Automatic Obstacle Detection for USV's Navigation Using Vision Sensors

Oren Gal

Abstract. This paper presents an automatic method to acquire, identify, and track obstacles from an Unmanned Surface Vehicle (USV) location in marine environments using 2D Commercial Of The Shelf (COTS) video sensors, and analyzing video streams as input. The guiding line of this research is to develop real-time automatic identification and tracking abilities in marine environment with COTS sensors. The output of this algorithm provides obstacle's location in x-y coordinates. The ability to recognize and identify obstacles becomes more essential for USV's autonomous capabilities, such as obstacle avoidance, decision modules, and other Artificial Intelligence (AI) abilities using low cost sensors. Our algorithm is not based on obstacles characterization. Algorithm performances were tested in various scenarios with real-time USV's video streams, indicating that the algorithm can be used for real-time applications with high success rate and fast time computation.

1 Introduction

One of the most difficult challenges for USV navigation is to recognize and identify obstacles around the vehicle without human intervention. This task is known as Automatic Target Detection (ATD). An efficient ATD system should achieve high detection percentage for targets while maintaining a minimal false-alarm rate. This means that it must preserve an optimal balance between a high detection rate and a low error probability.

Although, ATD algorithms are very sensitive and unstable regarding clutter elements, i.e. elements that are not targets but still part of the scenes with similar characteristics as the targets. Dealing with clutter in ATD algorithms was extensively studied [1, 2, 3].

Oren Gal
Technion, Israel Institute of Technology
e-mail: orengal@tx.technion.ac.il

One of the ATD algorithms methods is based on the target temperature. The contrast of the target were based on environment gradient of the target and the environment's contrast to recognize targets. These methods suffer from false alarms due to targets and environment similarity. Several methods were developed to characterize the targets and to distinguish between the target's and the environment's characteristic, avoiding false alarms [3, 4].

Heuristic methods were introduced in the early 1980s based on threshold gradient in the image. The threshold was determined by the contrast of an object with its local background [3]. The segmentation part of such algorithms is based on standard edge operator using closing shape algorithms and filling steps [5, 6, 7, 8].

Most of the previous works were studied in aerial and ground environments without considering special phenomenas in marine environments, such as waves, clutter, vehicle's stability, etc. For the first time, our algorithm deals with ATD algorithms for USV's motion planning and automatic decision models using COTS sensors. The algorithm based on common video format as input, and computes targets locations around the vehicle.

2 General Description

Our new algorithm gets an input video in one of the common formats (avi, mpeg, mov, etc.), starts a few processing steps and eventually finds all the targets in the specific image.

The main steps as shown in Figure 1 are:

1. Read input image from the COTS device
2. Resize and convert the image into gray-scale reducing CPU time
3. Reduce search space for targets by preforming initial cleaning and horizon identification
4. Learn the sea pattern using a co-occurrence matrix
5. Morphologic cleaning, which cleans the image after distinguishing between sea and possible targets
6. Skeletonize the recognized targets in order to characterize them with a simple structure
7. For each target: Lower the characteristic structure into the "interesting points" according to the target skeleton structure that was found

Basic methods in image processing enable to indicate targets position at a specific frame with a small effort. Naive algorithms report target's current parameters immediately. Instead, we analyze the 20^{th} input frame at each time to find possible targets. We count the input frames and treat only the current 20^{th} input frame, comparing and correlating the results with the last 20^{th} input frame .We decided to use the 20^{th} input frame as optimal frame number based on the parameters of minimal

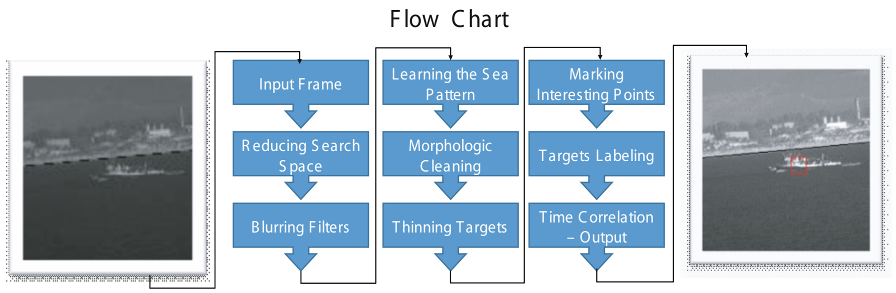

Fig. 1 Flow Chart of the Algorithm: Describing algorithm's functions from input frame to algorithm output using major steps: Reduce search space, Learning sea pattern, Morphologic cleaning and Skeletone algorithm

target changing in marine environments and CPU time computation. We analyzed these parameters on our real-time video movies with several frame numbers and decided to use the 20^{th} input frame.

By that, the algorithm ignores waves and clutter interferes and decreases the rate of false alarms [8, 11, 12]. Later on, we define the Interesting Points concept. The Interesting Points, which survived the time correlation are eventually defined as targets. Multiple targets can be found in each frame and each identified target is reported only once.

3 The Algorithm

3.1 *Initial Cleaning*

The marine environment suffers from variety of noises of different kind. In the first step, we apply a number of image blurring techniques to get a smoother texture of the sea. The input image might contain many noised pixels that will interfere with reducing the search space. This may become critical for real-time performances.

Therefore, a number of Gaussian softening filters are used. In order to preserve the horizontal structure of the sea pattern, we use convolution with a specific matrix designed to preserve the horizontal structure.

The initial cleaning part includes the Adaptive Smooth filter. As well known in the image processing world, this filter is designed to blur the image and save the sharp edges. This characteristic becomes very crucial in later stages to recognize and identify the targets. The Adaptive Smooth filter $w(x,y)$ is a very efficient filter, but hard to deal with "salt and pepper" clutter. The filter computes as:

$$w(x,y) = e^{-\frac{G_x^2+G_y^2}{2\cdot fac^2}} \tag{1}$$

where:

$$G_x(x,y) = \frac{l(x+1,y) - l(x-1,y)}{2} \tag{2}$$

$$G_y(x,y) = \frac{l(x,y+1) - l(x,y-1)}{2} \tag{3}$$

where fac in Eq. 1 sets the smoothing quality.

The Adaptive Smooth filter contains two Gaussian matrixes: Cleaning and Blurring matrix calculated by Eq. 1. The Cleaning Matrix, $C[i,j]$ is an $N \times N$ matrix (in our case $N = 7$), the specific values that were found to be the most efficient in the simulations are detailed below:

$$C[i,j] = \begin{pmatrix} 0.04 & 0.04 & 0.04 & 0.04 & 0.04 & 0.04 & 0.04 \\ 0.02 & 0.02 & 0.02 & 0.02 & 0.02 & 0.02 & 0.02 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\ 0.02 & 0.02 & 0.02 & 0.02 & 0.02 & 0.02 & 0.02 \\ 0.04 & 0.04 & 0.04 & 0.04 & 0.04 & 0.04 & 0.04 \end{pmatrix} \tag{4}$$

The Blurring Matrix $G[i,j]$ is a $N \times N$ matrix, where $N = 6$, the matrix blurs the image and is known as a very effective one dealing with noises:

$$B[i,j] = \begin{pmatrix} 0.04 & 0.04 & 0.04 & 0.04 & 0.04 & 0.04 \\ 0.04 & 0.04 & 0.04 & 0.04 & 0.04 & 0.04 \\ 0.04 & 0.04 & 0.04 & 0.04 & 0.04 & 0.04 \\ 0.04 & 0.04 & 0.04 & 0.04 & 0.04 & 0.04 \\ 0.04 & 0.04 & 0.04 & 0.04 & 0.04 & 0.04 \end{pmatrix} \tag{5}$$

3.2 Reducing the Search Space

Motion planning and autonomous decision models are often to be done in real-time. For that, the algorithm must consider computation time constraints and optimize the CPU time at each time step.

Therefore, the initial step is to recognize the "interesting" parts in the frame; in our case, the part that contains sea texture. We characterize the sea texture by defining the horizon line l and the horizon line orientation θ. The identification is computed using the Canny Edge Detector algorithm computing the gradient intensity at each point (x,y) in the frame:

$$l(x,y) = \sqrt{\nabla x^2 + \nabla y^2} \tag{6}$$

$$\theta = tan^{-}1\frac{\nabla y}{\nabla x} \tag{7}$$

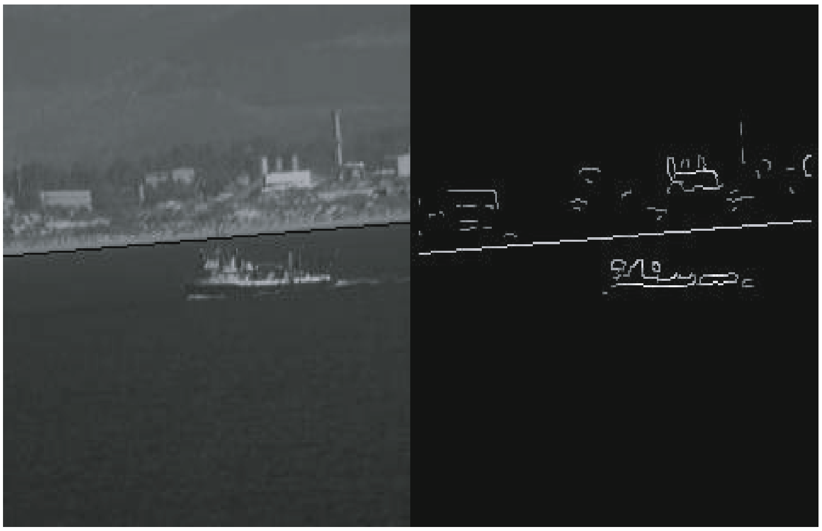

Fig. 2 Initial Cleaning of the Image. The left side shows the original input image after the transformation into a gray scale format. The right side shows the output image of the Canny Edge Detector.

Figure 2 shows the original input and the output image of the Canny Edge Detector. The next step is to reduce the search space using a simple Hough Line Transformation to get the most intense line in the image.

3.3 Learning Sea Pattern

This part is the actual recognition part of the algorithm. We assume that there are no targets within 10 pixels from each edge of the frame. This assumption is legitimate, based on the fact that the input device is usually a video device, which can be easily adjusted and focused on the center of the frame.

Learning the sea pattern can be done by an occurrence matrix. The occurrence matrix is $N \times N$, where N is the possible number of gray levels inside the frame [9].

We define a length function f from current pixel (x,y):

$$f(x) = x + l_d \tag{8}$$

where l_d is the distance to the proportional pixel. Eq. 8 is used to determine the occurrence matrix values, depending only on x values based on the sea texture assuming that the sea texture is horizontal. In the next stage we scan the image and set the pixel values the following way:

1. For each pixel in the learning zone, check if this pixel is inside the search space
2. If so, set pixel (x,y) value to k
3. Choose the next pixel according to f value

The occurrence matrix allows us to find similar texture inside the image. The learning zone can vary from image to image as shown in Figure 3. For an (n,m) image we use 10 percent of n on each side (left and right), and 10 percent of m at the bottom as learning zone.

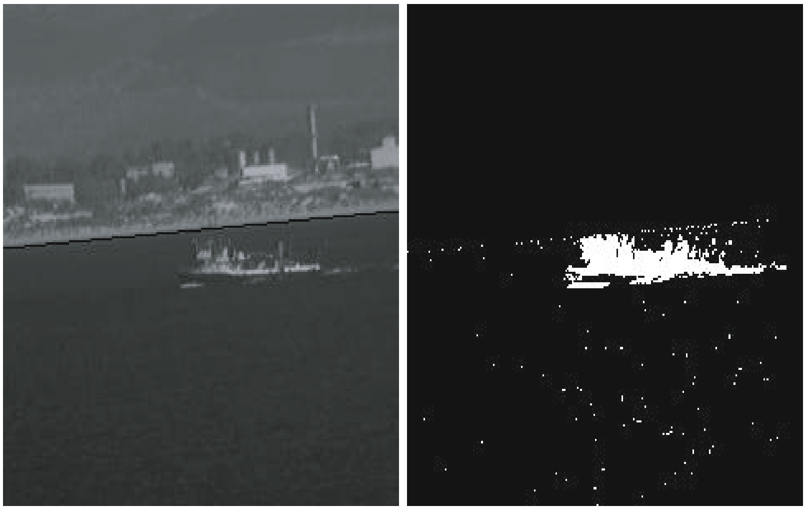

Fig. 3 Learning Sea Pattern: on the left we can see the original image and the right side shows the image after applying occurrence matrix

3.4 Morphologic Cleaning

After completing the detection process of the sea pattern, there are still few sea pixels that were not detected. We use a unique cleaning filters to improve and smoothen the image, so target identification will be easier. The first filter is the "Fill Filter". This filter fills the target with missing pixels, i.e. for each black pixel (sea pixel). We scan the eight neighbors of each pixel. If the pixel's eight neighbors are white (target pixels), we change the pixel value to be a target pixel.

We use the Fill Filter Matrix, F_f:

$$F_f[i,j] = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \end{pmatrix} \tag{9}$$

The second filter is called "Cleaning Filter". This filter makes exactly the opposite action from the Fill Filter. For each white pixel (target pixel), if all of its eight neighbors are black (sea pixels), we change this pixel value to be a sea pixel.

The Cleaning Filter Matrix C_f is:

$$C_f[i,j] = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix} \tag{10}$$

The third filter is called "Connection Filter". In some cases Fill filter and Cleaning filter can not recover damaged pixels if the problem occurres in more than one adjacent pixel. Therefore, the Connection filter detects two separate components of a target and connects them. The Connection Filter Matrix C_{nf} is:

$$C_{nf}[i,j] = \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \tag{11}$$

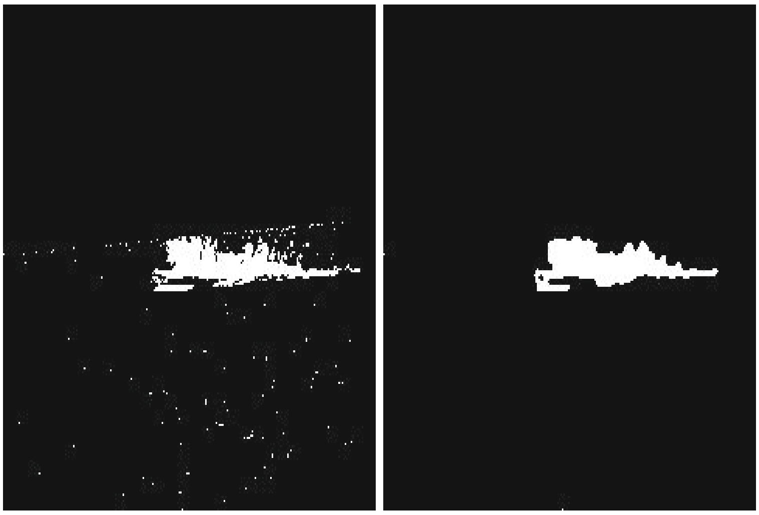

Fig. 4 Morphologic Cleaning: The left side shows the image after detecting the sea pattern and the right side shows the image after applying the morphologic methods: Cleaning and Fill filter

The forth filter is based on the assumption that each pixel value is determined by its neighbor pixel values. This filter is called "Majority Filter". For each pixel, is calculates the eight neighbor values and changes the current pixel value according to the majority.

The left side of Figure 4 shows the image after detecting the sea pattern and before applying filters. The right side of Figure 4 shows the image after applying the cleaning filters. We can see that all the damaged pixels were removed and we managed to derive a clean target image.

3.5 Marking Interesting Points

The video device and the detection algorithm are not perfectly accurate and damaged pixels will always be a part of the image. We assume that the basic structure of the target does not change from one frame to another. The algorithm characterizes a target by using a Skeleton algorithm. This algorithm applies a thinning process to an image and finds a simpler structure of the recognized targets. This process simplifies the target tracking in the next frames and decreases the CPU time.

We use an extended version of the Skeleton algorithm. The extended version is based on doubled thinning process of the target. For each pixel x belongs to the thin target profile, the pixel at $x+1$ location also marked as a pixel in a thin profile of the target.

The left side of Figure 5 shows the target image before the thinning process and the right side shows the thin target after applying the Skeleton algorithm.

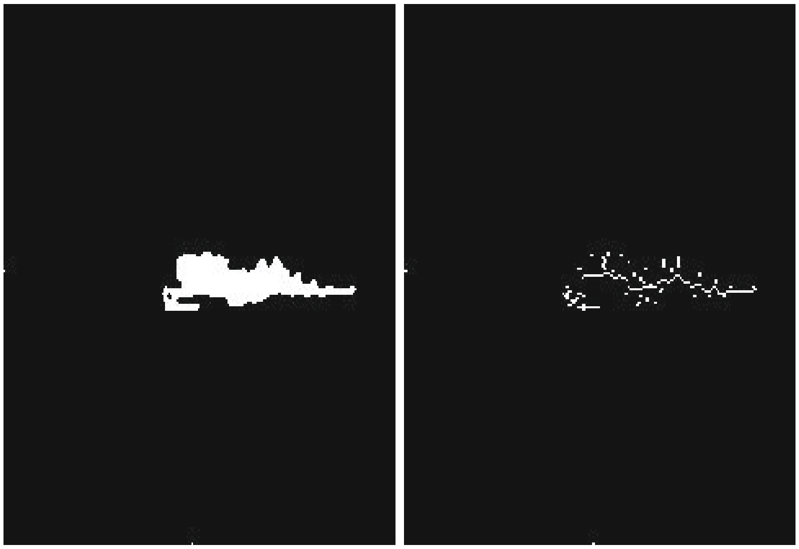

Fig. 5 Skeleton Algorithm: The left side shows the target image before the thinning process and the right side shows the thin target after Skeleton algorithm without Interesting Point

Marine environment can be unstable and thin structure targets usually can't be seen in the frame. Therefore, the target structure should be more flexible, supporting target recognition even if not all of the target is in the frame. For that, we define the Interesting Point concept:

Definition: Interesting Point - is a point in the target thin profile which is an intersection of two other lines. i.e. considering thin target, for each white pixel (Skeleton pixel) we perform the Convolution Operator C_o with the matrix:

$$C_o[i,j] = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} \tag{12}$$

The matrix C_o identifies most significant points within a two line intersection with a predefined threshold S_t in the Skeleton algorithm.

As mentioned before, CPU time is a major issue in our algorithm implementation. Tracking targets in the next frames is based on a simple structure of the target, reducing CPU time. Moreover, it is much more accurate to characterize a target with a few points, rather than using a large number of points in such a clutter and noisy environment.

Figure 6 shows the skeleton image after the thinning process, the left side demonstrate the target interesting points implementation.

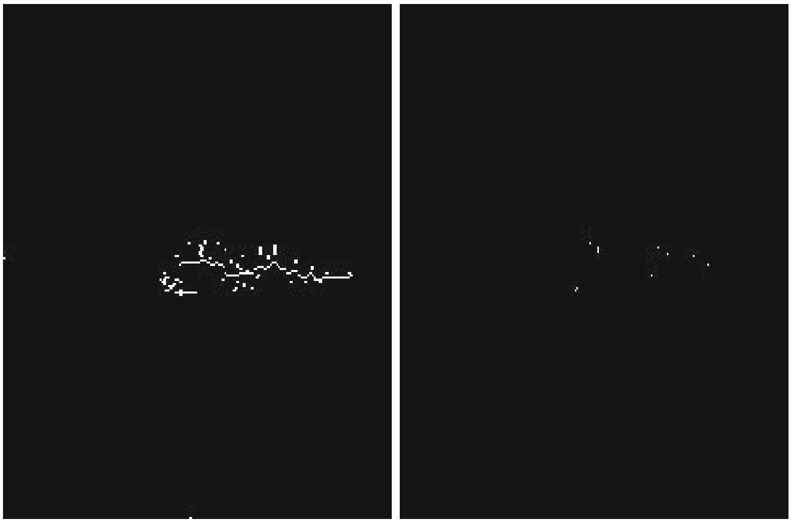

Fig. 6 Skeleton Algorithm with Interesting Points: The left side shows the target image after Skeletone thinning process and the right side shows the Skeleton algorithm with Interesting Point

3.6 Time Correlation

This is the final step in the recognition algorithm part. After we have achieved two sequential frames and identified the targets in each one of them, we apply the time correlation step. Our goal is to make the final decision about a possible target that was found in the first frame and to decide whether it is a real target or a false alarm.

Given a set of interesting points in the first frame $N = n_1, , n_n$, and another set of interesting points in the second frame $M = m_1, , m_m$. For each interesting point $n_1, , n_n$, we build a surrounding rectangle of size AxA. In the second frame, we locate a BxB rectangle at the same place around the interesting point. In this stage, we start to search the interesting point of the smaller rectangle inside the bigger rectangle. If the interesting point is found, it will appears in the sequential frame. The same search process is implemented for all of the interesting points $n_1, , n_n$ in the same way. The target position is determined by the average of all it's interesting points.

Figure 7 shows the first frame interesting points. The red rectangle is marked and all of the interesting points inside it are registered. The next step is to build the blue rectangle in the sequential frame and search all of the registered interesting points inside. Once 80 percent of the matching is complete, we declare those points as relevant and continue searching the next points as a candidate target.

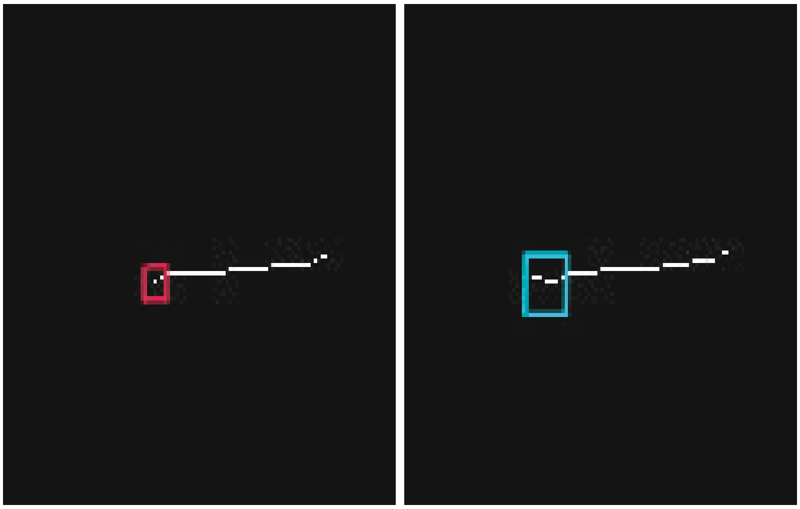

Fig. 7 One Step of Time Correlation Process: In the left side, the interesting points are marked with red rectangle and in the right side the next time frame introduced with blue rectangle searching the same interesting points from the previous frame

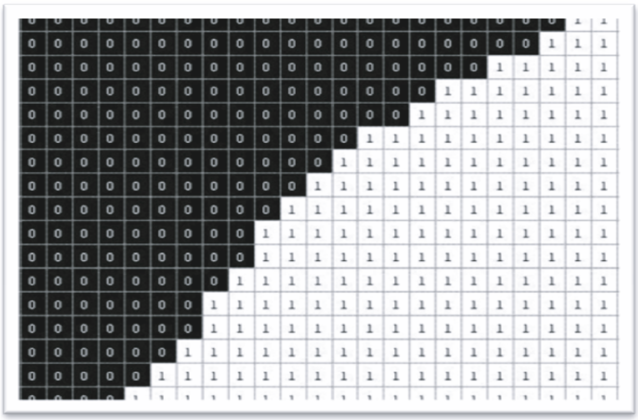

Fig. 8 Labeling Algorithm Presentation: Each label is colored with a different color, in this case there is only one target colored in red

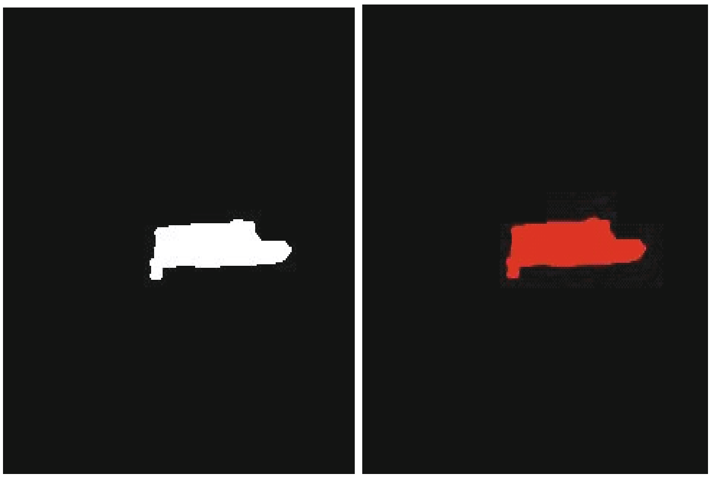

Fig. 9 Labeling Algorithm Matrix: $n \times n$ matrix for each target, each cell in the matrix $[i, j]$ defines the connected component related to the interesting point, in this case we classify only one target with serial numbers of 1 and 0

3.7 *Multiple Targets*

One of the main goals of the algorithm is the ability to track multiple targets at the same time. This can be done by classifying the interesting points. Classification of the interesting points is based on connected components using labeling algorithm. The input image for the labeling algorithm is a target image after morphologic cleaning. In case of multiple targets, the labeling algorithm recognize and label each component in a different serializing number. Then, each interesting point is classified according to its matching number. Example of labeling algorithm matrix $n \times n$ can be seen in Figure 8. Each cell in the matrix $[i, j]$ defines the connected component related to the interesting point. This method enables us to track multiple targets in an efficient way.

Figure 9 shows the output of the labeling algorithm. Each label is colored with a different color, in this case a single component colored in red.

4 Results

Our algorithm was tested with five hours of video stream from a database with real video recorded with a camera on a USV in different light and sea states. The algorithm was tested on real-time video movies from the USV system, the first frame can be seen in Figure 10.

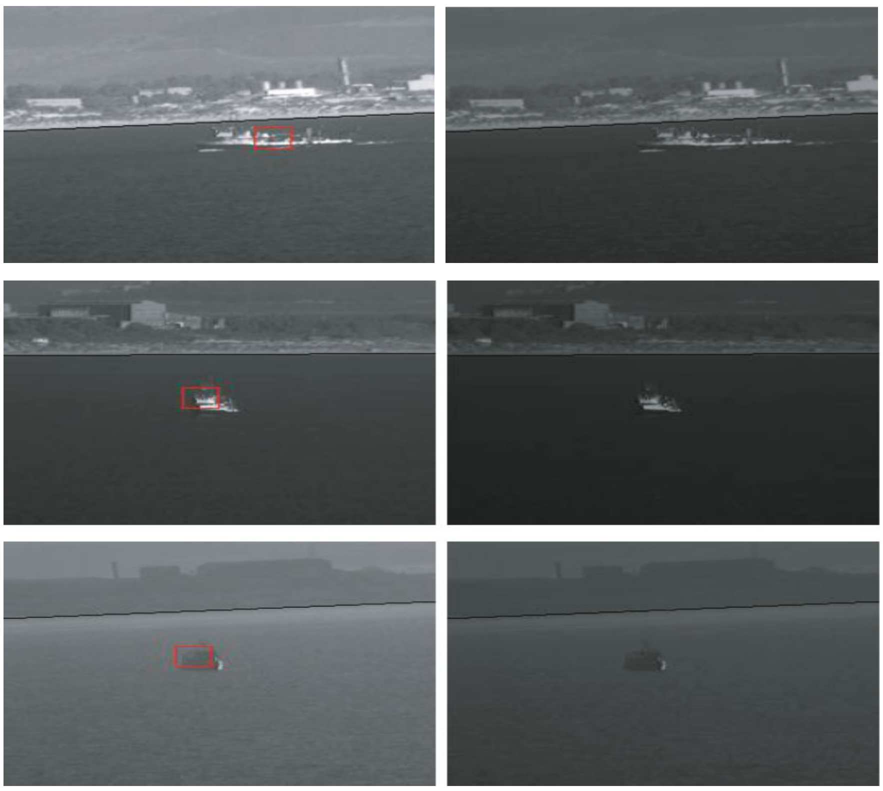

Fig. 10 Real-time Video Algorithm Performance: Real video camera on USV in different light and sea states and targets radius. Selected movies can be found on-line at [13].

Target recognition and identification marked with a red rectangular and can be found at: *http://sites.google.com/site/orenusv/home/research-1/usvvision.*

The parameter values in our simulation are: $l_d = 10$, the lower threshold value was set to 20 and the upper value was set to 100, the surrounding rectangles size were set to $A = 7, B = 16$ and $fac = 3$, $S_t = 4$.

The tested scenes contained different sea states at day and night with different light conditions. The target radius changes between $10-60[m]$. The horizon line was examined with and without buildings in the background of the USV.

The run-time for average on frame was measured as 0.005 millisecond, with the minimum value 0.003 millisecond and the maximum value 0.014 millisecond. Based on these results, the algorithm can be used for real-time application for autonomous movement of USV, identifying targets around the vehicle.

5 Discussion and Conclusion

This paper presents a basic and very efficient algorithm for marine environments that was tested on different scenes. Yet, there are some limitations and untested cases that should be treated in future research. Our algorithm is limited to sensor noises and false alarm dealing with very cluttered environments. The algorithm has been tested with stabilize sensor on the platform, which simplify horizon line recognition. In case of not stabilized sensor or panoramic image integrating several video streams, we expect a very limited success rate.

Another well know problem in marine environments is related to sun effects. The algorithm might classify sun light effects as targets, which will cause false alarms. This challenge should be treated in our further research, for more accurate ATD abilities for accurate USV trajectory planning.

The highlight of the algorithm is the simple concept that can be use on COTS sensor without special hardware, and as far as we know this specific issue was not yet studied extensively in marine environments. The algorithm is based on simple basic algorithms from the image processing world, which are suitable for real-time application.

We presented a new algorithm to acquire identify and track obstacles from USV systems using COTS sensor for autonomic navigation. The algorithm is based on previous image processing filters and algorithms, and been been adapted to the marine environment challenges.

The algorithm was successfully tested on real-time video from USV systems, and can be apply in real-time applications dealing with CPU time constraints. Further research directions include light effects and not stabilized systems.

Acknowledgements. The author would like to thank Yaron Vazana for his help during this research and to Dr. Blaurock and Dr. Schlaefer for their helpful comments.

References

1. Bhanu, C., Holben, R.D.: Model-based segmentation of FLIR images. IEEE Trans. Aerosp. Electron. Syst. 26, 2–10 (1990)
2. Ben-Yosef, N., Bahat, B., FeiginBhanu, G., Holben, C.: Simulation of IR images of natural backgrounds. Appl. Opt. 22, 190–193 (1983)
3. Ratches, J.A., Walters, C.P., Buser, R.G., Guenther, B.D.: Aided and automatic target recognition based upon sensor inputs from image forming systems. IEEE Trans. Pattern Anal. Mach. Intell. 19, 1004–1019 (1997)
4. Casasent, D.P., Neiberg, L.M.: Classifier and shift-invariant automatic target recognition neural networks. Neural Networks 8, 1117–1129 (1995)
5. Schachter, B.J., Lev, A., Zucker, S.W., Rosenfeld, A.: An application of relaxation methods to edge reinforcement. IEEE Trans. Syst. Man Cybern. 7, 813–816 (1997)
6. Walters, D.K.W.: Computer vision model based on psychophysical experiments. Pattern Recognition by Humans and Machines 2 (1986)
7. Broy, M.: Early vision. In: Rosenfeld, A. (ed.) Perspectives in Computing, pp. 190–206. Academic Press, New York (1986)
8. Marham, K.C.: Comparison of segmentation processes for object acquisition in infrared scenes. IEEE Radar Signal Process 136, 13–21 (1989)
9. Davies, E.R.: Machine Vision: Theory, Algorithms, Practicalities, 2nd edn. Academic Press, New York (1997)
10. Marham, K.C.: Clutter metrics for target detection systems. IEEE Trans. Aerosp. Electron. Syst. 30, 81–91 (1994)
11. Vijaya, K.: A tutorial survey of composite filter designs for optical correlators. Appl. Opt. 31, 4773–4798 (1992)
12. Flannery, D., Loomis, J., Milkovich, M.: Transform-ratio ternary phase-amplitude filter formation for improved correlation discrimination. Appl. Opt. 27, 4079–4083 (1988)
13. `http://sites.google.com/site/orenusv/home/research-1/usvvision`