




Ships Detection on Inland Waters Using Video Surveillance System

Tomasz Hyla¹  and Natalia Wawrzyniak² 

¹ West Pomeranian University of Technology, Szczecin, Poland
thyla@zut.edu.pl

² Marine Technology Ltd., Szczecin, Poland
n.wawrzyniak@marinetechonology.pl

Abstract. The video surveillance is used to monitor ships in order to ensure safety on waterways. The ships detection is a first step in a ship automatic identification process based on video streams. The paper presents a new algorithm for ships detection on inland waterways. The algorithm must detect moving ships of all kinds, including leisure craft, that are visible on a video stream and is designed to work for stationary cameras. Furthermore, it only requires an access to video streams from existing monitoring systems without any additional hardware or special configuration of cameras. The algorithm works in variable lightning conditions and with slight changes of background. In the paper, the test application implementing the algorithm is presented together with a series of experimental results showing the algorithm quality depending on different parameters' sets. The main purpose of the tests was to find the optimal set of twelve parameters that will become the default setting. All moving ships, including small boats and kayaks, must be detected, which is the main difference from existing solutions that mostly focus on detection of only one vessel type. In the proposed algorithm, all objects that are moving on water are detected and then non-ships are eliminated by usage of some logic rules and excluding additional image processing methods.

Keywords: Video surveillance · Ship detection · Inland waterway · Dynamic background · Real-time detection

1 Introduction

The ships traffic is monitored in order to ensure safety on waterways. Usually, the traffic is monitored using radar networks, Automatic Ship Identification (AIS) receivers, and video surveillance systems. These three systems complement each other. The video surveillance is mostly used to monitor non-conventional (according to International Convention for the Safety of Life at Sea (SOLAS)) ships and to confirm results of detection and recognition obtained from other systems. It is mostly used in restricted or special areas of heavy or complicated traffic. Currently, an operator must visually monitor ships that are passing in front of the cameras. Especially, when it is required to detect and identify a wide array of ships starting from large cargo vessels and ending at kayaks.

The problems related to ship detection on inland waters were discussed by Wawrzyniak and Hyla [1]. The ship in a video stream can be detected using two basic approaches. The first one is to use a pixel-based detection method that allows detecting any moving object on constant or slightly changing background. The second approach is to use an object-based detection using a classifier. The second approach is better when it is possible to find a distinctive property of a class of objects, e.g., a mast of a sailing vessel, because it usually provides better detection result.

1.1 Related Works

One of the possible solutions to the ship detection problem was presented by Ferreira et al. [2]. The authors use two cameras: one camera with low resolution that detects movement and another camera with high resolution that is used to take a photo when the first camera detects movement. Their solution is designed to detect fishing vessels. They achieved the best results when using object-based detection based on Histogram of Oriented Gradients (HOG) classifier [3]. In contrast, Hu et al. [4] used pixel-based detection in their visual surveillance scheme for cage aquaculture that automatically detects and tracks ships. They used the median scheme to create a background image from previous N frames with some additional improvements that allowed to reduce the influence of sea waves. The problem of ship detection in the presence of waves was also addressed by Szpak and Tapamo [5]. They present techniques that solve a problem of moving vessels' tracking in the presence of a moving dynamic background (the ocean). Other works related to the problem of ship detection include [6, 7] and a survey [8].

Moving objects can be detected using a background subtraction algorithm or, to be more precise, using a background/foreground segmentation algorithm. The background subtraction algorithms were evaluated by [9] and compared by [10]. Several background subtraction algorithms are implemented in the OpenCV library [11]. To begin with, Gaussian Mixture-based Background- Foreground Segmentation (MOG) algorithm (that uses a mixture of K , $K = 3$ to 5) gaussian distributions to model each background picture. The probable values of background pixels are the ones that are more static and present in most of the previous frames [12]. Next, Gaussian Mixture-based Background- Foreground Segmentation Algorithm version 2 (MOG2) [13] is available which is an improved version of MOG.

The algorithm Godbehere-Matsukawa-Goldberg (GMG) [14] uses by default 120 frames for background modelling and per-pixel Bayesian segmentation. Another algorithm, *CouNT* (CNT) was designed by Sagi Zeevi [15] to reflect the human vision. It is designed for variable outdoor lighting conditions and it works well on IoT hardware. Other algorithms include k Nearest Neighbours (KNN) that implements K -nearest neighbours background subtraction from [16], the algorithm created during Google Summer of Code (GSOC) [11], and Background Subtraction using Local SVD Binary Pattern (LSBP) [17].

1.2 Motivation and Contribution

This paper is a part of an ongoing research in Ship Recognition (SHREC) [18] project, which concerns automatic recognition and identification of non-conventional (according to International Convention for the Safety of Life at Sea) ships in areas covered by River Information System and Vessel Traffic Service systems. The ships detection is a first step in a ship identification process based on video streams processing [19].

The main contribution is a new algorithm for ships' detection on inland waterways. The algorithm is designed to detect all kinds of moving ships and to work efficiently, so it can be used to process data from multiple cameras (20 or more) with usage of economically acceptable amount of server resources. Moreover, the test application implementing the algorithm is presented together with a series of experimental results showing the algorithm quality depending on different parameters' sets.

1.3 Paper Organisation

The rest of this paper is organised as follows. Section 2 introduces the novel detection algorithm. Section 3 contains description of test environment (test application, experimental data sets) and shows the results of several tests concerning detection quality depending on six different sets of parameters. The final section discusses results and concludes the paper.

2 Detection Algorithm

2.1 Requirements and Problems

The ships' detection algorithm must detect moving ships that are present on a video stream. The algorithm is designed to work for stationary cameras and detect all kinds of ships, including leisure crafts. The algorithm should use only video streams from an existing monitoring system without any additional hardware or special configuration of cameras. The output of the algorithm are ships pictures cut out of frames in which ships were detected.

The task of selecting foreground objects in a scene is easy in indoor environment, but in outdoor environment it is much more difficult due to many factors that must be considered. It becomes easier when background and lightning are constant at the scene. This is the opposite situation for the one in which the developed detection algorithm should work.

The main problems related to detection of moving ships are [1]: the camera usually is not directed strictly at the water, several other moving objects are present in the frame (e.g., trees that moves in windy conditions), moving animals (e.g., birds) and people, different water state (waves), different camera angles or waterway crossroads, obstacles blocking the view, and different lightning conditions.

2.2 Algorithm Description

The algorithm uses the pixel-based approach. Mostly since it is designed to detect a wide range of ships and it is difficult to find the distinguishing features of each type of ship. The algorithm takes as an input 12 different parameters. A pseudocode for the algorithm is presented below. Several phases in processing a video frame can be distinguished. At first, a frame is pre-processed (lines 6–8), i.e., is resized, blurred and converted to grey scale (using all RGB channels or only using red channel). Next, the algorithm check weather to update the background model. Many background subtraction algorithms can be used. The model is not updated every frame, because in such case some slow-moving ships merge with the background.

Afterwards, the algorithm is looking for the contours on the mask and calculates bounding boxes for each contour. The filtering of objects that are not a ship is performed in two stages. In the first stage (lines 23–28), objects which lie entirely within negative regions are removed (the negative regions are pre-configured for each camera and mark the areas, i.e. land, a bridge, or sky). In the second stage (lines 29–33), the objects that are too small or have inadequate proportions are removed. Finally, the algorithm merges overlapping objects and cuts them out from the original frame. Then returns the resulting images.

Algorithm 1: Detect-Ships

```

Input: video_stream, detection_parameters = {alg_ext,
    res, blur_size, flag_red, n_init, n_learn,
    neg_regions, n_detect, min_width, min_height,
    max_ratio, min_area}
Output: array<bmp>
1 array<bmp> ships = init_empty_array();
2 frame_counter = 0;
3 fgbg = initiate_background_substractor(alg_ext);
4 foreach frame ff in video_stream do
5     frame_counter++;
6     f = resize_to(ff, res);
7     f = covert_to_gray(f, flag_red);
8     f = blur_frame(f, blur_size);
9     if ((frame_counter < n_init) or
10        (frame_counter mod n_learn != 0))
11     then
12         continue;
13     else
14         fgbg.update(f);
15     end
16     if (frame_counter mod n_detect != 0) then
17         mask = fgbg.get_mask();
18         contours = find_contours(mask);
19         array<br> tmp_ships = init_empty_array();

```

```

20     foreach contour c in contours do
21         br = get_bounding_rectangle(c);
22         flag = false;
23         foreach region r  $\in$  neg_regions do
24             if (br  $\subseteq$  r) then
25                 flag = true;
26                 break;
27             end
28         end
29         if ((flag == false) and
30             (br.width > min_width) and
31             (br.height > min_height) and
32             (br.ratio < max_ratio) and
33             (br.area > min_area))
34             then
35                 tmp_ships.add(br);
36             end
37         end
38         tmp_ships = merge_overlapping_br(tmp_ships);
39         foreach bounding_rectangle br in tmp_ships do
40             ship_bmp = get_frame_fragment(ff,res,br);
41             ships.add(ship_bmp);
42         end
43     end
44 end

```

Notation:

res	- resolution {1920x1080,720x540,960x540}
alg_ext	- a foreground extraction algorithm {FF, MOG, MOG2, GMG, CNT, KNN, GSOC, LSBP}
flag_red	- convert to gray using only Red channel {true/false}
n_init	- number of initialisation frames
n_learn	- a learning interval for alg_ext (number of frames)
n_detect	- a detecting interval (number of frames)
neg_regions	- a set of polygons that describe frame area not designated for detection
max_ratio	- maximum height to width (or width to height) ratio

3 Experimental Evaluation

The main purpose of the test was to find the optimal set of twelve parameters that will become default setting. Firstly, the performance of algorithms was tested [1] and, based on an expert opinion, values for nine parameters were set. Next, for the remaining three parameters two values have been selected, each with the greatest influence on the detection quality.

3.1 Test Environment

In order to test the algorithm, the test application was written in C# and is using Emgu CV version 3.4.3 (C# wrapper for OpenCV). The application (see Fig. 1) allows user to select different parameters and to see how they affect the detection quality. It is possible to watch detection performed by two methods simultaneously to allow for better comparison. Additionally, it is possible to store detection results into the files.

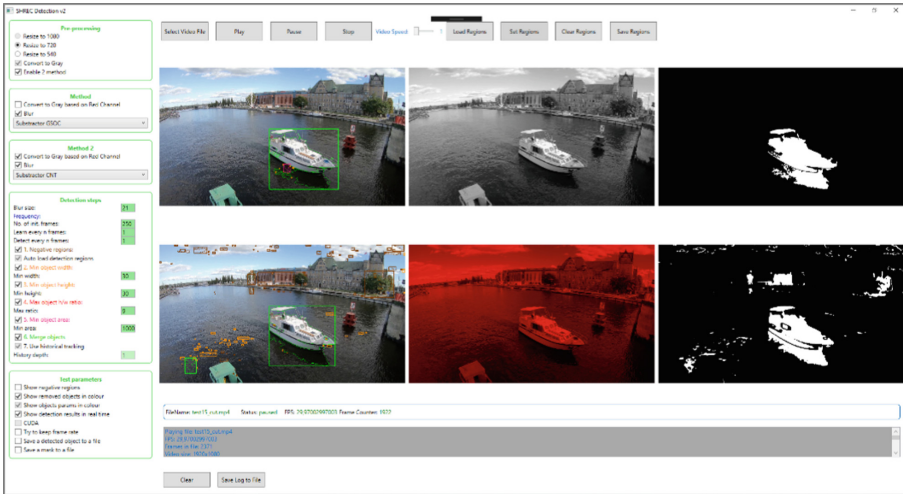


Fig. 1. A screenshot from the test application

The data set that contain more than 200 video samples was recorded on the waterways near Szczecin during the summer 2018. The 15 different video samples (Full High Definition 1920×1080 , 30 fps, bitrate: 20 Mb/s, AVC Baseline@L4, duration: between 30 s and 120 s) that show different type of ships from different angles, recorded in different places, were selected for the test. The Fig. 2 shows screenshots of a few samples. The following parameters was set to the constant values: $blur_size = \{21\}$, $flag_red = \{false\}$, $n_init = \{250\}$, $neg_regions = \{loaded\}$, $n_detect = \{30\}$, $min_width = \{30\}$, $min_height = \{30\}$, $max_ratio = \{9\}$, $min_area = \{1000\}$. The processing resolution and a background model update rate were tested with the following six sets of parameters:

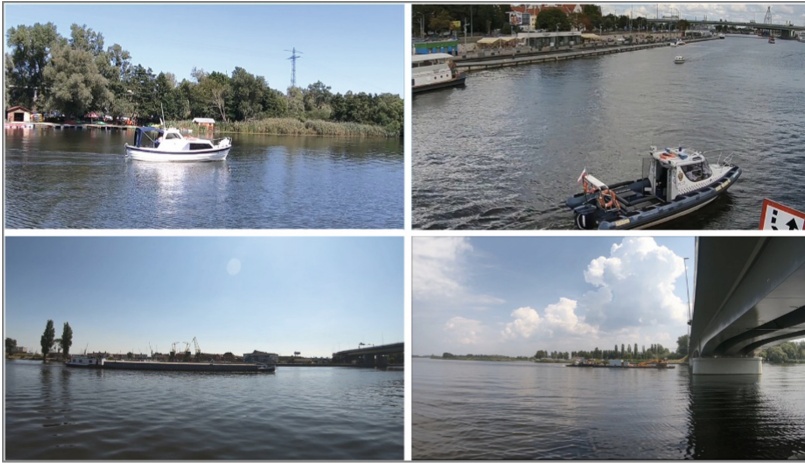


Fig. 2. Examples of video samples

1. $res = \{960 \times 540\}; n_learn = \{1\};$
2. $res = \{960 \times 540\}; n_learn = \{5\};$
3. $res = \{960 \times 540\}; n_learn = \{10\};$
4. $res = \{1280 \times 720\}; n_learn = \{1\};$
5. $res = \{1280 \times 720\}; n_learn = \{5\};$
6. $res = \{1280 \times 720\}; n_learn = \{10\}.$

A test computer (Core i7-8700 K, 32 GB RAM, SSD 1TB, NVIDIA Quadro P4000) was used in the tests. The tests were run using 6 sets of parameters described above for each video sample. Each test outputted a set of image files.

3.2 Results

The output images for each test were divided into the following categories:

1. correctly detected ships;
2. partially detected ships (an image contains a part of the ship);
3. artefacts (not ships) – anything else detected except water;
4. artefacts (not ships) – only water detected as object.

Additionally, the number of correctly detected different ships in each video sample was counted (i.e., ships that are present on at least one image). The samples from each category are presented on the Fig. 3.

The algorithm is designed to output possible ships' images that will be used further by a ships' classification algorithm. Therefore, the most important parameter is FNR (False Negative Rate) that shows how many ships were not detected at least once per passage in front of the camera. The FNR (see Fig. 4) is significantly lower when the background model is updated every 10 frames (three times per second). The FNR for

GSOC-10-720 is only 2% and for *GSOC-10-540* is 6%. The FNR for *GSOC-1-540* is 29% and for *GSOC-1-720* is 19%. Generally, higher resolution provides better detection results.

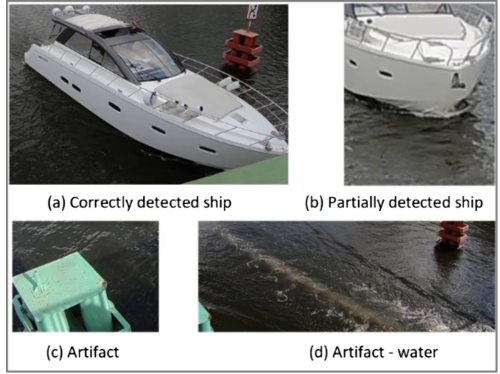


Fig. 3. Examples of different categories of output images

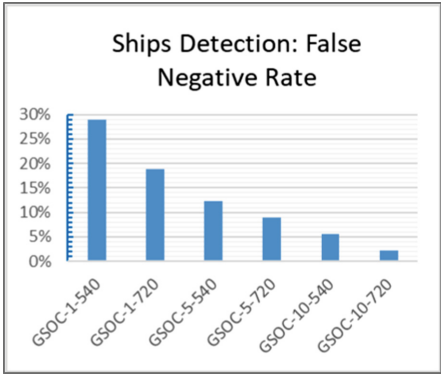


Fig. 4. Ships detection: False Negative Rate

In the following test, each video sample was tested using each set of parameters, which resulted in 90 tests. The results of each test were categorised into one of four categories mentioned above. Then, number of elements in each category was counted. The element was considered as the 'partial correct detection' result when it contains some distinguishable visible property showing that the element is indeed a ship. The ratios of each category broken down by different parameters' set are presented on the Fig. 5.

The best ratio of correctly detected ships to all detection results was achieved using *GSOC-5-540* (52%), *GSOC-5-540* (51%), and *GSOC-5-540* (51%). Further, counting together correct detection and partial detection the best results was achieved by *GSOC-*

5-540 (87%). The artefacts constitute 21% for *GSOC-10-720* (the worst case) and 13% for *GSOC-5-540* (the best case). However, in the test results around 2/3 of artefacts contain water only. Mainly, some waves caused by passing ships and sudden wind gusts. The percentage of artefacts (incorrect detection results excluding water) varies from 3% *GSOC-5-540* to 6% (*GSOC-1-540*, *GSOC-1-720*, and *GSOC-10-720*).

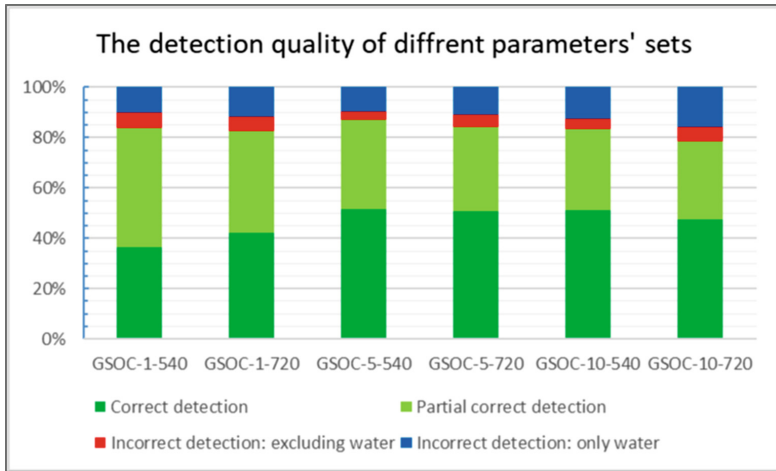


Fig. 5. The detection quality of different parameters' sets

4 Conclusion

The background model is updated not every frame, because in a such situation slow moving, long ships, for example barges, merge with a background. From the other hand, when the update rate is higher the background model has problems with capturing waves and merging them into the background. Therefore, the update rate around three times per second in our type of video streams is optimal. The higher resolution, i.e. HD 1280×720 , provides better FNR. This probably results from the fact that small or slow-moving ships, after downsizing the frame, are more easily merged with the background model. The FHD resolution (1920×1080) might provide better FNR, but calculations take too long to use it practice.

It is worth to mention that some partial detection results are inevitable from the fact that a ship is moving outside the camera view. The highest number of water artefacts result from less frequent updates of the background model. Possibly, the ratio of erroneous artefacts could be increased by finetuning the filtering using negative regions. To sum up, the best set of parameters is *GSOC-10-720* as it will be easier to further minimize the number of artefacts than improve the FNR.

Our algorithm differs from solutions mentioned in the first section with a different design philosophy, use of GSOC background subtraction algorithm, and using all RGB channels to create grayscale image instead of using only red channel. In our approach, we try to detect all objects that are moving on water and then eliminate non-ships by

usage of some logic but not image processing. Additionally, the algorithm must detect all moving ships, including small boats and kayaks, which is the main difference from existing solutions that mostly focus on only one vessel type.

The main limitation of the study is that the algorithm is designed to work with stationary FHD streams with good quality. It should work with 4 K and 720 p streams equally good, but it will require further testing. Other limitation is that algorithm works only in daytime without large atmospheric precipitation.

Future works include improving the algorithm by adding a subroutine that removes water artefacts. Additionally, the tracking algorithm will be integrated with the detection algorithm, so when consecutive frames show the same ship, they are given some ID.

Acknowledgement. This scientific research work was supported by National Centre for Research and Development (NCBR) of Poland under grant No. LIDER/17/0098/L-8/16/NCBR/2017.

References

1. Hyla, T., Wawrzyniak, N.: Automatic ship detection on inland waters: problems and a preliminary solution. In: Proceedings of ICONS 2019 The Fourteenth International Conference on Systems, Valencia, Spain, pp. 56–60. IARIA (2019)
2. Ferreira, J.C., Branquinho, J., Ferreira, P.C., Piedade, F.: Computer vision algorithms fishing vessel monitoring—identification of vessel plate number. In: De Paz, J.F., Julián, V., Villarrubia, G., Marreiros, G., Novais, P. (eds.) ISAMi 2017. AISC, vol. 615, pp. 9–17. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-61118-1_2
3. McConnell, R.K.: Method of and apparatus for pattern recognition. US Patent 4,567,610 (1986)
4. Hu, W.-C., Yang, C.-Y., Huang, D.-Y.: Robust real-time ship detection and tracking for visual surveillance of cage aquaculture. *J. Vis. Commun. Image Represent.* **22**(6), 543–556 (2011)
5. Szpak, Z.L., Tapamo, J.R.: Maritime surveillance: tracking ships inside a dynamic background using a fast level-set. *Expert Syst. Appl.* **38**(6), 6669–6680 (2011)
6. Kaido, N., Yamamoto, S., Hashimoto, T.: Examination of automatic detection and tracking of ships on camera image in marine environment. In: 2016 Techno-Ocean, pp. 58–63 (2016)
7. Kim, Y.J., Chung, Y.K., Lee, B.G.: Vessel tracking vision system using a combination of Kaiman filter, Bayesian classification, and adaptive tracking algorithm. In: 16th International Conference on Advanced Communication Technology, pp. 196–201 (2014)
8. da Silva Moreira, R., Ebecken, N.F.F., Alves, A.S., Livernet, F., Campillo-Navetti, A.: A survey on video detection and tracking of maritime vessels. *Int. J. Res. Rev. Appl. Sci.* **20**(1), 37–50 (2014)
9. Brutzer, S., Höferlin, B., Heidemann, G.: Evaluation of background subtraction techniques for video surveillance. In: CVPR 2011, Colorado Springs, CO, USA, pp. 1937–1944 (2011)
10. Benezeth, Y., Jodoin, P.-M., Emile, B., Laurent, H., Rosenberger, C.: Comparative study of background subtraction algorithms. *J. Electron. Imaging* **19**(3), 1–30 (2010)
11. Emgu CV Library Documentation version 3.4.3. <http://www.emgu.com/wiki/files/3.4.3/document/index.html>. Accessed 15 Apr 2019

12. KaewTraKulPong, P., Bowden, R.: An improved adaptive background mixture model for real-time tracking with shadow detection. In: Remagnino, P., Jones, G.A., Paragios, N., Regazzoni, C.S. (eds.) *Video-Based Surveillance Systems*. Springer, Boston (2002). https://doi.org/10.1007/978-1-4615-0913-4_11
13. Zivkovic, Z.: Improved adaptive gaussian mixture model for background subtraction. In: *Proceedings of the 17th International Conference on Pattern Recognition (ICPR 2004)*, vol. 2, pp. 28–31. IEEE Computer Society, Washington (2004)
14. Godbehere, A.B., Matsukawa, A., Goldberg, K.: Visual tracking of human visitors under variable-lighting conditions for a responsive audio art installation. In: *2012 American Control Conference (ACC)*, pp. 4305–4312 (2012)
15. Zeevi, S.: BackgroundSubtractorCNT Project. <https://github.com/sagi-z/Background-SubtractorCNT>. Accessed 15 Apr 2019
16. Zivkovic, Z., van der Heijden, F.: Efficient adaptive density estimation per image pixel for the task of background subtraction. *Pattern Recogn. Lett.* **27**(7), 773–780 (2006)
17. Guo, L., Xu, D., Qiang, Z.: Background subtraction using local SVD binary pattern. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition Workshops, Las Vegas, NV*, pp. 1159–1167 (2016)
18. Wawrzyniak, N., Stateczny, A.: Automatic watercraft recognition and identification on water areas covered by video monitoring as extension for sea and river traffic supervision systems. *Pol. Marit. Res.* **25**(s1), 5–13 (2018)
19. Wawrzyniak, N., Hyla, T.: Automatic ship identification approach for video surveillance systems. In: *Proceedings of ICONS 2019 The Fourteenth International Conference on Systems, Valencia, Spain*, pp. 65–68. IARIA (2019)