23^{rd} *Computer Vision Winter Workshop*
Zuzana Kúkelová and Júlia Škovierová (eds.)
Český Krumlov, Czech Republic, February 5–7, 2018

Depth Fingerprinting for Obstacle Tracking using 3D Point Cloud

Jon Muhovič, Rok Mandeljc, Janez Perš
Faculty of Electrical Engineering
University of Ljubljana
Tržaska 25, SI-1000 Ljubljana
jon.muhovic@fe.uni-lj.si

Borja Bovcon
Faculty of Computer and Information Science
University of Ljubljana
Večna pot 113, 1000 Ljubljana

Abstract. *We present a method for automatic detection and tracking of obstacles on water surface that uses solely the point cloud obtained from the surroundings of the unmanned surface vehicle (USV). For this purpose, we use a calibrated pair of stereo cameras, affixed to the mast at the front of the USV. Reliable obstacle tracking in outdoor environment is a difficult task, but unlike the monocular approaches, our framework offloads a large part of the problem onto the method that provides a point cloud. In absence of other visual features, our method introduces* depth fingerprint*, a histogram-like feature obtained from the point cloud of an object. The method has been evaluated on the yet unreleased MODD2 dataset and shows promising results, with the depth fingerprinting significantly outperforming tracking based solely on optimal assignment weighted by geometrical distance between object detections (Munkres algorithm). The proposed method is capable of running in real time on board of a small-sized USV.*

1. Introduction

As the ground and aerial autonomous robots become more common, robots that can autonomously navigate the water surface (unmanned surface vehicles, USV) have also become an interesting topic of research. As the environment for such robots is very dynamic, the GPS is usually not enough to ensure safe path planning. Accurate detection, classification and tracking of nearby obstacles is needed. Unique circumstances on the water surface (reflection, waves, mirroring) require an adaptation of standard procedures and the development of new methods, as the performance of computer vision in the realistic sea conditions is still insufficient for entirely autonomous operation [6]. The overall schematics of our system for detecting and tracking obstacles on water surface is shown in 1.

1.1. Problem statement

When introducing computer vision to the robotic vehicle for the purpose of obstacle detection and avoidance, there are two basic modalities of data that can be used (and combined):

- **Pixel data.** Monocular vision relies on a gray-scale or RGB image sequence from a single camera. The key assumption in this approach is that the obstacles somehow *visually differ from the background*, and that that they can be *segmented from the background* using only the pixel data, exploiting the visual differences. A clear example of this approach is the work on semantic image segmentation by Kristan et al. [6].
- **Depth data.** There are many techniques to obtain depth data in outdoor environment, some more robust than others. LIDAR sensors are the most robust method of obtaining depth data and widely used in experimental self-driving vehicles, however, the cost of sensors that are able to generate reasonably dense data (e.g. hundreds of horizontal lines at the time) is prohibitive for everyday use. Time-of-flight cameras are usually robust, but their range is usually too small. Finally, depth data can be obtained from the calibrated pair of relatively low-cost cameras, such as in [5], if there is enough texture in the scene to allow robust stereo matches between points in the left and the right camera.

Both approaches have their advantages and disadvantages. Simply said, monocular approaches can-

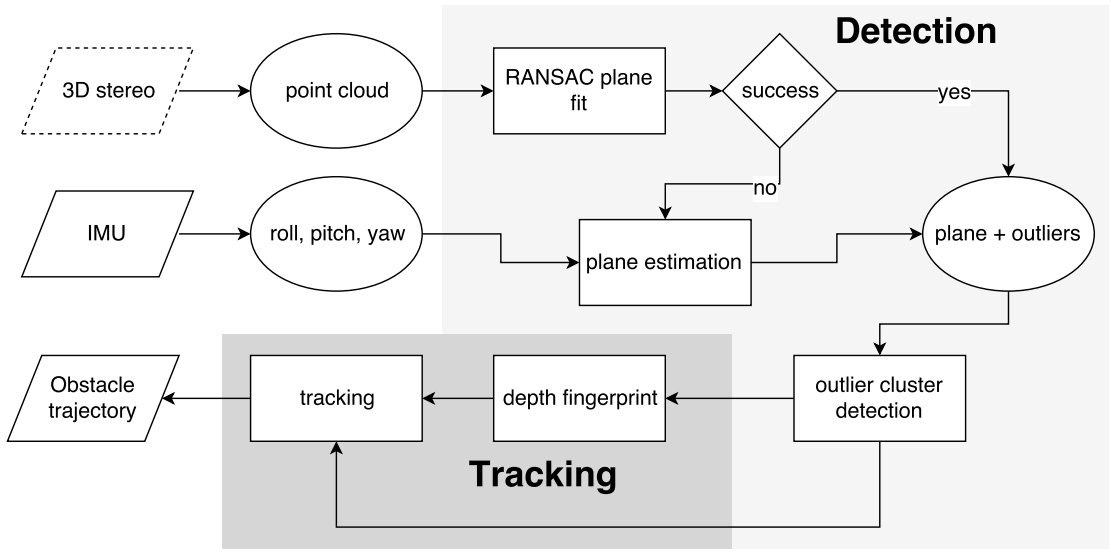

Figure 1. Processing pipeline. Detection part (plane pop-out) is described in Algorithm 1. Our second contribution, tracking by depth fingerprinting is described in Section3.3.

not distinguish between *an image of an obstacle* and *the obstacle*. However, this becomes less important in water-borne scenarios, since the visual features of ocean surface are far easier to model than (for example) urban driving environment. However, there are still many challenges that are especially pressing in coastal waters, as shown in [6]. Those are mainly related to variation on the water surface due to variable lighting, reflections and general visual cluttering in areas such as marinas. Stereo approaches are, in theory, extremely reliable – that is, if the depth data is reasonably error free. Obstacles can be detected, measured, and avoided. However, in our preliminary testing, we have established, that the closer we get to the shore, more problems are revealed by the stereo approaches as well, and these problems have not been sufficiently addressed by the related work.

The *overall aim* of our work is to get high quality obstacle detection both from the depth stereo data and the pixel data and then apply late-stage fusion to provide situational awareness for the USV. However, in this paper we limit ourselves on obtaining high quality obstacle detection from the depth data only, and try to be agnostic regarding the source of the depth data.

2. Related work

Although water surface vehicles receive far less attention than ground and aerial robots, several works have been published that deal with automatic obstacle detection on the water surface. Approaches proposed by different authors share several similarities, but based on the purpose of the method (following dynamic obstacles, path planning *etc*.) also have quite distinct emphases. This section will present several approaches.

Authors Larson *et al*. [8, 9] were one of the first to use the approaches used with ground robots with a waterborne platform. Their platform uses a NASA JPL stereo system to acquire depth information. The point cloud is projected into an obstacle grid. An additional functionality is the monocular approach that first detects the horizon and then estimates the distance to the obstacles under the horizon line by using the distance from the horizon to the obstacle. This is performed by using information from nautical charts.

Huntsberger *et al*. [5] present the Hammerhead system that uses two pairs of stereo cameras to attain a very large angle of view. The camera images are used to calculate a point cloud, then a plane representing the water surface is fitted into the cloud. The points above the water surface are then projected into a 2D map which is subsequently temporally and spatially filtered, the obstacles are classified and tracked. This system is by design the most similar to the one presented in this paper.

A group of researchers [16] presented their system for detection and tracking of obstacles on the water surface. Their unique approach is using saliency detection in Lab color space along with Harris corner detector and optical flow which produce regions of interest. The disparity for each region of interest is then calculated by using the disparity from the previous time step with normalized cross-correlation.

The same authors improve their approach in their next papers [15, 14] by using temporal information when estimating the depth of an obstacle and using RANSAC to calculate the water surface plane from a subsampled point cloud. The points above the plane are projected into two separate maps, the occupancy grid and the height grid. Both maps are later merged and hexahedrons (cubes) that determine the obstacles are calculated. The detected obstacles are projected into the 2D coordinate system of the left image to simplify the experimental evaluation.

An even further improvement was proposed in [13] by using tracking between neighboring frames. The detected obstacles were tracked in between neighboring frames to find consistent objects. The comparison was performed using the size, location and HSV histogram of the detections. The authors also proposed an image feature based particle filter to handle potential occlusions of tracked objects. This includes a transition model (for coordinate and scale change prediction) and an observation model (HSV and SIFT histograms). The particle filter takes over when the data association fails and stops if/when the obstacle is redetected.

Sinisterra *et al*. [11, 12] reversed the standard order of actions by first using HSV color segmentation to acquire regions of interest and then calculating their depth using a stereo system. An extended Kalman filter was then used to track the detected objects and estimate their velocity.

Kristan *et al*. [6] performed a semantic image segmentation using Markov random fields. Their visual model includes weak priors for segmenting the image into the sky, the coast and the sea. The segmentation is then iteratively improved. Regions that violate the visual model are presumed to represent obstacles on the water surface.

The paper from authors Cho *et al*. [2] used the FAST corner detector and the information about the horizon to detect obstacles. The detected points were clustered based on the Euclidean distance and the relative distance to the camera was calculated using the distance to the horizon. To attain a wider camera viewing angle they used a panning module.

Halterman *et al*. have in their paper [4] evaluated LIDAR as a method of detection obstacles in the vicinity of the boat. They used a LIDAR Velodyne HDL-64E that has 64 separate lasers that cover an angle of $360°$. Because laser sensors typically do not return on the water surface, they could be used to detect and classify obstacles up to $100\,\mathrm{m}$ away. An issue that arises is the fact that laser pulses interfere with GPS receivers and that LIDAR system themselves are susceptible to noise induced by radar pulses. Lately, approaches have emerged that use convolutional neural networks to detect and classify obstacles in images [1, 17].

An approach to visual tracking was proposed by Fefilatyev *et al*. in [3]. Their method performs segmentation on color images, then tracks the output regions with a Multiple-Hypothesis Tracking framework. The linear Kalman filter is used to manage each hypothesis and predict the motion of the tracked object.

In contrast to most of the approaches described above, we rely solely on 3D point cloud data, and make no assumption about its source. As a consequence, our approach can be used directly on the state-of-the-art 3D sensors that do not provide visual information (e.g. Velodyne LIDAR sensors that are commonly used in prototype self-driving vehicles) and much cheaper passive or active 3D stereo systems.

3. Our approach

Obstacles that could pose problems for our USV robot are objects on the surface of the water such as buoys, boats and swimmers. An accurate estimate of the distance to the obstacle is required for safe navigation. For the analysis of the surroundings of the boat we used a stereo camera system that generates a point cloud, but we do not assume anywhere in our approach that the the source of the point cloud is a stereo system. Alongside the point cloud we also used information from the onboard inertial sensor (IMU), but no visual information (RGB data from cameras) were used directly in any way except for generation of the point cloud. Schematics of our approach are shown in Figure 1.

3.1. Hardware and image acquisition

The robot platform (boat) has two cameras of the type Vrmagic VRmS-14, that are affixed to the main mast, approximately 70cm above the water surface. The distance between the cameras is 20cm and their resolution 1280x960 pixels. The cameras use 3.5mm lenses. A fast bus connects the cameras with the control unit that ensures the pixel synchronization of the cameras. USB 2.0 is used to transfer the image data to the CPU (Intel i7-4770), which allows the capture

of 10 images per second. These parameters allow us to acquire a high-quality point cloud up to 20m away. The cameras are calibrated using the calibration tool included in OpenCV library and a target pattern with asymmetrical circles. A target of size 1m is required to calibrate the stereo system geometry. Our platform is additionally equipped with a high precision GPS receiver that is able to accurately estimate the boat position down to 10cm. An IMU unit is also used to provide the roll, pitch and yaw information of the boat.

3.2. Obstacle detection

We independently devised and implemented the straightforward "plane pop out" method for obstacle detection, which bears some similarity to [5]. We model the water surface as a plane, and the outliers represent possible obstacles. The approach is presented in pseudo-code, shown in Algorithm 1.

Algorithm 1 Obstacle detection using a point cloud

procedure OBSTACLES($ImagePair, IMU$)
  $Cloud \leftarrow Stereo$(ImagePair) ▷ OpenCV
  $SeaSurface \leftarrow PlaneFit$(Cloud) ▷ RANSAC
  if $SeaSurface \neq OK$ **then**
    $SeaSurface \leftarrow IMUFit$(IMU)
  end if
  $AboveSurface \leftarrow$Cloud($Z > \epsilon$)
  $PointDensity \leftarrow PDEstimation$(AboveSurface)
  $3Dblobs \leftarrow Threshold$(PointDensity)
  $Objects \leftarrow FloodFill$(3Dblobs)
  $Obstacles \leftarrow MeanMinMax$(Objects)
  return $Obstacles$
end procedure
function PDESTIMATION($Cloud$)
  $3DDensity \leftarrow 3DConvolution$(Cloud, box)
  return $3DDensity$
end function

Parts of the algorithm are implemented in the following manner: Depth calculation (stereo) is done by the OpenCV library by using the block matching method which produces a point cloud. The block matching method is not the most sophisticated stereo method, but it can be tuned to outperform other methods in the case of use on a USV and thus gives better results for highly differing visual conditions encountered on the water surface. Other authors have also preferred block matching to other stereo methods [14, 10]. A critical part of our algorithm is estimating parameters of the water surface. In our case, because we are working in 3D space, we represent the water and the air as two separate volumes, divided by a plane.

In this case, the water volume always lies below the air. The **PlaneFit** method calculates the position of the water surface by fitting a plane into the point cloud using the RANSAC method. This requires 3 points to be selected in each iteration, with which we can define a 3D plane. We extend the RANSAC by incorporating the specific problem knowledge. If the roll and pitch angles of the plane defined by the randomly selected points differ too much from the plane defined by the IMU data, the candidate plane is discarded. Thus, only fairly reasonable planes are evaluated in the RANSAC method. We assume that the largest fraction of the point cloud lies on the water surface. That assumption holds well when the boat is far enough from the shore or the pier. In the case of a large number of outliers, RANSAC cannot produce a solution in a reasonable timespan, and the plane define by the IMU data is used instead. The plane equation is used to discard all the points that lie below the surface or on it ($Z \leq \epsilon$). A useful estimate for ϵ is $20\,\mathrm{cm}$ where the value is bounded by the size of the waves that we still tolerate and by the smallest obstacle size that we still wish to detect. The rest of the points in the cloud is transformed into a discrete 3D matrix of point density by filtering it with a 3D filter (**PDEstimation**). The filter size depends on the minimum size of the obstacle we want to detect. By using thresholding (**Threshold**) and minimal required point density the discretized space we acquire voxels that have the value 1 where we have detected an obstacle and 0 otherwise. The required point density varies with the distance to the camera, since the density of the point cloud diminished with the distance. Using the (**FloodFill**) algorithm the points are merged into objects (obstacles) and their parameters are estimated (**MeanMinMax**) - *i.e.* the obstacle center and its span in all dimensions. It is obvious that the estimate is unreliable in the direction parallel to the camera optical axis. But for the purpose of avoiding obstacles the distance between the boat and the closest obstacle is crucial.

3.3. Tracking

The point cloud recovered from the stereo system is not completely dense and contains noise. Regions that are connected in the real world might be reported as group of smaller obstacles. On many occasions, false stereo matches occur due to repeated vertical structures (such as multiple boat masts in the marina). Finally, the surface of the water may cause false positives as well. These are the pressing problem, since they disrupt the tracking of the real obstacles. One such example is shown in Figure 2. To filter out these inconsistencies, a temporal component must be included, to track only objects that are appearing consistently in multiple sequential frames.

The data association algorithm runs in four stages:

1. New detections are assigned to previously tracked objects.
2. Remaining new detections are assigned to potential objects and their counter is incremented.
3. New potential objects are created from remaining detections.
4. Lost counter is incremented for unassigned tracked objects, unmatched potential objects are removed.

When an obstacle is first detected, it is added to the list of potential obstacles. If it is continuously redetected in next N frames, it is added to the tracked obstacles list. If not, it is removed from the potential list. To prevent tracked objects from being prematurely discarded, a counter is kept and incremented if a tracked object is not detected in the current frame. If this counter reaches a certain threshold, the object is assumed to be lost and removed from the list.

To match the detections between sequential frames, we need to connect ones representing the same object as well as recognize non-matched objects and newly seen ones. This was performed by using the Hungarian method (also known as Munkres assignment algorithm [7]). The method requires a matrix of distances between two sets of objects on which we are performing the assignment. An optimal assignment of objects from one list to objects from the second list is then calculated (*i.e.* the assignment with the lowest cost). If the sizes of both sets do not match, some objects can be left unassigned.

In general, we strive to minimize function of the form

$$S(a,b) = \alpha\, distg(a_c, b_c) + (1-\alpha)\, H(a_h, b_h) \quad (1)$$

where a and b are detected obstacles and a_c, b_c and a_h, b_h are their positions (centers), while a_h and b_h are their *depth fingerprints*, respectively. $distg$ denotes the geometric distance, e.g. Euclidean, which was used in our case, and H denotes the histogram-type distance, for example Hellinger distance, which was used in our case to compare depth fingerprints:

$$H(p,q) = \frac{1}{\sqrt{2}} \sqrt{\sum_{i=1}^{k} (\sqrt{p_i} - \sqrt{q_i})^2} \quad (2)$$

Eq. (1) is a general form, which includes both geometrical (spatial) constraints (e.g. object in the next frame should be close to its previously detected position) and quasi-visual similarity constraint (distance between depth fingerprints). By setting α to 1, purely spatial constraints are taken into the account, and the result is classic Hungarian-type assignment.

3.4. Depth fingerprint

The *depth fingerprint* is essentially a histogram-like feature, which was devised to encapsulate as much information about the object as possible, while being reliant only on the 3D point cloud, without using RGB image data. The histogram for each 3D obstacle was calculated from the depth value (Z) of 3D points that comprise the obstacle, as seen from the direction of the USV. To acquire only the relative depth, the mean depth value was subtracted before calculating the histogram. This calculation is very fast, and can be implemented even on low-end computational devices, provided that the 3D point cloud is readily available. The histogram format also allows us to compare detections of different sizes. As the features are only used to compare detected objects in consecutive frames, long-term stability is not a requirement.

The same holds for scale invariance, since the size is not expected to vary by much between neighboring frames – even more, the objects with large scale variation in such short time interval should not be matched – as it is very likely the case of two similarly shaped objects being observed.

Finally, the bin width was determined heuristically by evaluating all dataset sequences at different histogram sizes.

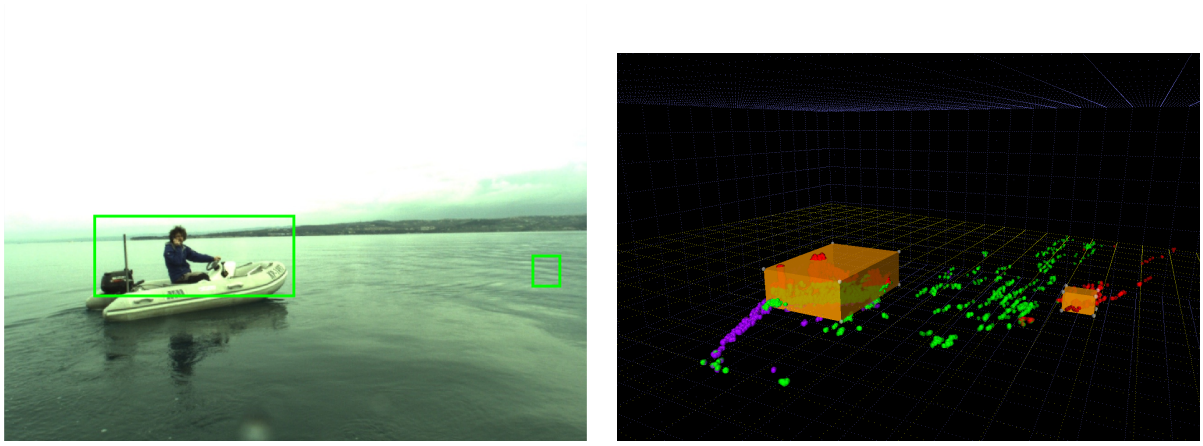

Figure 2. Illustration of imperfections in the point cloud that cause false positives. Left: one image from the stereo pair. Right: 3D representation of the point cloud with one true detection (inflatable boat) and one false detection (on the right side of the image). Reflection on the water surface resulted in a false match and a group of 3D points appeared above the surface, prominently enough to be detected as a potential obstacle.

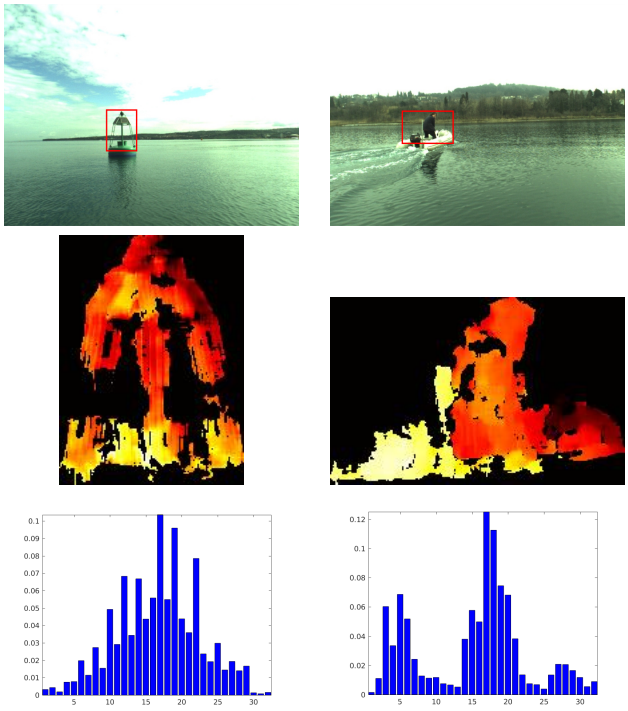

Figure 3. Illustration of depth fingerprinting. First row: one of the original images from the stereo pair. Left column: buoy at the entrance to the Port of Koper, right column: inflatable boat with the boat operator visible. Second row: closeup of the cropped point cloud of the detected obstacles. Reddish hues represent distant points, while the yellow ones represent the closer ones. Third row: depth fingerprints - the relative object depth histograms.

4. Evaluation

The evaluation was performed in the coordinate system of the images, which allows comparison with methods that do not use a calibrated system of cameras. The vertices of the cuboids that represent obstacles are projected back into the image and represented by a bounding box. Overlapping and too small boxes were filtered out. Annotations of obstacles further away than the range of our system were ignored in the evaluation, as they cannot be detected using any algorithm (the point cloud data is unavailable past certain distance from the USV). For the evaluation we followed a protocol similar to the one described in [6]. We report the following metrics that illustrate the performance of our approach: the average number of false positives per sequence frame (aFP), the F-score, and the number of false negatives per sequence.

4.1. Parameters

The parameters for obstacle detection were set to detect obstacles no closer than $1\,\mathrm{m}$ and no smaller than $20\,\mathrm{cm}$. When calculating the assignment cost, we ignored pairs of detections more than $1\,\mathrm{m}$ apart, as well as those whose histograms scored more than 0.2 similarity (Hellinger distance defines lower values as more similar). The weight α used to combine both distances was empirically determined to work best when set to a value of 0.5, thus equally combining both factors. The buffer size for promoting potential objects to tracked objects and the number of frames we wait before declaring a tracked object lost also needed to be defined. For the purpose of our experiments, both values were set to 5 (*i.e.* a detection has to be detected five times before we start tracking it and a tracked object can be unassigned for

a maximum of five frames before it is removed). A minimum overlap between the ground truth and our reported detection also had to be defined. We used a standard value of 0.5. The overlap was calculated as the intersection over union.

4.2. Dataset

For the evaluation of our system, a collection of 28 sequences was recorded on our platform (MODD2 - Marine Obstacle Detection Dataset 2). The sequences include the images from both cameras of the stereo system. Obstacles on the water surface and points that define the horizon have been manually annotated for all images. A larger part of the proposed algorithm was developed before the establishment of the dataset so it represents an unrelated test set. We also expect the dataset to be released for public use in the near future. For the evaluation of our tracking method we split the dataset into two subsets, based on the presence of detectable objects (*i.e.* annotations of objects within the range of our stereo system - $20\,\mathrm{m}$). Table 1 contains the mean result over all sequences in a subset as well as the standard deviation. For the sequences with no detectable obstacles we only report the average of false positives as there cannot be any true or false positives. For the subset containing obstacles we report the average of false positives as well as the number of false negatives and the F-score.

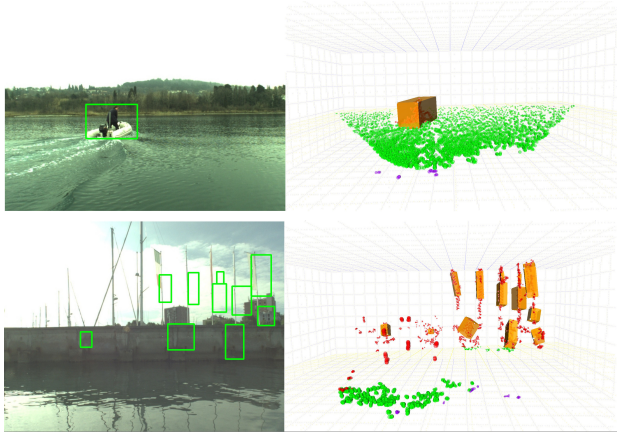

Figure 4. Example of properly working obstacle tracking (top), and the failure in the presence of large objects (e.g. piers, large ships, etc. The presented approach is unable to properly detect the obstacles that are larger than a field of view. Instead, they become fragmented.

4.3. Results

The plane estimation using the point cloud is dependent on a large enough number of points lying on the water surface. This method is thus not particularly suitable for cluttered areas such as harbors or coastal regions. Table 1 shows the results for both subsets of out dataset. We can notice a large reduction in the average number of false positives when using Munkres assignment with only geometrical constraints (cost based only on Euclidean distance). An even greater improvement can be seen when using the cost augmented with our depth fingerprints. The average F-score over all sequences confirms the improvement. It has to be noted that the average number of false negatives (*i.e.* a ground truth obstacle that was not detected/tracked) increases when using our method. This is due to the delay created by the tracker while waiting for the confirmation of a potential region. Increase when introducing depth fingerprint is relatively small.

5. Conclusion

We presented a method for detecting obstacles on the water surface and an approach that allows tracking of obstacles that are consistently detected. We have also prepared a dataset of 28 fully annotated sequences that can be used for testing similar systems. The sequences are diverse regarding weather and lighting conditions and include seaborne objects such as inflatable boats, buoys, sailboats, *etc*. We showed that our tracking approach significantly improves the performance of the detection system by reducing the amount of false positives while not affecting the number of false negatives in the meaningful way. In further work a problem of detecting large objects (e.g. coast and piers) will be addressed, as currently these obstacles violate our assumption of the water surface as the most dominant part of the point cloud. This problem is shown in Figure 4 and illustrates the limits of the presented approach. Additional visual information could also be used for an even more robust obstacle tracking, however, it is our intent to delay fusion with visual data to as late stage as possible.

Acknowledgements

This work has been supported by the Slovenian Research Agency grants J2-8175 and P2-0095. Visual recordings and other sensor data have been acquired onboard of the USV owned and operated by Harpha Sea, d.o.o, Koper.

Table 1. Results of the testing on the MODD2 dataset.

	with obstacles			without obstacles
	a-FP	F-score	FN	a-FP
detections	0.6549 ± 0.7195	0.4695 ± 0.2686	44.2353 ± 65.2816	0.3146 ± 0.4835
Munkres algorithm ($\alpha = 1$)	0.3190 ± 0.4077	0.5490 ± 0.2818	48.7059 ± 70.2814	0.1488 ± 0.3301
Munkres+depth fingerprint	$\mathbf{0.1049 \pm 0.1493}$	$\mathbf{0.6159 \pm 0.2734}$	50.8824 ± 67.5582	$\mathbf{0.0295 \pm 0.0792}$

References

[1] F. Bousetouane and B. Morris. Fast cnn surveillance pipeline for fine-grained vessel classification and detection in maritime scenarios. In *Advanced Video and Signal Based Surveillance (AVSS 2016)*, pages 242–248. IEEE, 2016. 3

[2] Y. Cho, J. Park, M. Kang, and J. Kim. Autonomous detection and tracking of a surface ship using onboard monocular vision. *2015 12th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI)*, pages 26–31, 2015. 3

[3] S. Fefilatyev, D. Goldgof, M. Shreve, and C. Lembke. Detection and tracking of ships in open sea with rapidly moving buoy-mounted camera system. *Ocean Engineering*, 54:1–12, 2012. 3

[4] R. Halterman and M. Bruch. Velodyne hdl-64e lidar for unmanned surface vehicle obstacle detection. In *SPIE Defense, Security, and Sensing*, pages 76920D–76920D. International Society for Optics and Photonics, 2010. 3

[5] T. Huntsberger, H. Aghazarian, A. Howard, and D. C. Trotz. Stereo vision–based navigation for autonomous surface vessels. *Journal of Field Robotics*, 28(1):3–18, 2011. 1, 2, 4

[6] M. Kristan, V. S. Kenk, S. Kovačič, and J. Perš. Fast image-based obstacle detection from unmanned surface vehicles. *IEEE transactions on cybernetics*, 46(3):641–654, 2016. 1, 2, 3, 6

[7] H. W. Kuhn. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2(1):83–97, 1955. 5

[8] J. Larson, M. Bruch, and J. Ebken. Autonomous navigation and obstacle avoidance for unmanned surface vehicles. In *Defense and security symposium*, pages 623007–623007. International Society for Optics and Photonics, 2006. 2

[9] J. Larson, M. Bruch, R. Halterman, J. Rogers, and R. Webster. Advances in autonomous obstacle avoidance for unmanned surface vehicles. Technical report, DTIC Document, 2007. 2

[10] B.-S. Shin, X. Mou, W. Mou, and H. Wang. Vision-based navigation of an unmanned surface vehicle with object detection and tracking abilities. *Machine Vision and Applications*, pages 1–18, 2017. 4

[11] A. J. Sinisterra, M. R. Dhanak, and K. von Ellenrieder. Stereo vision-based target tracking system for an usv. In *Oceans-St. John's, 2014*, pages 1–7. IEEE, 2014. 3

[12] A. J. Sinisterra, M. R. Dhanak, and K. Von Ellenrieder. Stereovision-based target tracking system for usv operations. *Ocean Engineering*, 133:197–214, 2017. 3

[13] H. Wang, X. Mou, W. Mou, S. Yuan, S. Ulun, S. Yang, and B.-S. Shin. Vision based long range object detection and tracking for unmanned surface vehicle. In *Cybernetics and Intelligent Systems (CIS) and IEEE Conference on Robotics, Automation and Mechatronics (RAM), 2015 IEEE 7th International Conference on*, pages 101–105. IEEE, 2015. 3

[14] H. Wang and Z. Wei. Stereovision based obstacle detection system for unmanned surface vehicle. In *Robotics and Biomimetics (ROBIO), 2013 IEEE International Conference on*, pages 917–921. IEEE, 2013. 3, 4

[15] H. Wang, Z. Wei, C. S. Ow, K. T. Ho, B. Feng, and J. Huang. Improvement in real-time obstacle detection system for usv. In *Control Automation Robotics & Vision (ICARCV), 2012 12th International Conference on*, pages 1317–1322. IEEE, 2012. 3

[16] H. Wang, Z. Wei, S. Wang, C. S. Ow, K. T. Ho, and B. Feng. A vision-based obstacle detection system for unmanned surface vehicle. In *Robotics, Automation and Mechatronics (RAM), 2011 IEEE Conference on*, pages 364–369. IEEE, 2011. 2

[17] J. Yang, Y. Xiao, Z. Fang, N. Zhang, L. Wang, and T. Li. An object detection and tracking system for unmanned surface vehicles. In *Target and Background Signatures III*, volume 10432, page 104320R. International Society for Optics and Photonics, 2017. 3