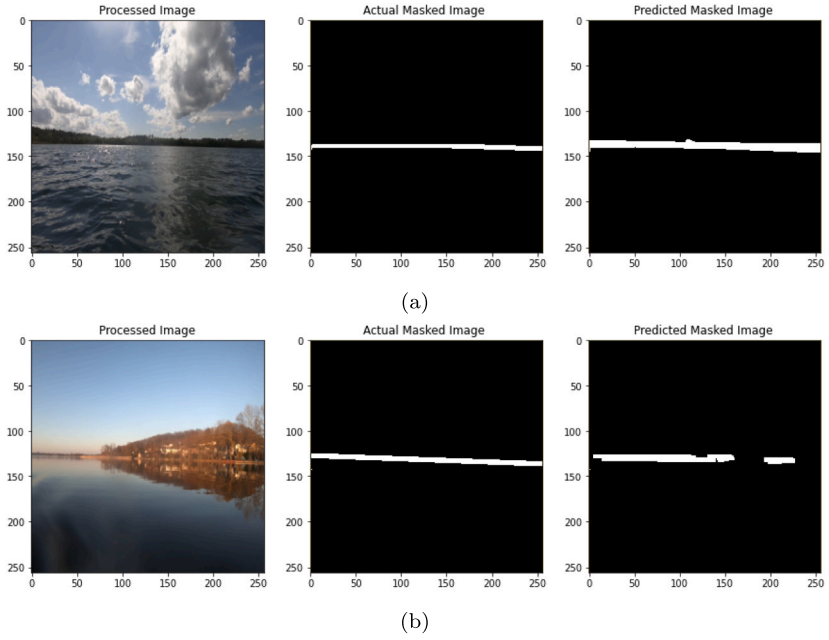

Fig. 7. Examples of images and their processing after 4 corrections of the parameters in the process of creating masks.

requirements specified in Eq. (24). The requirements were met after 3 corrections (in the first test) and after 5 corrections (in the second test). Random samples from validating tests with created masks and predicted shorelines are shown in Figs. 6 and 7. With more parameter corrections, the masks are much more accurate and less susceptible to weather conditions. The shoreline prediction is also more accurate and more like a straight line.

The obtained results indicate that the proposed method can be used in the detection of the shoreline and return good results with optimal values of the heuristic algorithm. The proposal is based on the automatic generation of U-net training masks. In order not to cause too much overhead when finding the best parameters for image processing filters, the network is trained on a small number of iterations. During searching for coefficients, the number of iterations was equal to 3, and then training was performed for 10 iterations on the entire data set. A trained network can quickly process an image for analysis. During the analysis, it was noticed that the loaded network model predicts one image in a time equal to 0.046 s. In the case of 24 video frames per second, causes that they will be processed for 1.104 s. This calculation indicates that it is not possible to analyze video frames in real-time assuming all video frames are processed. However, this is possible with a reduced number of frames to 21.

To analyze the network model, a publicly available dataset named Dasha River Dataset [32]. The data was gathered by a USV along the Dasha River in Shenzhen, China. It consists of 360 full-HD RGB images and binary masks separate the water surfaces from other visible objects. The ground-truth masks were labeled manually. To test our model, we divided the available data into training and test sets in an 80:20 ratio.

Table 1
Result comparison on the Dasha River dataset.

Model	Accuracy	Precision	Recall	F1-score
[32]	97.43	97.37	95.17	96.26
Proposed	99.00	97.86	98.30	98.08

Fig. 8. The unmanned water vehicle during the case study.

The obtained results are displayed in 1 along with the comparison with state-of-the-art. As can be observed, the proposed model scores higher along all the metrics, especially in terms of recall. This indicates, that the presented approach performs better in correctly classifying pixels belonging to the visible water surface.

4.3. Solution analysis in practical application

After the simulation tests were performed, practical tests were also made to verify the operation in real conditions. The camera was installed on the mast of an unmanned water vehicle (see Fig. 8) that recorded the shore while sailing. The data was transmitted to the computer that processed it. The video file was divided into 12 frames per second (12 from 24 frames). After detecting the shoreline, the algorithm allowed to crop the image only to the area above it. Such action allows for the obtaining of data for the easier analysis of environmental changes.

During the analysis of the video image, the already trained U-net network in previous tests was used. The obtained results made it possible to generate a set of images. The study showed that the processing could be done in real-time. However, with an additional delay due to data transfer, as well as an additional processing task — video frame extraction. Fig. 9 presents an analysis of the time needed to process the captured image. The figure describes the average time needed to process/send an image during 60-s video. The charts describe the time needed to process video (like loading and splitting into frames), process frame by U-net, and transmission time.

Obtained results indicate that some time is needed for additional operations not foreseen in the algorithm. This is because the camera is placed on a vehicle that is in the water. Hence, the only option is to transfer data via wifi. Possible delays due to transmission speed were also noted. However, these are elements that do not depend on the operation of the method itself. It has been noticed that the operation of the proposed technique allows the transmitted data to be processed but with a slight delay due to the transmission as well as the computing power.

Test runs using an unmanned vehicle were also performed during drizzle and fog. This action was intended to perform evaluation tests of the proposed solution in worse weather conditions. Worse conditions are interpreted as likely to affect the method's performance by blurring or streaking. An example frame from such a test is shown in Fig. 10. The segmentation network returned a mask that covered the shoreline despite the poorer quality of the video frame. We can see that the line is drawn correctly across the entire frame. However, on the right side, the line is slightly pulled upwards, which should not happen. This situation was also noticed in other frames, although not in all of them. Despite these dozen or so pixels, the shoreline itself is very well extracted, which allows the proposed method to be used in practical applications.

5. Conclusions

New solutions in the use of machine learning and modification of these methods allow for the automation of operations. In this paper, we

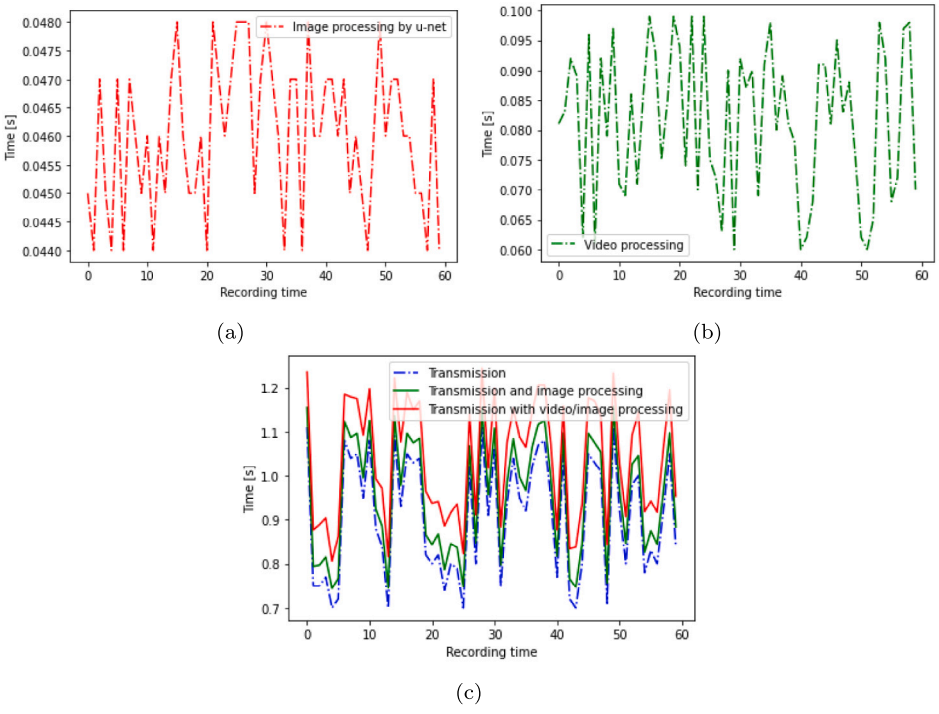

Fig. 9. Time analysis during the case study.

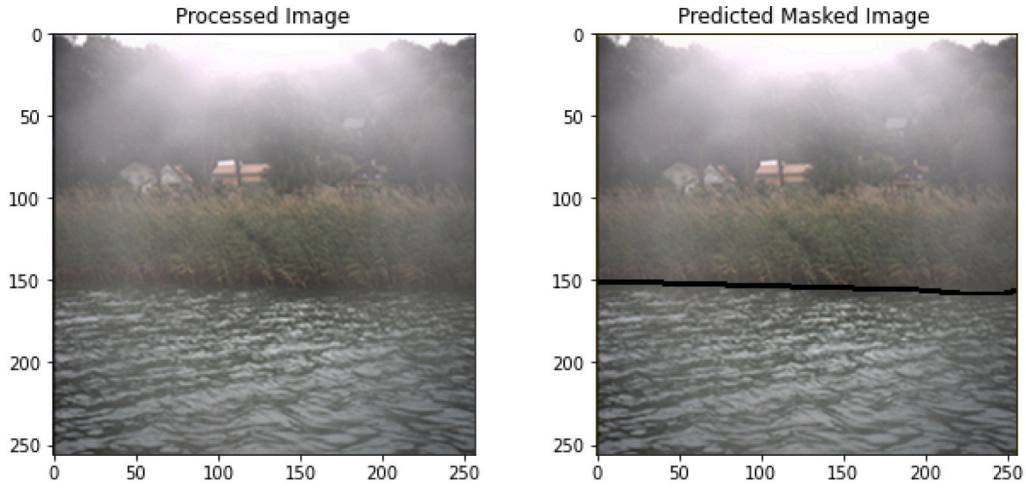

Fig. 10. Sample frame during drizzle and fog. On the left is the original frame and on the right is the same frame with a mask returned by the network model applied.

described that with a small number of training iterations and potential repetition of this operation, it is possible to automatically generate a mask's database for training U-net networks. The idea was to use a heuristic algorithm with a dedicated function to enable the detection of the shoreline. Based on the performed experiments, it was shown that with certain parameters of the algorithms, it is possible to automate the operation with very good results. It is particularly worth noting that subsequent iterations made it possible to detect the shoreline without major errors. In addition, the problems of analyzing such images due to weather conditions have been significantly eliminated. It was visible in the measurements related to the reflection on the water's surface and fog. The proposed solution enables the analysis of data from a camera mounted on an unmanned ship, which allows for quick detection of potential problems with an uneven coastline. This is a problem due to the possibility of an object being between the vehicle and the shore, or even waves, which may contribute to potential flooding. One of the main limitations of the proposal is repeated learning (in the heuristic model) while creating the dataset with masks. However, the operation is performed once to find the best parameter values.

The presented approach demonstrates an efficient scheme for developing a segmentation model for shoreline detection with limited training data. In the research conducted, we found it fitting for the shoreline detection task. However, it could also be applied to other environmental challenges involving borderline detection such as afforestation management, which we plan on exploring in the succeeding research.

In future work, we want to develop the idea with additional parallelism mechanisms that will enable the reduction of the operating time. Moreover, the proposed image extraction over the shoreline can be used in fusion with lidar data. Other research goals involve exploring attention mechanisms designed specifically for the task of shore-line detection, enriching training data with samples from challenging conditions (mainly sun-flares, rain and fog) through augmentation, and using shoreline prediction to detect objects on land using specialized cameras.

CRediT authorship contribution statement

Katarzyna Prokop: Conceptualization, Investigation, Methodology, Software, Writing – original draft. **Dawid Połap:** Conceptualization, Formal analysis, Methodology, Writing – original draft, Writing – review & editing. **Marta Włodarczyk-Sielicka:** Data curation, Formal analysis, Investigation, Supervision, Writing – original draft. **Karolina Połap:** Conceptualization, Methodology, Writing – original draft. **Antoni Jaszcz:** Investigation, Methodology, Software, Writing – review & editing. **Andrzej Stateczny:** Conceptualization, Data curation, Resources, Validation.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was supported by the National Centre for Research and Development (NCBR) of Poland under grant no. LIDER/4/0026/L-12/20/NCBR/2021. This work was also supported by the Silesian University of Technology, Poland, under grant no. 09/020/RGJ24/0030.

References

[1] S. Noto, F. Tauro, A. Petroselli, C. Apollonio, G. Botter, S. Grimaldi, Low-cost stage-camera system for continuous water-level monitoring in ephemeral streams, Hydrol. Sci. J. (2022) (just-accepted).
[2] H. Qiu, Z. Gong, K. Mou, J. Hu, Y. Ke, D. Zhou, Automatic and accurate extraction of sea ice in the turbid waters of the yellow river estuary based on image spectral and spatial information, Remote Sens. 14 (4) (2022) 927.
[3] A. Osipov, E. Pleshakova, S. Gataullin, S. Korchagin, M. Ivanov, A. Finogeev, V. Yadav, Deep learning method for recognition and classification of images from video recorders in difficult weather conditions, Sustainability 14 (4) (2022) 2420.
[4] I. Mehidi, D.E.C. Belkhiat, D. Jabri, A high accuracy segmentation method for retinal blood vessel detection based on hybrid filters and an adaptive thresholding, J. Ambient Intell. Humaniz. Comput. (2022) 1–13.
[5] A. Mishra, P. Gupta, P. Tewari, Global u-net with amalgamation of inception model and improved kernel variation for mri brain image segmentation, Multimedia Tools Appl. (2022) 1–16.
[6] M. Gümüs, S. Durduran, K. Gümüs, Investigation of shoreline change rates using the digital shoreline analysis system in lake beysehir, turkey, Bull. Geophys. Oceanogr. 63 (1) (2021) 119–142.
[7] N. Halalsheh, O. Alshboul, A. Shehadeh, R.E. Al Mamlook, A. Al-Othman, M. Tawalbeh, A. Saeed Almuflih, C. Papelis, Breakthrough curves prediction of selenite adsorption on chemically modified zeolite using boosted decision tree algorithms for water treatment applications, Water 14 (16) (2022) 2519.
[8] T.V. Tran, R. Reef, X. Zhu, A. Gunn, Characterising the distribution of mangroves along the southern coast of vietnam using multi-spectral indices and a deep learning model, Sci. Total Environ. (2024) 171367.
[9] J. Taipalmaa, N. Passalis, H. Zhang, M. Gabbouj, J. Raitoharju, High-resolution water segmentation for autonomous unmanned surface vehicles: A novel dataset and evaluation, in: 2019 IEEE 29th International Workshop on Machine Learning for Signal Processing, MLSP, IEEE, 2019, pp. 1–6.
[10] V. Stepanyants, M. Andzhusheva, A. Romanov, A pipeline for traffic accident dataset development, in: 2023 International Russian Smart Industry Conference, SmartIndustryCon, IEEE, 2023, pp. 621–626.
[11] C.S. Sekar, R.S. Kankara, P. Kalaivanan, Pixel-based classification techniques for automated shoreline extraction on open sandy coast using different optical satellite images, Arab. J. Geosci. 15 (10) (2022) 1–19.

[12] M. Karaman, Comparison of thresholding methods for shoreline extraction from sentinel-2 and landsat-8 imagery: Extreme lake salda, track of mars on earth, J. Environ. Manag. 298 (2021) 113481.
[13] X. Wei, W. Zheng, C. Xi, S. Shang, Shoreline extraction in sar image based on advanced geometric active contour model, Remote Sens. 13 (4) (2021) 642.
[14] J. Li, Study on automatic shoreline extraction based on multi-spectral remote sensing images, in: 2021 5th International Conference on Advances in Image Processing, ICAIP, 2021, pp. 68–75.
[15] C. Seale, T. Redfern, P. Chatfield, C. Luo, K. Dempsey, Coastline detection in satellite imagery: A deep learning approach on new benchmark data, Remote Sens. Environ. 278 (2022) 113044.
[16] M. Aghdami-Nia, R. Shah-Hosseini, A. Rostami, S. Homayouni, Automatic coastline extraction through enhanced sea-land segmentation by modifying standard u-net, Int. J. Appl. Earth Obs. Geoinf. 109 (2022) 102785.
[17] Y. Yin, Y. Guo, L. Deng, B. Chai, Improved pspnet-based water shoreline detection in complex inland river scenarios, Complex Intell. Syst. (2022) 1–13.
[18] L. Yao, D. Kanoulas, Z. Ji, Y. Liu, Shorelinenet: an efficient deep learning approach for shoreline semantic segmentation for unmanned surface vehicles, in: 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS, IEEE, 2021, pp. 5403–5409.
[19] E. McAllister, A. Payo, A. Novellino, T. Dolphin, E. Medina-Lopez, Multispectral satellite imagery and machine learning for the extraction of shoreline indicators, Coast. Eng. (2022) 104102.
[20] A. Abdollahi, B. Pradhan, G. Sharma, K.N.A. Maulud, A. Alamri, Improving road semantic segmentation using generative adversarial network, IEEE Access 9 (2021) 64381–64392.
[21] H.-F. Zhong, Q. Sun, H.-M. Sun, R.-S. Jia, Nt-net: A semantic segmentation network for extracting lake water bodies from optical remote sensing images based on transformer, IEEE Trans. Geosci. Remote Sens. 60 (2022) 1–13.
[22] A. Van Soesbergen, Z. Chu, M. Shi, M. Mulligan, Dam reservoir extraction from remote sensing imagery using tailored metric learning strategies, IEEE Trans. Geosci. Remote Sens. 60 (2022) 1–14.
[23] A.R. Smith, Color gamut transform pairs, ACM Siggraph Comput. Graph. 12 (3) (1978) 12–19.
[24] J. Canny, A computational approach to edge detection, IEEE Trans. Pattern Anal. Mach. Intell. (6) (1986) 679–698.
[25] J. Illingworth, J. Kittler, A survey of the hough transform, Comput. Vis. Graph. Image Process. 44 (1) (1988) 87–116.
[26] W. He, K. Yuan, An improved canny edge detector and its realization on fpga, in: 2008 7th World Congress on Intelligent Control and Automation, Ieee, 2008, pp. 6561–6564.
[27] O. Ronneberger, P. Fischer, T. Brox, U-net: Convolutional networks for biomedical image segmentation, in: International Conference on Medical Image Computing and Computer-Assisted Intervention, Springer, 2015, pp. 234–241.
[28] D.P. Kingma, J. Ba, Adam: A method for stochastic optimization, in: 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings.
[29] C. Venkateswaran, M. Ramachandran, K. Ramu, V. Prasanth, G. Mathivanan, Application of simulated annealing in various field, Mater. Charact. 1 (1) (2022) 01–08.
[30] J.B. Awotunde, M.O. Arowolo, A.L. Imoize, Y. Farhaoui, A.E. Adeniyi, A machine learning-based model for energy efficiency classification of an unmanned aerial vehicle, in: The International Conference on Artificial Intelligence and Smart Environment, Springer, 2022, pp. 54–63.
[31] K. Teixeira, G. Miguel, H.S. Silva, F. Madeiro, A survey on applications of unmanned aerial vehicles using machine learning, IEEE Access (2023).
[32] R. Zhou, Y. Gao, P. Wu, X. Zhao, W. Dou, C. Sun, Y. Zhong, Y. Wang, Collision-free waterway segmentation for inland unmanned surface vehicles, IEEE Trans. Instrum. Meas. 71 (2022) 1–16.
[33] Y. Qiao, J. Yin, W. Wang, F. Duarte, J. Yang, C. Ratti, Survey of deep learning for autonomous surface vehicles in marine environments, IEEE Trans. Intell. Transp. Syst. 24 (4) (2023) 3678–3701.