Robust A*-Search Image Segmentation Algorithm for Mine-like Objects Segmentation in SONAR Images

Original Scientific Paper

Ivan Aleksi
Josip Juraj Strossmayer University of Osijek,
Faculty of Electrical Engineering, Computer Science and Information Technology
Kneza Trpimira 2B, 31000 Osijek, Croatia
ivan.aleksi@ferit.hr

Tomislav Matić
Josip Juraj Strossmayer University of Osijek,
Faculty of Electrical Engineering, Computer Science and Information Technology
Kneza Trpimira 2B, 31000 Osijek, Croatia
tomislav.matic1@ferit.hr

Benjamin Lehmann
ATLAS Elektronik GmbH
Sebaldsbrücker Heerstraße 235, 28309 Bremen, Germany
benjamin.lehmann@atlas-elektronik.com

Dieter Kraus
Hochschule Bremen, University of Applied Sciences,
Institute of Water-Acoustics, Sonar-Engineering and Signal-Theory
Neustadtswall 30, D-28199 Bremen, Germany
dieter.kraus@hs-bremen.de

Abstract *– This paper addresses a sonar image segmentation method employing a Robust A*-Search Image Segmentation (RASIS) algorithm. RASIS is applied on Mine-Like Objects (MLO) in sonar images, where an object is defined by highlight and shadow regions, i.e. regions of high and low pixel intensities in a side-scan sonar image. RASIS uses a modified A*-Search method, which is usually used in mobile robotics for finding the shortest path where the environment map is predefined, and the start/goal locations are known. RASIS algorithm represents the image segmentation problem as a path-finding problem. Main modification concerning the original A*-Search is in the cost function that takes pixel intensities and contour curvature in order to navigate the 2D segmentation contour. The proposed method is implemented in Matlab and tested on real MLO images. MLO image dataset consist of 70 MLO images with manta mine present, and 70 MLO images with cylinder mine present. Segmentation success rate is obtained by comparing the ground truth data given by the human technician who is detecting MLOs. Measured overall success rate (highlight and shadow regions) is 91% for manta mines and 81% for cylinder mines.*

Keywords *– A*-search, image segmentation, path planning, synthetic aperture sonar*

1. INTRODUCTION

Image segmentation plays an important role in image processing, and with the development of computers it became usable in many practical applications: sonar, radar and medical imaging. However, it enables the selection of specific objects in an input image such that a human can more easily detect isolated areas of interest. Usually, image segmentation is used in autonomous systems, where the human interaction is replaced by the machine vision systems.

Image segmentation method considered in this work deals with a practical application of automated detection of underwater Mine-Like Objects (MLO). After the Second World War, underwater mines possess a real threat to the human population by endangering missions that include ships, sub-marines and other sea vessels. There are several types of anti-ship and -submarine mines (moored, limpet, contact, manta, cylinder, etc.) that are usually recognized by acoustic vision systems. MLO detection missions are usually done with

a Side-scan SONAR system, being attached with a tow cable to a ship that is pulling the system [1]. After inspection of a certain area, a huge SONAR image is recorded. From this huge image, the so-called regions of interest (ROI) are extracted. Effective ROI selection is done in [2]. Each ROI is then used as an input image to the MLO segmentation process. MLO is defined by highlight (object) and shadow regions, i.e. regions of high and low pixel intensities in a side-scan sonar image. Overall segmentation represents the union of the object and shadow region segmentation. Subsequently, after MLO map is reconstructed the mines are usually destroyed with remotely operated vehicles (ROVs) or with autonomous underwater vehicles (AUVs).

The rest of this paper is organized as follows. Section 2 gives the literature overview of the paper topic. The proposed method for sonar image segmentation is explained in Section 3. Experimental results are presented in Section 4. Finally, Section 5 concludes the paper.

2. RELATED WORK

Image segmentation methods can be grouped in two categories: contour-based and region-based segmentation methods. Main task of contour-based segmentation method is finding a border that separates two disparate regions. There are a lot of region-based methods with similar idea [3], [4], [5], [6], [7]. Contour is defined as a closed line, which is possible to write in parametric form, by using a spline function or a polygon. Contour shape is usually changed by the impact of out-side forces until a minimum energy constraint is satisfied. Forces are usually defined by curve contour on some other criteria. Minimum energy constraint is usually used in image segmentation methods [8]. Well known problem of contour-based methods is a definition of initial contour. Initial contour shape must be close to image segmentation result. Otherwise, contour methods often jam into a local minimum.

On the other hand, region-based segmentation methods directly classify dissimilar regions. One simple segmentation method is image thresholding, where certain gray level intensities define thresholds [9]. Usually, threshold methods usually provide poor results. Other simple grouping method is k-means [10], which is limited in application since it doesn't use spatial distance between image pixels. However, these methods are often used for generating initial conditions of certain image segmentation method. Waterfall image segmentation [11] represents pixel intensities as a height map and uses contours to connect points with the same height. High intensity represents a mountain and low intensity represents a valley. Valleys are filled with water until valley and mountain regions are not separated satisfactory enough. Parametric approach in image segmentation is done with EM-methods (Expect Maximization) [12], [13]. Firstly, it is required to know certain image characteristics and different density functions that correspond to certain image regions. Image properties depend on sonar system which is used to create an image. In sonar systems, image intensities are generally distributed by Gama or Rayleigh probability distribution. EM method places image pixels into a constant number of groups by iteratively changing initial parameter values and their corresponding weights. Similarity with Mean-shift method is described in [14]. Mean-shift method is a non-parametric classifying method [15], [16]. It is based on an idea that the mod of probability density function is in the middle of the density function's gradient. Markov Random Field (MRF) method is used in image seg-mentation with the idea that the intensity correlation is larger between pixels that are close together than the pixels that are distanced apart [17]. There are a lot of different approaches that solve segmentation problem with relatively little computation power, as it is the case with the use of graph search methods [18].

MLO segmentation and detection in side-scan sonar system is mostly done with the combination of before mentioned methods and machine learning approaches. In [19] authors use intrackability measurement and improved Bag of Words (BOW) algorithm in combination with support vector machine (SVM) to detect MLOs. BOW features are used for SVM training. SVM with histogram intersection kernel was used for classification. Authors state recognition rate of maximum 91.33 %. In [20] authors tested Convolutional Neural Network (CNN) for MLO segmentation. CNN consist of a convolutional layer with six filters, a pooling layer, configured to max-pool and a fully-connected layer with two outputs, zero that represents background and one that represents MLO. Approach was tested on limited size of the training and evaluation dataset. Authors report accuracy of 86 % in the worst case for a limited dataset. K-means in combination with Chan-Vese active contours and morphological operations are used for sonar image segmentation in [21]. Two measures are used to describe the MLOs, shadows and highlights distance d and central masses linear line horizontal angle Θ. Segmented image is converted to d,Θ coordinate space and several model parameters were defined for two possible hypotheses of random variable. Model parameters are determined based on a training set with Neyman-Pearson test and the model was confirmed with false alarm probability of 0.92×10^{-3} in the worst case.

In this work we present a new method for sonar image segmentation of MLOs based on A*-Search algorithm [22]. Method is not based on machine learning and therefore no training is necessary. The proposed RASIS method represents the image segmentation problem as a path-finding problem. Main modification concerning the original A*-Search is in the cost function that takes pixel intensities and contour curvature in order to navigate the 2D segmentation contour. Pixel intensity thresholding technique is used to build a goal distance map with the numerical navigation function (NNF). The NNF uses the L2 norm to estimate the Euclidean distance between the current pixel and the goal pixel. Estimated

distance represents predicted cost h to the goal pixel. Overcomed distance represents cost g, penalty cost i is calculated according to pixel intensities, while with cost ρ straight paths are preferred. Finally, the proposed method uses the cost function f, as the sum of g, h, i and ρ, in order to determine a contour around an MLO.

3. RASIS: ROBUST A*-SEARCHIMAGE SEGMENTATION

RASIS method is based on a graph search image segmentation with a hybrid region and contour-based image segmentation approaches. Utilized graph search is based on a modified version of the A*-Search method. Usually, the A*-Search is used in conjunction with *numerical navigation function* (NNF) for navigation in mobile robotics [22]. However, the application of A*-Search to image segmentation in general, as well as to MLO segmentation in SONAR images, is a novel practical application. RASIS uses Signal Change Detection (SCD) for image preprocessing and initial segmentation of MLO's object and shadow regions [23]. SCD is also used for initializing start/goal positions for the A^*-Search contour planning. SCD is a statistical method which can be used for detection of amplitude jumps in 1D signal. Output from SCD method is a list of indices that can be used for the approximation of a discrete stochastic signal with mean amplitude values. More detailed mathematical explanation of the SCD method is given in Appendix.

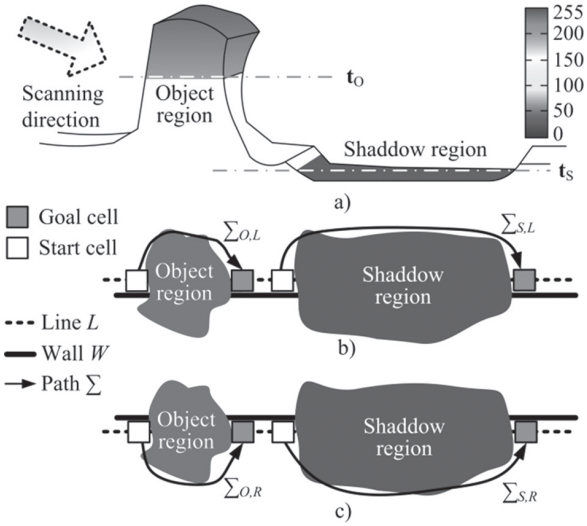


Fig. 1. Mine-like object (MLO) pixel intensities and threshold pixel intensity levels (a); inserted artificial walls that divide object and shadow regions into two parts for path planning on the left side (b); and the right side (c).

Let *obstacle region* be a set of pixels with intensity value set to 255, and let *free region* be a set of pixels with pixel intensity ranging from 0 to 244. *Obstacle region* is a term used for isolating specific region, either object or shadow, while the *free region* is its surrounding background. Path planner has a task to avoid *obstacle regions* and to plan a path in the *free region*.

Basic idea of RASIS method is to plan a path around an *obstacle region*, c.f. Fig. 1. In order to enable path planning around a certain isolated region, it is required to divide that region in two parts and apply a path planning method from the left and from the right side separately. Line L divides the object and the shadow region in two parts, i.e. *obstacle regions*, c.f. Fig. 1. Artificial wall W is placed on the left (right) side of L when path $\sum$ is planned on the right (left) side. Segmented regions are obtained by connecting left and right planned paths into contours around the object $\sum_O = \sum_{O,R} + \sum_{O,L}$ and the shadow region $\sum_S = \sum_{S,R} + \sum_{S,L}$.

3.1. IMAGE PREPROCESSING

Input ROI image I_M is an 8-bit image with typically 100 x 100 pixels [2]. Image I_M is oriented according to the sonar point-of-view so that the object region appears before the shadow region, c.f. Fig. 2a. In image preprocessing step, c.f. Fig. 3, image denoising is done and image I'_M is created. Image I'_M is padded with k=3 pixels and is filtered with median filter in order to reduce image noise and preserve lines. Intensity values of padded pixels are set to the background mean μ with standard deviation σ. In this work μ=50 and σ=5, c.f. Fig. 4a. Image thresholding technique is used for creation of thresholded I'_O and I'_S images, c.f. Fig. 4c and d. For object image I'_O, *obstacle region* corresponds to pixels with intensities above the threshold t_O, while the remaining pixels are assigned to *free* region. Similarly, for shadow image I'_S, *obstacle region* corresponds to pixels with intensities below the threshold t_S. SCD method is used for improving the image threshold results and assigning *obstacle/free* regions, c.f. Fig. 3. SCD segmentation is applied on pixel intensities along a column or a row in order to detect the start and the end of *obstacle/free* regions. Vertical SCD (SCD_V) is applied along image columns in order to create object I''_O and shadow I''_S images, c.f. Fig. 4e and f. Horizontal SCD (SCD_H) is applied along image rows and segmented object image I'''_O is created, c.f. Fig. 4g. Results from previous steps are combined and images I^*_O and I^*_S are created with eq. (1) and (2), c.f. Fig. 3.

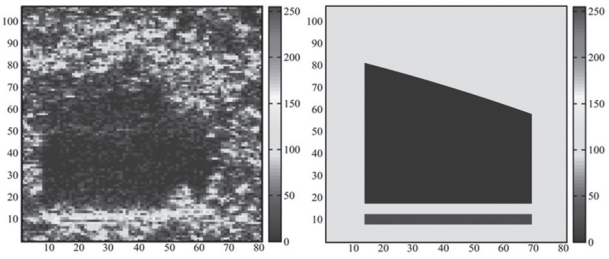


Fig. 2. Region of Interest (ROI) 8-bit grayscale Sidescan SONAR image with Mine-like Object (a); segmented object region (red color), shadow region (boue color) and the background (green color) (b).

$$I_O^*(i,j) = \begin{cases} I_M(i,j), & I'_O(i,j) \neq 255 \wedge I'''_O(i,j) \neq 255 \\ 255 \quad , & I'_O(i,j) = 255 \vee I'''_O(i,j) = 255 \end{cases} \quad (1)$$

$$I_S^*(i,j) = \begin{cases} I_M(i,j), & I'_S(i,j) \neq 255 \wedge I''_S(i,j) \neq 255 \\ 255 \quad , & I'_S(i,j) = 255 \vee I''_S(i,j) = 255 \end{cases} \quad (2)$$

Morphological operations and image erosion methods [24] are applied on I^*_O and I^*_S images with Alg. 1. Finally, segmented object I_O and shadow I_S region images have relatively homogenous *obstacle regions*, c.f. Fig. 4k and l.

Black and white image I_B is created from the image I'''_O by applying eq. (3), c.f. Fig. 4h. Image I_B is used for path planner initialization step.

$$I_B(i,j) = \begin{cases} 0, & I'''_O(i,j) \neq 255 \\ 1, & I'''_O(i,j) = 255 \end{cases} \quad (3)$$

After the image preprocessing step, obstacle regions are defined with pixel intensity value 255 in object I_O and shadow I_S images. Remaining pixel intensities represent the *free region*.

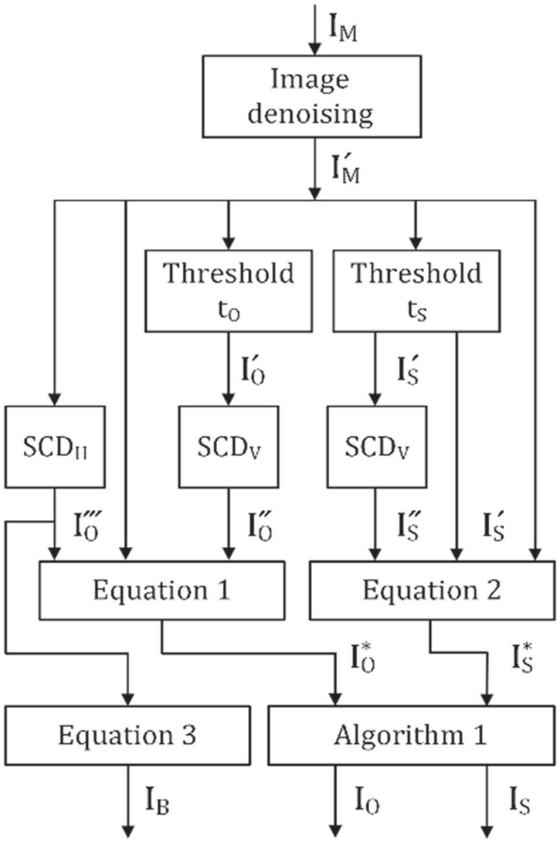


Fig. 3. Image preprocessing steps for RASSIS method.

3.2. PATH PLANNING INITIALIZATION

RASIS method considered in this work uses a path-planning algorithm. Start and a goal positions are required for a planner to plan a path. They are assigned according to the *obstacle regions* in the black and white image I_B (3). *Labeling technique* [24] is applied on image I_B in order to determine the number of *obstacle regions* n_R. In this way individual *isolated object regions* are created. Example with n_R=2 is illustrated in Fig 4h. Each *isolated object region* has assigned a centroid location. Vertical lines L are placed through centroid locations, c.f. Fig. 5. Along a line L, one pair of *start* and *goal* locations is placed in the object region image I_O and in the shadow region image I_S. For each pair of start and goal locations, a path planner plans a path on the left and on the right side of a line L.

When planning a path on the left (right) side of a line L, an artificial wall W is inserted right (left) to the line L. Wall lines W are inserted in I_O and the I_S, separately for each side left and right, respectively, c.f. Fig. 6a to d. Wall line W is parallel to the line L and its pixels are considered to be in *obstacle region*. Line L is placed at centroid coordinate x_c, while walls W is placed at x_c-1 and x_c+1 coordinates, respectively.

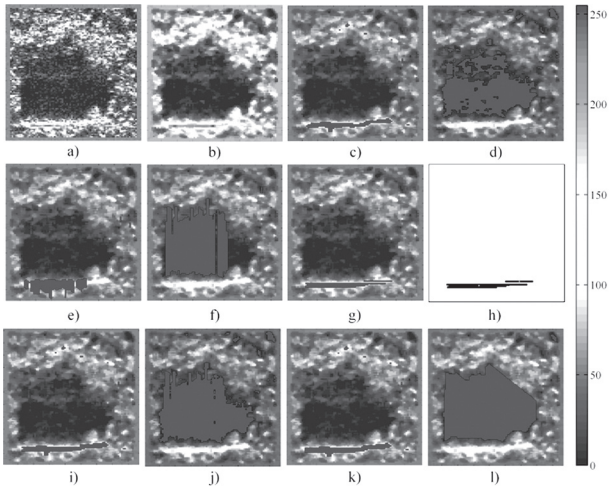


Fig. 4. Intermediate images for RASSIS method: I_M (a); I'_M (b); I'_O (c); I'_S (d); I''_O (e); I''_S (f); I'''_O (g); I_B (h); I^*_O (i); I^*_S (j); I_O (k); I_S (l).

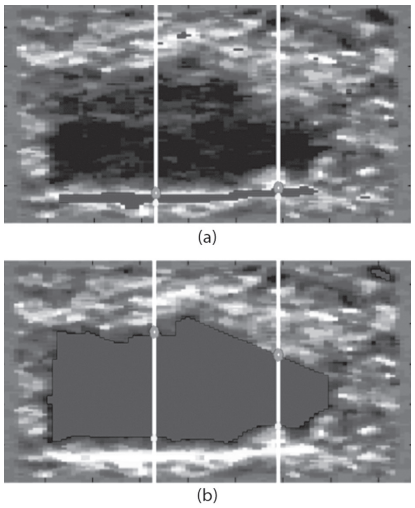


Fig. 5. Placing vertical lines L (white line) with start (green circle) and goal (red circle) positions in: object I_O (a); and shadow I_S image (b).

Algorithm 1. Homogenization of isolated regions in I_O^* and I_S^* images.

Inputs: $I_O^*, I_S^*, I_M, N_P, N_i$
Outputs: I_O, I_S
1: Set $I_O = I_O^*$ i $I_S = I_S^*$
2: ***for each*** k, from $k = 1$ to $k = N_P$, step 1
3: ***for each*** pixel $\mathbf{p}$ in image I_O
4: $n_i = 0$
5: ***for each*** point $\mathbf{p}'$ of 8 neighbor points of $\mathbf{p}$
6: ***if*** $I_O(\mathbf{p}') == 255$
7: $n_i = n_i + 1$
8: ***if*** $n_i \geq N_i$
9: $I_O(p) = 255$
10: ***for each*** point $\mathbf{p}$ in image I_S
11: $n_i = 0$
12: ***for each*** point $\mathbf{p}'$ of 8 neighbor points of $\mathbf{p}$
13: ***if*** $I_S(\mathbf{p}') == 255$
14: $n_i = n_i + 1$
15: ***if*** $n_i \geq N_i$
16: $I_S(\mathbf{p}) = 255$
17: Set $I_{BW,S}(I_S == 255) = 1$ and $I_{BW,S}(I_S \neq 255) = 0$
18: Set in $I_{BWE,S}$ the image erosion result of $I_{BW,S}$ with 3 × 3 unit matrix
19: Set $I_S(I_{BWE,S} == 0) = I_M$ and $I_S(I_{BWE,S} == 1) = 255$

Numerical navigation function (NNF) is a potential field-based navigation function (terrain information is also used in the literature). Firstly, it was defined for wheeled mobile robot trajectory planning strategies [25]. Those methods interact with a robot as with a ball in a configuration space affected by an imaginary potential field. This potential field pushes a robot away from *obstacle regions*, avoiding collision with obstacles in its environment. On the other hand, potential field pulls a robot into a goal configuration. Analytic construction of navigation function over a *free region* (*non-obstacle region*) of arbitrary geometry is in general a complex task. However, if *free region* is represented with grid of cells, numerical navigation function (NNF) can be very efficient with integer programming implementation [25]. In this work, NNF is used in a conjunction with artificial wall placement. It divides image in two parts and enables path planning on each part separately. Assigned NNF values are illustrated in Fig. 6e to h.

NNF value h_{ij} is assigned to the pixels that are in the *free regions,* while the pixels that are in *obstacle regions* do not have the NNF value assigned. NNF value h_{ij} is calculated based on eight neighbor pixels NNF values. In order to calculate NNF value $h_{ij+u,j+v}$ for neighbor pixels $p_{i+u,j+v}$, where $u, v \in \{-1, 0, 1\}$, following equations are used.

$$h_{i+u,j+v}(k+1) = min\left(h_{i+u,j+v}(k), h_{ij}(k) + \Delta h_{ij}(k)\right) \quad (4)$$

$$\Delta h_{ij}(k) = \begin{cases} 0, & u = v = 0 \\ 2, & |u| \neq |v| \\ 3, & |u| = |v| = 1 \end{cases}, \quad u, v \in \{-1, 0, 1\} \quad (5)$$

In addition to (4) and (5), one more constraint is used in this work. Diagonal neighbors $h_{i+u,\ j+v}$ (|u|=|v|=1) are not considered if both of their vertical and horizontal neighbors (|u|≠|v|) are in obstacle regions. In this way, the NNF wave propagates around, instead of over, the diagonally oriented thin obstacles. This is the case when an obstacle is a single pixel-wide thin line oriented in ±45°. Therefore, modified NNF function does not navigate a path planner over thin obstacles.

While this implementation is a fast integer programming method, its resulting propagation wave looks more octagonal than circular as it is supposed to be, c.f. Fig. 6e to h. However, this approximation gives satisfying results. NNF calculation starts in the goal position and ends when all pixels have their NNF values assigned. However, if NNF calculation would end in a start pixel, not all pixels would have their NNF values assigned. This approach may be used for mobile robot's route planners that have a task to minimize the route length. In this work, we calculated NNF values for all pixels in order to optimize the segmented path.

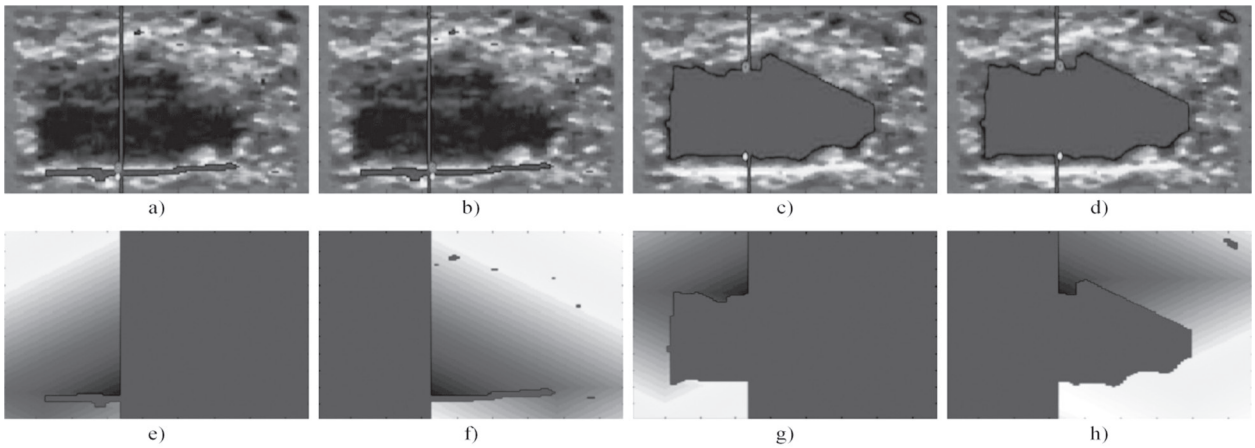


Fig. 6. Numerical Navigation Function (NNF) calculated for object region image I_O right side (a) and the left side (b). NNF calculated for shadow region image I_S right side (c) and the left side (d). Goal location has NNF value 0 (colored with dark blue color). Obstacle region has pixel intensity value 255 (colored with dark red color), while the free region has pixel intensities ranging from 0 to 254.

The NNF distance metric is calculated separately for the object and the shadow region, c.f. Fig. 6e to h. Starting in a goal pixel $\mathbf{p}_{goal}$, NNF values are assigned according to the distance between goal pixel $\mathbf{p}_{goal}$ and a certain pixel $\mathbf{p}_{ij}$ connected with $\mathbf{p}_{goal}$. Initially, NNF value assigned to $\mathbf{p}_{goal}$ is set to 0, as it is illustrated with dark blue color in Fig 6c to h. Let δ be pixel's $\mathbf{p}_{i,j}$ corresponding width and height in meters. In the first iteration, four diagonal neighbor cells are $\delta\sqrt{2}$ distance apart, that can be approximated as 1.5 δ, while distance from other four neighbor cells is δ. After normalization with 2/ δ, the integer NNF gradient for neighbor cells is 2 or 3.

3.4. PATH PLANNING WITH MODIFIED A*-SEARCH ALGORITHM

Let graph $G(V, E)$ be an undirected graph, which is created dynamically during a planning procedure, where V is a vertex set and E is an edge set. Vertex $v \in V$ is connected with vertex $v' \in V$ if there is an edge $e \in E$ between v and v'. Vertices v and $v'$are defined with eq. (6),

$$v = (\mathbf{p}_{i,j}, \varphi), \quad v' = (\mathbf{p}', \varphi') \tag{6}$$

where $\mathbf{p}_{i,j}$ stands for the pixel at location (i, j) and φ is the angle calculated from the current pixel $\mathbf{p}_{i,j}$ to the neighbor pixel $\mathbf{p}'$.

Let T be a tree structure with vertex v_{start} as its root. Root vertex is inserted in T with the ROOT (v, T) function. Tree T consists of visited vertices v'. Each visited vertex v' is connected in T with its corresponding parent vertex v. This connection in T is done with a function CONNECT (v', v, T) and is also denoted with $\overline{\mathbf{pp}'}$.

Let Q be the list of vertices sorted according to their cost function value f (v), which is calculated as follows

$$f(v) = g(v) + h(v) + i(v) + \rho(v) \tag{7}$$

where $g(v)$ is accumulated Euclidean distance calculated between connected vertex pairs from vertex v_{init} to a current vertex v in a solution tree T. The NNF value $h(v)$ is estimated Euclidean distance from a current location $\mathbf{p}$ to a goal location $\mathbf{p}_{goal}$. Pixel intensity penalty $i(v)$ guides a path planner to plan a segmented path close to the *obstacle region* (object/shadow) by taking neighbor pixel intensities into account. In this way, a path is planned at preferable intensity levels. Curvature of planned path is controlled with $p(v)$ function which applies penalty to curved paths.

The following functions are used to manipulate with the list Q and tree T. Function INSERT (v, Q) inserts vertex v into the list Q that is sorted according to the vertex v cost function value (7). Initial vertex is inserted with the zero cost. Function EMPTY (Q) returns *true* if the list Q is empty, otherwise *false*. Function SORT (Q) sorts the list by ascending cost function $f(v)$ value. Function FIRST (Q) returns a first vertex v from the sorted list Q and removes the same vertex from Q. Function GOAL (v) returns *true* if a goal vertex is reached, otherwise *false*. Goal is reached when a vertex v_{goal} has position equal to the goal pixel $\mathbf{p}_{goal}$ which is input to (Alg.1). Function DEPTH (T, v) returns the current depth d_T for a vertex v in tree T, which is the number of vertices connected to v. On the other hand, function NEXT (T, v, d_T) returns a vertex v from T at the depth d_T. Function VISITED (v) increments and returns a counter of how many times a vertex v is visited. It is used in order to avoid repeated search at the same vertex v. This function is a modification when compared to a real information of how many times a vertex v is visited. Instead of having a 3D binary array of visited vertices with 2D locations and 1D orientations (6), a 2D matrix of locations is used to store *visited counters* for each location $\mathbf{p}_{i,j}$ and not concerning the orientation from which a certain vertex is visited.

Algorithm 2. Modified A^*-Search algorithm.

Inputs: $G, h, v_{start}, v_{goal}, N_{visits\_max}, D_{limit}, D_{step}$
Output: T as a path from v_{goal} to v_{start}

```
Create empty list Q and tree T
INSERT(v_start, Q)
ROOT(v_start, T)
Set d_T = D_limit
while not EMPTY(Q) do
    SORT(Q)
    Set v ←FIRST(Q)
    if GOAL(v) then return T as the solution.
    if DEPTH(T, v) > d_T then
        Remove all vertices from the list Q
        Set d_T =DEPTH(T, v) − (D_limit − D_step)
        Set v ←NEXT(T, v, d_T)
        INSERT(v, Q)
        Mark all vertices as not visited
    if VISITED(v) < N_visited_max then
        set X ←EXPAND(G, h, v)
        for each vertex v' in X
            CONNECT(v', v, T)
            INSERT(v', Q)
return solution not found.
```

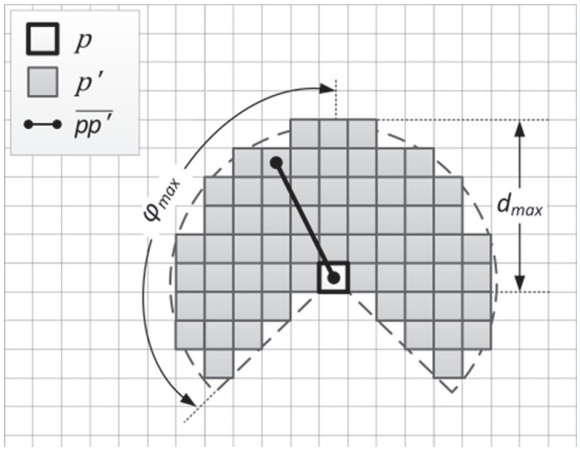


Fig. 7. Set x of visited vertices $v' \in V$ (corresponds to pixel p') that are connected with their parent vertex $v' \in V$ (corresponds to pixel p).

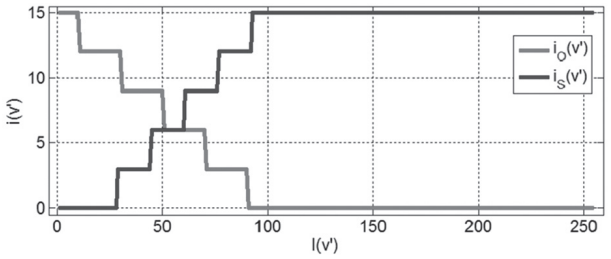


Fig. 8. The impact of image intensity $I(v')$ to the cost function part $i_O(v')$ for planning a path around the object and the $i_S(v')$ for shadow region.

Modified depth-limited A-Search* method (Alg. 2) is derived from [27] and has the following inputs: graf G, NNF matrix h, start/goal vertices v_{start} and v_{goal}, $N_{visit\ max}$ the number of times a vertex can be visited, graph search depth limit D_{limit} and depth change D_{step}. Typical depth limit values used in this work are D_{limit}=4 and D_{step}=1. Output is a tree T with a set of edges that connects vertices v_{goal} and v_{start}. A*-Search algorithm begins with the empty list Q and the empty tree T, with initially inserted starting vertex v_{start} in both of them (Alg.2 lines 1-3). While Q is not empty (Alg.2 lines 5-19), Q is sorted and the vertex v with smallest cost function value is taken from Q (Alg.2 lines 6-7). If a v_{goal} is reached, tree T is returned as the solution (Alg.2 line 8).

Value d_T controls the depth limit (Alg.2 line 4). When the graph search depth exceeds the current depth limit, a vertex v with lowest cost function value (7) is placed in an empty list Q (Alg.2 lines 9-14). In this way, a graph search planning is done until the $d_T + D_{limit}$ depth is reached, and the planning is restarted at the $d_T + D_{step}$ depth.

If a vertex v is not visited maximum number of times $N_{visited\ max}$, vertex v is marked as visited by a function VISITED (Alg.2 line 15). The EXPAND function returns a set x of vertices v' that are connected with v (Alg.2 line 16). Vertices v' that are connected with the current vertex v are inserted into T and Q (Alg.2 lines 17-19). Finally, a tree T consists of expanded vertices in G, i.e. it contains edges and vertices that connects a goal vertex v_{goal} with the root vertex v_{start}. Finally, a path Σ is reconstructed by following edges in T, backwards, from a goal vertex v_{goal} to a start vertex v_{start}.

Edges of a graph G define a set X of neighbor vertices $v' \in V$ that are connected with their parent vertex $v \in V$. Connections between v and v' are determined by the so called *vertex expansion*. Neighbor pixels **p'**, i.e. visited vertices v', are defined with the following two *geometrical constraints* and one *pixel intensity constraint*, c.f. (8-10). *Geometrical constraints* are the maximum distance change d_{max} (8) and the maximum angle of direction change φ_{max} (9), c.f. Fig. 7. *Pixel intensity constraint* (10) ensures that each vertex v' in x has a corresponding location **p'** in the *free region*.

$$d_{max} \leq \|\mathbf{p}, \mathbf{p}'\| \tag{8}$$

$$\varphi_{max} \leq |\angle(\mathbf{p}, \mathbf{p}')| \tag{9}$$

$$I(\mathbf{p}') \leq 255 \tag{10}$$

Additionally, a feasible constraint is also taken into account when considering which vertex v' remains in the set X. Vertex v' is removed from the set X if there are more than $N_{feasibile}$ pixels in the obstacle region which are geometrically positioned on a line $\overline{\mathbf{PP}'}$. In this paper, for planning a path in the image I_O, where *object region* is *obstacle region*, parameter $N^O_{feasible} = 0$. For the image I_S, where the *shadow region* is the *obstacle region*, parameter $N^S_{feasible} = 3$. The *feasible constraint* is used in order to avoid path planning over thin obstacle regions in an image I_O. Since *obstacle region* in shadow image I_S is less homogenous, it is allowed to plan a path over thin lines which are in the *obstacle region*.

Finally, *feasible constraint* together with eq. (5-7) define the set X of neighbor vertices v' from their parent vertex v.

3.5. COST FUNCTION

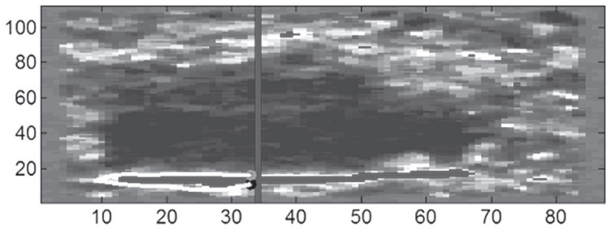
Cost function $f(v')$ is defined in eq. (7). It consists of four parts. First two parts involve path distance. Cumulative distance sum $g(v') = g(v) + |v, v'|$ is the overcomed distance between v_{start} and v'. Estimated distance $h(v')$ between v' and v_{goal} is calculated with the NNF. Third part $i(v')$ involve pixel intensity and is calculated differently for object I_O and shadow I_S image, with eq. (11) and eq. (12), respectively, c.f. Fig. 8.

Weight factor w_I determines how much $i(v')$ impacts the cost function $f(v')$. Intensity is quantized into n_I groups by transforming the interval [0, t_O] to [0, n_I].Values used in this work are w_I=15, n_I=5 , t_O=100 and $\bar{t}_S = 1.6\ t_s = 25$. Therefore, the object intensity penalty $i_O(v')$ value increases with increasing pixel intensity value $I_O(v')$. In this way, path planning in the I_O image is preferred to be over pixels with high intensity. On the other hand, shadow intensity penalty $i_S(v')$ increases with smaller pixel intensity value $I_S(v')$. In this way, path planning in I_S image is preferred to be over pixels with low intensity.

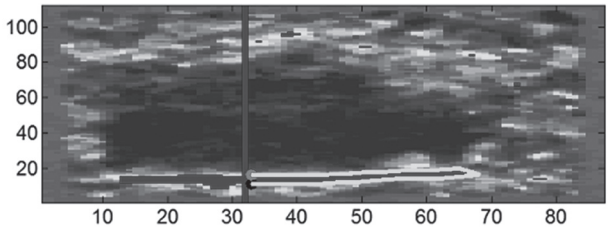
$$i_O(v') = \begin{cases} \dfrac{w_I \left\lfloor n_I \cdot \frac{t_O - I_O(v')}{t_O} \right\rfloor}{n_I}, & I_O(v') < t_O \\ 0, & I_O(v') \geq t_O \end{cases} \tag{11}$$

$$i_S(v') = \begin{cases} 0, & I_S(v') < \bar{t}_S \\ \dfrac{w_I \left(n_I - \left\lfloor n_I \frac{t_O - \bar{t}_S - I_S(v')}{t_O - \bar{t}_S} \right\rfloor \right)}{n_I}, & \bar{t}_S < I_S(v') < t_O \\ w_I, & I_S(v') \geq t_O \end{cases} \tag{12}$$

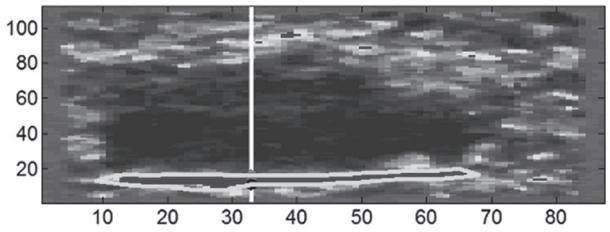
Fourth part of the cost function is a path curvature $\rho(v')$. It enables straight path planning, as it is defined with eq. (10).



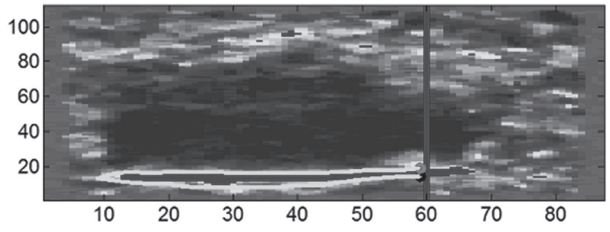
(a)



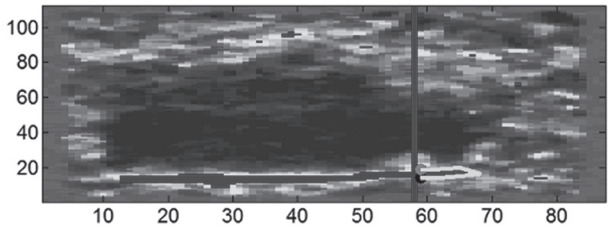
(b)



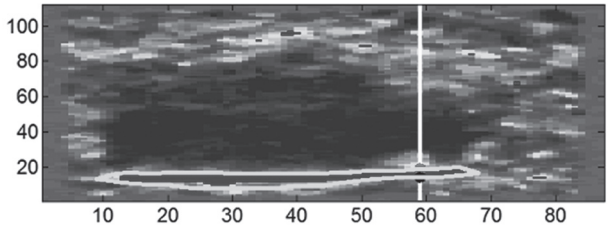
(c)



(d)

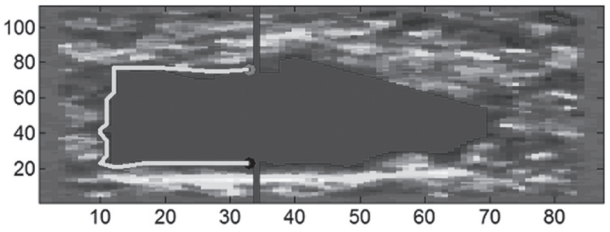


(e)

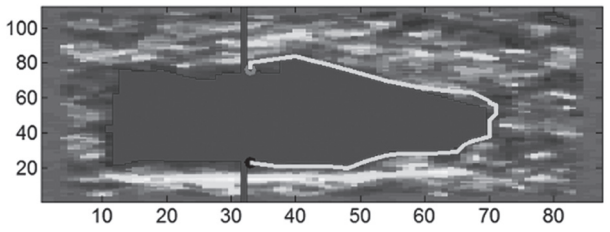


(f)

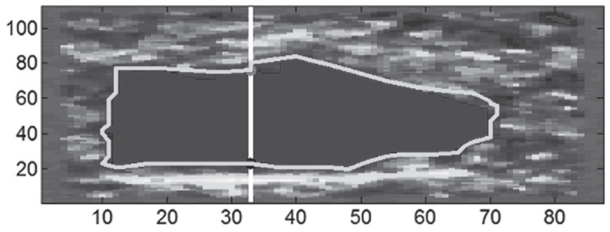
Fig. 9. Planned paths around the object region for: line L_1 from left side $\sum_{O,L,1}$ (a), right side $\sum_{O,R,1}$ (b) and connected paths $\sum_{O,1}$ (c); line L_1 from left side $\sum_{O,L,2}$ (d), right side $\sum_{O,R,2}$ (e), connected paths $\sum_{O,2}$ (f).



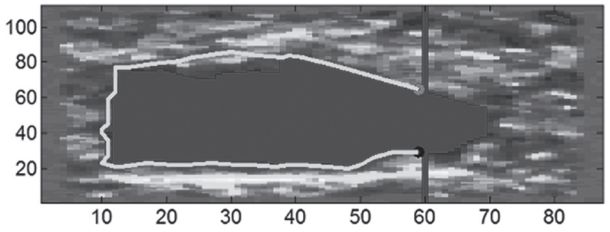
(a)



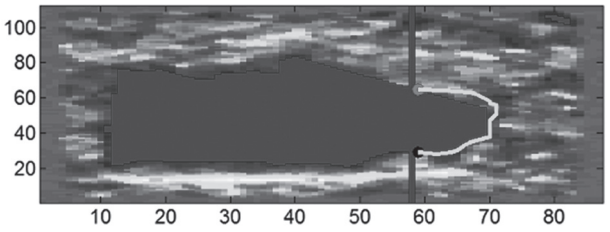
(b)



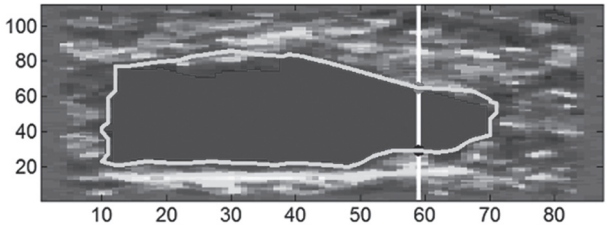
(c)



(d)



(e)



(f)

Fig. 10. Planned paths around the shadow region for: line L_1 from left side $\sum_{S,L,1}$ (a), right side $\sum_{S,R,1}$ (b) and connected paths $\sum_{S,1}$ (c); line L_2 from left side $\sum_{S,L,2}$ (d), right side $\sum_{S,R,2}$ (e), connected paths $\sum_{S,2}$ (f).

$$\rho(v') = \left\lfloor w_\rho \frac{\Delta\varphi'' - \Delta\varphi'}{2\,\Delta\varphi_{max}} \right\rfloor \quad (13)$$

Weight factor w_ρ determines how much a path curvature $\rho(v')$ impacts total cost function $f(v')$ value. In this work, w_ρ=5. Angular change $\Delta\varphi' = \varphi - \varphi'$ is calculated between visited vertex v' and its corresponding parent vertex v, while $\Delta\varphi'' = \varphi'' - \varphi$ is calculated between the current vertex v and its parent vertex v'' in tree T.

3.6. IMAGE SEGMENTATION BASED ON PATH PLANNING WITH MODIFIED A*-SEARCH ALGORITHM

Proposed modified A^*-Search algorithm is used for MLO image segmentation four times, for each line L_z, $z = 1,...,Z$, where Z is the number of lines L in an MLO image. Two lines, L_1 and L_2, are illustrated in Fig. 5. Modified A^*-Search is used two times for image I_O and two times for image I_S, for each line L_z. In image I_O, path is planned on the left $\sum_{O,L,z}$ and on the right $\sum_{O,R,z}$ side of the line L_z. In image I_S, path is planned on the left $\sum_{S,L,z}$ and on the right $\sum_{S,R,z}$ side of the line L_z. Segmentation result is obtained by connecting left and right paths $\sum_{O,z} = \sum_{O,R,z} \oplus \sum_{O,L,z}$, i.e. $\sum_{S,z} = \sum_{S,R,z} \oplus \sum_{S,L,z}$, c.f. Fig. 9 and Fig. 10. Contours $\sum_{O,z}$ and $\sum_{S,z}$ are converted to binary images $R_{O,z}$ and $R_{S,z}$ by applying eq. (14).

$$R_{O,z} = \begin{cases} 1, & if\ p_{i,j} \in I_O\ is\ inside\ \Sigma_{O,z} \\ 0, & otherwise \end{cases}$$
$$R_{S,z} = \begin{cases} 1, & if\ p_{i,j} \in I_S\ is\ inside\ \Sigma_{S,z} \\ 0, & otherwise \end{cases} \quad (14)$$

In case when there are more than one line L_z, i.e. Z>1, union of binary images is used, i.e. $R_O = R_{O,1} \vee R_{O,2} \vee ... R_{O,Z}$ and $R_S = R_{S,1} \vee R_{S,2} \vee ... R_{S,Z}$. In order to represent the final segmentation result, binary images R_O and R_S are joined into a single 8-bit color image R by using eq. (15).

$$R = 122 \cdot \left(I + R_O + R_S \wedge (R_O \wedge R_S)\right) \quad (15)$$

In eq. (15), I is identity matrix and factor $R_S \wedge (R_O \wedge R_S)$ is used to represent the *object region* pixels in the front of the *shadow region* pixels, in the case when they are overlapping. Resulting 8-bit color image R is illustrated in Fig. 11, where pixel intensity value 244 is assigned to the *object region* (red color) 0 to the *shadow region* (blue color) and 120 to remaining pixels, i.e. background (green color).

4. EXPERIMENTAL RESULTS

Proposed RASIS algorithm is tested on real MLO images. MLO images are obtained from a large Side-scan sonar image by a pre-segmentation method proposed in [2]. The tested dataset consists of 140 Side-scan sonar images divided in two equal sets. First set represents 70 MLO images of manta mine, and the second set 70 MLO images of cylinder mine. For speed up parallel RASIS method is implemented in MATLAB. Parallel algorithm is depicted in Fig. 12. It consists of several sequential and parallel steps. After the image preprocessing step, the number of lines L is calculated and each line L is considered for path planning. Artificial wall W is placed on the left and on the right side of each line L. In this way object ($I_{O,L}$ and $I_{O,R}$) and shadow ($I_{S,L}$ and $I_{S,R}$) images are created. Start and goal locations are calculated for path planning. The NNF matrix calculation and the A^*-Search path planning are executed four times for each line L.

Planned paths on the left and on the right side are connected into a single contour around the object and shadow region, respectively. Finally, segmentation result is calculated as a union of surfaces inside contours.

Experimental analysis of the proposed method includes execution time measurement and MLO segmentation performance measurement for highlight and shadow regions.

Execution times are measured on an Intel i5 CPU @ 3.33 GHz, 8 GB RAM and Windows 10 operating system. In order to additionally shorten the execution time, proposed SCD method [Appendix] is implemented as *MATLAB's C++/Mex function* [28], with the speedup value 100. Mean execution time values for each part of the RASIS method, c.f. Fig. 12, are illustrated in Table I. RASIS method segmentation results are illustrated for manta mine, c.f. Fig. 13, and for cylinder mine, c.f. Fig. 14. RASIS segmentation performance is depicted in Table II. Segmentation success rate is obtained by comparing the ground truth data given by the human technician who is detecting MLOs, c.f. Table II.

Regarding execution time, the maximum value is below 15 s for manta mines, and below 30 s for cylinder mines. Shorter execution time is expected with a pure C++ implementation and parallel implementation on hybrid platforms. Regarding segmentation performance, measured worst case success rate is 91% for manta mines and 81% for cylinder mines. Manta mine segmentation performance is better than cylinder mine since cylinder mine has smaller surface and has a similar round shape to the surrounding seabed. Therefore, more processing time is spent on the path planning of the round objects. When compared to other MLO segmentation methods [19], [20], [21] the proposed RASIS method gives satisfactory segmentation results with no need for the training stage.

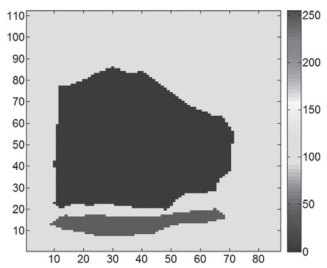


Fig. 11. Segmentation result with RASIS method.

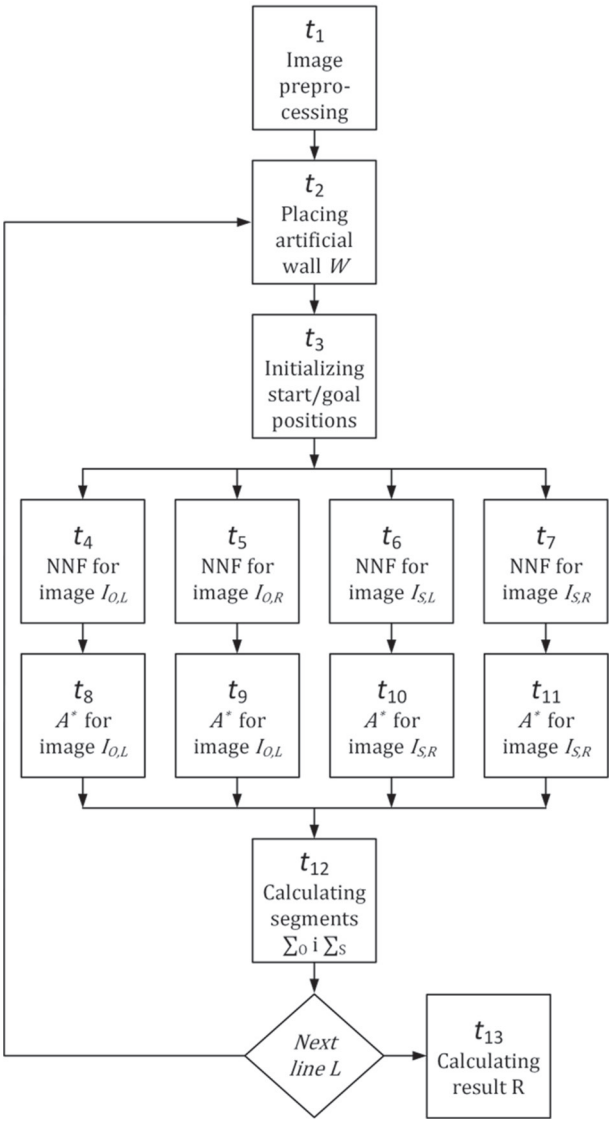


Fig. 12. Parallel RASIS method implementation.

Table 1. RASSIS method average execution times.

Execution times, c.f. Fig. 11.	Manta mine [ms]	Cylinder mine [ms]
$\overline{t}_1$	0,1201	0,1294
$\overline{t}_2$	0,0003	0,0002
$\overline{t}_3$	0,0009	0,0008
$\overline{t}_4$	3,5445	3,1996
$\overline{t}_5$	3,2880	3,0796
$\overline{t}_6$	3,3896	2,5056
$\overline{t}_7$	3,2050	2,6088
$\overline{t}_{NNF}$	3,9581	3,6605
$\overline{t}_8$	0,6455	1,7841
$\overline{t}_9$	0,7228	1,7024
$\overline{t}_{10}$	3,3924	4,8671
$\overline{t}_{11}$	2,8187	4,3889
$\overline{t}_{A^*}$	4,1692	6,3646
$\overline{t}_{12}$	0,0007	0,0005
$\overline{t}_{13}$	1,2021	3,4004
$\overline{t}$	9,4514	13,6571

Table 2. RASIS method's performance.

Under water mine type	Manta	Cylinder
Execution time ($\overline{t}\pm\sigma_t$)	9.45 ± 2.27 s	13.7 ± 4.51 s
Minimum execution time	5.01 s	5.42 s
Maximum execution time	14.7 s	27.7 s
Object region segmentation success	100%	81%
Shadow region segmentation success	91%	84%
Overall segmentation success	91%	81%

5. CONCLUSION

In this paper a novel Robust A*-Search Image Segmentation (RASIS) method is proposed. It uses Signal Change Detection for region-based image pre-segmentation and a modified A*-Search path planning method for contour-based segmentation. Modification of A*-Search method is in the cost function that takes pixel intensities and contour curvature in order to navigate the 2D segmentation contour. RASIS method represents a hybrid combination of region and contour-based image segmentation methods. It is not machine learning based method and therefore no training is necessary.

Method is implemented in MATLAB and the SCD method implemented in MATLAB's C++/Mex function. Test dataset consist of real Side-scan sonar MLO images. In the experimental analysis execution time was measured and MLO segmentation performance for highlight and shadow regions was evaluated. Regarding execution time, the maximum value is below 15 s for manta mines, and below 30 s for cylinder mines. Shorter execution time is expected with a pure C++ implementation and parallel implementation on hybrid platforms. Regarding segmentation performance, measured worst case success rate is 91% for manta mines and 81% for cylinder mines.

Several avenues for future work remain open. Firstly, it would be interesting to see how it behaves on larger dataset with different types of MLOs. Also, future work will include parallel implementation of the RASIS method on hybrid platforms multicore CPUs and GPGPU units to speed up the method execution time. Furthermore, classification of MLO type should be considered based on segmented highlight and shadow regions.

6. APPENDIX. SIGNAL CHANGE DETECTION

Signal Change Detection (SCD) is a statistical method which can be used for the detection of Ξ changes in a 1D signal x, i.e. $x(n)$, n=1,...,N. SCD method's output is a list of indices ($n_1, n_2, ..., n_\xi$), which enables the approximation of the original signal with only Ξ +1 distinct

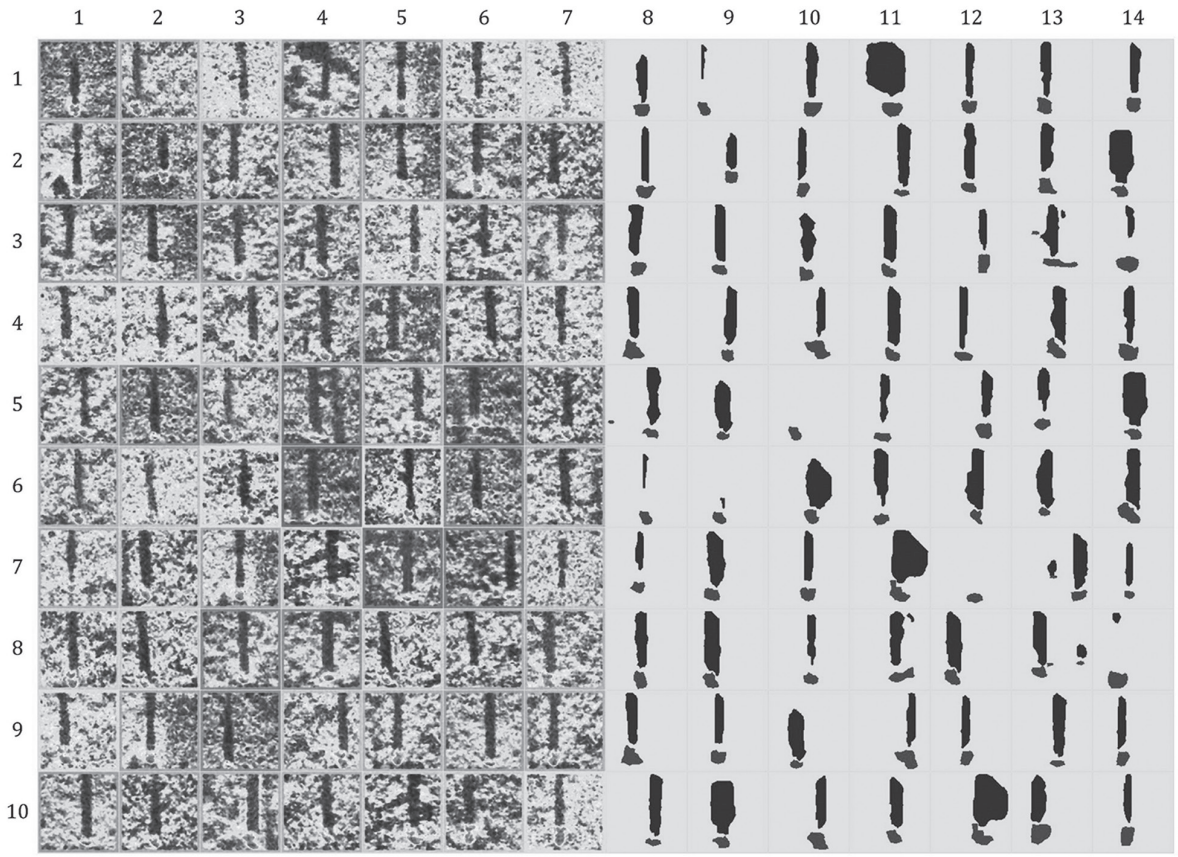


Fig. 13. Sonar images of underwater *manta mines* (left) and corresponding images with segmented object (red), shadow(blue) and background (green) regions (right).

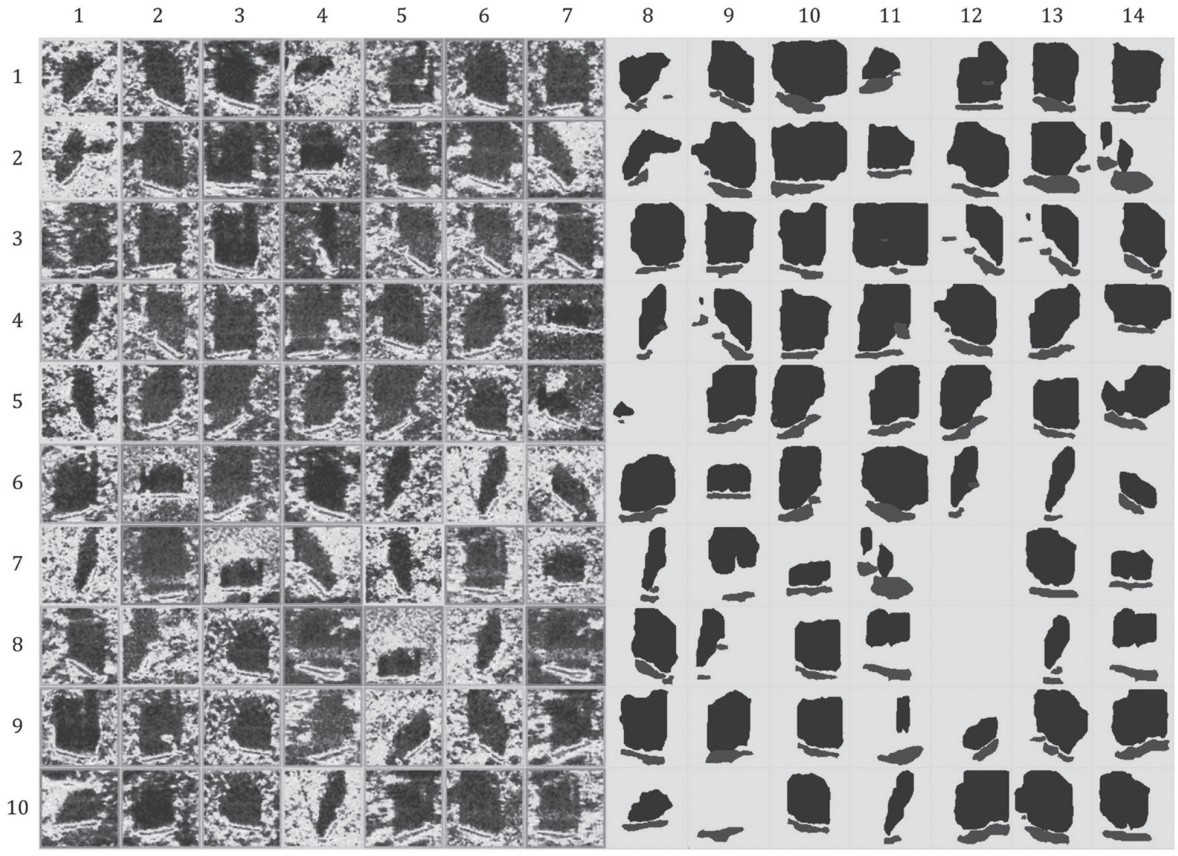


Fig. 14. Sonar images of underwater cylinder mines (left) and corresponding images with segmented object (red), shadow(blue) and background (green) regions (right).

amplitudes ($A_1, A_2, \ldots, A_{\xi+1}$). This approach is applicable for known/unknown amplitude changes $\Delta A = A_{(\xi+1)} - A_{\xi}$, which may occur at known/unknown indices n_{ξ}, $\xi = 1, \ldots, \Xi$ [29]. Specifically, in this work the SCD method has $\Xi = 2$ changes.

Let x be a *referent signal* (known original) with two amplitude changes ΔA at indices n_1 and n_2. The original signal x is defined with (A.1) and it has three amplitude levels A_1, A_2 and A_3 which are defined by the mean values of x, before, between and after indices n_1 and n_2 (A.2).

$$x(n) = \begin{cases} A_1, n = 1, \ldots, n_1 \\ A_2, n = n_1 + 1, \ldots, n_2 \\ A_3, n = n_2 + 1, \ldots, N \end{cases} \quad \text{(A.1)}$$

$$\begin{aligned} A_1 &= \frac{1}{n_1} \sum_{n=1}^{n_1} x(n) \\ A_2 &= \frac{1}{n_2 - n_1} \sum_{n=n_1+1}^{n_2} x(n) \\ A_3 &= \frac{1}{N - n_2} \sum_{n=n_2+1}^{N} x(n) \end{aligned} \quad \text{(A.2)}$$

Let $\tilde{x} = x + u$ be a *noised signal*, where u represents an additive Gaussian white noise with the variance σ^2. Let x' be an *approximation* of x with SCD method and measured *noised signal* $\tilde{x}$. Thus, SCD can be considered as an optimization problem that provides indices $n_1' \in [1, n_2' - 1]$ and $n_2' \in [n_1' + 1, N-1]$ such that x' approximates x with maximum probability of measurements $\tilde{x}$, (A.3).

$$p(\tilde{x}; n_1, n_2) = \frac{1}{(2\pi\sigma^2)^{\frac{N}{2}}} \exp\left[-\frac{1}{2\sigma^2} \left(\sum_{n=1}^{n_1} \left(\tilde{x}(n) - \tilde{A}_1\right)^2 + \sum_{n=n_1+1}^{n_2} \left(\tilde{x}(n) - \tilde{A}_2\right)^2 + \sum_{n=n_2+1}^{N} \left(\tilde{x}(n) - \tilde{A}_3\right)^2 \right) \right] \quad \text{(A.3)}$$

The optimization problem of finding indices n_1 and n_2, for which $p(\tilde{x}; n_1, n_2)$ has the maximum value, can be rewritten as a problem of finding indices n_1 and n_2 for which eq. (A.4)-(A.5) has the minimum value.

$$V_{SCD}(n_1, n_2) = \sum_{n=1}^{n_1} \left(\tilde{x}(n) - \tilde{A}_1\right)^2 + \sum_{n=n_1+1}^{n_2} (\tilde{x}(n) - \tilde{A}_2)^2 + \sum_{n=n_2+1}^{N} \left(\tilde{x}(n) - \tilde{A}_3\right)^2 \quad \text{(A.4)}$$

$$V_{SCD,min}(n_1', n_2') = \min_{1 < n_1 < n_2 < N} V_{SCD}(n_1, n_2) \quad \text{(A.5)}$$

For an arbitrary large number of jumps Ξ, a general solution of eq. (A.5) can be obtained by applying dynamic programming and integral images [28]. In this work, where $\Xi = 2$, the direct solution is used which considers all combinations of n such that $1 < n_1 < n_2 < N$. This minimization problem, eq. (A.5), has the time complexity of $O\left(\frac{N^2}{2}\right)$. In order to speed-up calculations, the direct solution is used in conjunction with the integral image method for the mean value calculation, c.f. Alg. A.1. Let the referent signal $x(n)$ have $N=100$ samples, amplitude levels $A_1=0$, $A_2=3\sigma$ and $A_3=0$ at indices $n_1=30$ and $n_2=70$, c.f. black line in Fig. A.1. Let the measured signal $\tilde{x}=(n)$ be a sum of x and u with the standard deviation $\sigma^2 = 2.5$, c.f. blue line in Fig. A.1. As a result of SCD method, signal x is approximated with x' at indices $n'_1=31$ and $n'_2=69$, and mean values A'_1, A'_2 and A'_3 calculated at intervals $[0, n'_1]$, $[n'_1+1, n'_2]$ and $[n'_2+1, N]$,c.f. red line in Fig. A.1. Normalized matrix $P(n1, n2)$ with calculated probabilities, eq. (A.3), has the maximum value at $n_1=n'_1$ and $n_2=n'_2$. Matrix $P(n1, n2)$ is an upper triangular matrix since $1<n_1<n_2<N$, c.f. Fig. A.2.

To determine the stability of the SCD method, a stohastic experiment is done and the ratio $\Delta A/\sigma$ is considered. Additive Gausian noise u is generated in 10000 instances with $N=100$ samples and $\sigma = 1.58$ ($\sigma^2 = 2.5$). Each group of 1000 instances is added to the the same amplitude jump $\Delta A_j = j \cdot \sigma$, $j = 1, \ldots, 10$. In this way $\Delta A_j/\sigma$ is ranging from 1 to 10. Amplitude levels of x are $A_1=0$, $A_2=\Delta A_j$ and $A_3=0$, and are changing at indices $n_1=30$ and $n_2=70$. Errors of estimated amplitude jump indices $n'_{1,i}$ and $n'_{2,i}$ are defined with (A.6), where $i=1,\ldots,1000$.

$$\begin{aligned} \varepsilon^+ &= \max\left(\max\left(\left|n_{1,i} - n'_{1,i}\right|\right), \max\left(\left|n_{2,i} - n'_{2,i}\right|\right)\right) \\ \varepsilon^- &= \min\left(\min\left(\left|n_{1,i} - n'_{1,i}\right|\right), \min\left(\left|n_{2,i} - n'_{2,i}\right|\right)\right) \\ \varepsilon &= \max(\varepsilon^+, |\varepsilon^-|) \end{aligned} \quad \text{(A.6)}$$

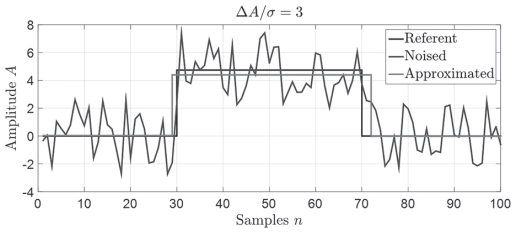


Fig. A.1. Signal regonstruction by using Signal Change Detection (SCD).

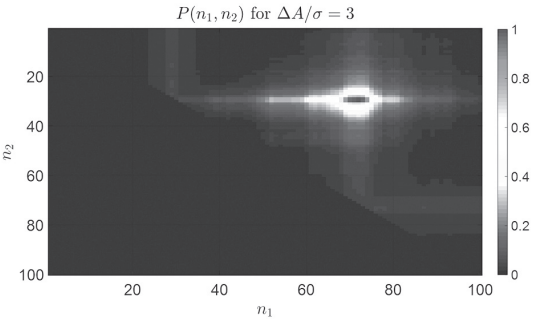


Fig. A.2. Normalized probability matrix $P(n_1, n_2) \in [0,1]$.

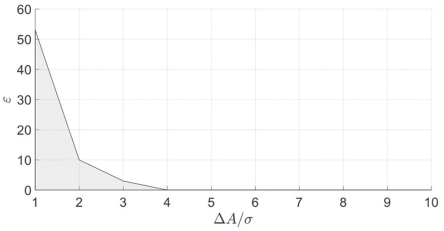


Fig. A.3. Error of estimated amplitude jump indices.

Alg. A.1. Direct solution for Signal Change Detection (SCD) and two amplitude jumps ($\Xi = 2$) with integral image-based mean calculation.

Input: $\tilde{x}$
Outputs: $n_1', n_2', A_1', A_2', A_3'$
Set y as 1D integral image of $\tilde{x}$
for each $n_1 = 1$ to $n_1 = N - 2$
  $A_1 = y(n_1 + 1)/n_1$
  $S_1 = \sum_{i=1}^{n_1}(x(i) - A_1)^2$
    for each $n_2 = n_1 + 1$ to $n_2 = N - 1$
      $A_2 = (y(n_2 + 1) - y(n_1 + 1))/(n_2 - n_1)$
      $S_2 = \sum_{i=n_1+1}^{n_2}(x(i) - A_2)^2$
      $A_3 = (y(N + 1) - y(n_2 + 1))/(N - n_2)$
      $S_3 = \sum_{i=n_2+1}^{N}(x(i) - A_3)^2$
      $V_{SCD} = S_1 + S_2 + S_3$
      if $V_{SCD} < V_{SCD,min}$
        $V_{SCD,min} = V_{SCD},\ n_1' = n_1,\ n_2' = n_2$
$A_1' = y(n_1' + 1)/n_1'$
$A_2' = (y(n_2' + 1) - y(n_1' + 1))/(n_2' - n_1')$
$A_3' = (y(N + 1) - y(n_2' + 1))/(N - n_2')$

Error is considered to be 0 when the estimated amplitude jump index n' is equal to the original amplitude jump index n. From the result illustrated in Fig. A.3, one can observe that the SCD method is stable with no errors, c.f. eq. (A.6), when $\Delta A/\sigma \geq 4$. In this work, the ratio $\Delta A/\sigma$ is several times larger than the minimum value required.

7. REFERENCES

[1] P. Blondel, "The Handbook of Sidescan Sonar". Praxis Publishing Ltd, 2009.

[2] B. Lehmann, K. Siantidis, I. Aleksi, D. Kraus, "Efficient pre-segmentation algorithm for sidescan-sonar images", 2011 7th International Symposium on Image and Signal Processing and Analysis (ISPA), Dubrovnik, Croatia, 4-6 September 2011, pp. 471-475.

[3] M. Kass, A. Witkin, D. Terzopoulos, "Snakes: Active contour models", International Journal of Computer Vision, Vol. 1, No. 4, 1988, pp. 321–331,.

[4] B. Dong, R. Jin, G. Weng, "Active contour model based on local bias field estimation for image segmentation", Signal Processing: Image Communication, Vol. 78, 2019, pp. 187-199.

[5] C. Xu, J. Prince, "Gradient vector flow: a new external force for snakes", Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, San Juan, Puerto Rico, USA, 17-19 June 1997, pp. 66–71.

[6] L. D. Cohen, "On active contour models and balloons", CVGIP: Image Understanding, Vol. 53, No. 2,1991, pp. 211–218.

[7] C. Xu, J. Prince, "Snakes, shapes, and gradient vector flow", IEEE Transactions on Image Processing, Vol. 7, No. 3, 1998, pp. 359–369.

[8] S. Z. Li, "Markov random field modeling in image analysis", Springer-Verlag London, 2009.

[9] R. C. Gonzalez, R. E. Woods, "Digital Image Processing", 4th Ed., Pearson, 2017.

[10] N. Dhanachandra, K. Manglem, Y. J. Chanu, "Image Segmentation Using K -means Clustering Algorithm and Subtractive Clustering Algorithm", Procedia Computer Science, Vol. 54, 2015, pp. 764-771.

[11] L. Vincent, P. Soille, "Watersheds in digital spaces: an efficient algorithm based on immersion simulations", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 13, No. 6, 1991, pp. 583 –598.

[12] P. Dempster, N. M. Laird, D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm", Journal of The Royal Statistical Society, Series B, Vol. 39, No. 1, 1977, pp. 1–38.

[13] C. Carson, S. Belongie, H. Greenspan, J. Malik, "Blobworld: image segmentation using expectation-maximization and its application to image querying", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 24, No. 8, 2002, pp. 1026–1038.

[14] M. Carreira-Perpinan, "Gaussian mean-shift is an em algorithm", IEEE Transactions on Pattern Analysis and Machine Intelligence, V0ol. 29, No. 5, 2007, pp. 767–776.

[15] K. Fukunaga, L. Hostetler, "The estimation of the gradient of a density function, with applications in pattern recognition", IEEE Transactions on Information Theory, Vol. 21, No. 1, 1975, pp. 32 – 40.

[16] D. Comaniciu, P. Meer, "Mean shift: a robust approach toward feature space analysis", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 24, No. 5, 2002, pp. 603-619.

[17] S. Geman, D. Geman, "Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. PAMI-6, No. 6, 1984, pp. 721–741.

[18] V. Kolmogorov, R. Zabin, "What energy functions can be minimized via graph cuts?", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 26, no. 2, pp. 147–159, 2004.

[19] S. Yu, J. Shao, Z. Han, L. Gao, Y. Lin, Y. Tang, C. Wu, "Mine like object detection and recognition based on intrackability and improved BOW", 2016 IEEE International Conference on Cyber Technology in Automation, Control, and Intelligent Systems (CYBER), Chengdu, China, 2016, pp. 222-227.

[20] I. Dzieciuch, D. Gebhardt, C. Barngrover, K. Parikh, "Non-linear Convolutional Neural Network for Automatic Detection of Mine-Like Objects in Sonar Imagery", Proceedings of the 4th International Conference on Applications in Nonlinear Dynamics (ICAND 2016). ICAND 2016. Lecture Notes in Networks and Systems, Vol 6., 2017, pp. 309-314.

[21] A. Sinai, A. Amar, G. Gilboa, "Mine-Like Objects detection in Side-Scan Sonar images using a shadows-highlights geometrical features space", OCEANS 2016 MTS/IEEE Monterey, Monterey, Canada, 01 December 2016, pp. 1-6.

[22] R. Cupec, I. Aleksi, G. Schmidt, "Step sequence planning for a biped robot by means of a cylindrical shape model and a high-resolution 2.5D map", Robotics and Autonomous Systems, Vol. 59, No. 2, 2011, pp. 84-100.

[23] T. Matić, I. Aleksi, Ž. Hocenski, D. Kraus, "Real-time Biscuit Tile Image Segmentation Method Based on Edge Detection", ISA Transactions, Vol. 76, 2018, pp. 246-254.

[24] U. Qidwai, C. H. Chen, "Digital Image Processing: An Algorithmic Approach with MATLAB", Chapman and Hall/CRC, 2009.

[25] J. C. Latombe, "Robot Motion Planing", Kluwer Academic Publishers, 1991.

[26] R. Cupec, G. Schmidt, "An Approach to Environment Modeling for Biped Walking Robots", International Conference on Intelligent Robots and Systems, Edmonton, Alta., Canada, 2-6 August 2005, pp. 424-429.

[27] S. J. Russell, P. Norvig, "Artificial Intelligence: A Modern Approach", 3rd Ed., Prentice Hall, 2009.

[28] I. Aleksi, D. Kraus, Ž. Hocenski, "Multi-Language Programming Environment for C++ Implementation of SONAR Signal Processing by Linking with MATLAB External Interface and FFTW", ELMAR 2011, 53rd International Symposium, Zadar, Croatia, 14-16 September 2011, pp. 195-200.

[29] S. M. Kay, "Fundamentals of Statistical Signal Processing: Estimation Theory", Prentice-Hall Inc., 1993.