Vision-based Object Tracking in Marine Environments using Features from Neural Network Detections

Schöller, Frederik Emil Thorsson; Blanke, Mogens; Plenge-Feidenhans'l, Martin Krarup; Nalpantidis, Lazaros

Published in:
IFAC-PapersOnLine

Link to article, DOI:
10.1016/j.ifacol.2020.12.1455

Publication date:
2021

Document Version
Publisher's PDF, also known as Version of record

Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnline CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 53-2 (2020) 14517–14523

Vision-based Object Tracking in Marine Environments using Features from Neural Network Detections

F. E. T. Schöller * **M. Blanke** * **M. K. Plenge-Feidenhans'l** * **L. Nalpantidis** *

* *Automation and Control Group, Department of Electrical Engineering, Technical University of Denmark, 2800 Lyngby, Denmark, (e-mail: fets,mb,mkpl,lanalpa@elektro.dtu.dk).*

Abstract: Autonomous decision support is desired to enable navigation with a temporally unattended bridge or to have the vessel navigated remotely. In order to have safe navigation, it is crucial to correctly interpret the current situation given any scenario. Proper perception of the surrounding environment is essential for good situational awareness. This paper suggests a method for tracking objects that have been detected by a neural network. The method utilises features that have been computed during the detection step, thereby ensuring good features that are representative for the given objects while saving the time it would take to compute new features. The suggested method is evaluated on data acquired in Danish near-coastal waters. Evaluation shows that the tracking method is able to track the detections well with few switches of object identity. The method is shown to outperform a similar tracking algorithm, while keeping the speed needed for real-time applications.

Keywords: Autonomous Marine Vessels, Navigation, Computer Vision, Object Tracking

1. INTRODUCTION

Autonomous marine transport has gained significant attention in recent years. Different levels of autonomy in ships can provide services ranging from on-board decision support to ships navigated by a remotely located navigator or by an algorithm. The cornerstone of such services is situation awareness, which has three main elements, perception, understanding of the environment and anticipation of the development. Evidence of the crucial role of situation awareness for autonomous shipping is reflected by the multiple research resources attributed to the topic internationally (Blanke et al., 2019), (Porathe et al., 2014). When perception and understanding is supposed to be implemented as computer algorithms, ability to reliably track individual objects is crucial in order gain insight in other objects' possible future behaviour.

Tracking can be achieved by means of detection of presence using radar, by received automatic identification system (AIS) messages from larger vessels, or by tracking objects using cameras and other electro-optical sensors. Only some fleet vessels are obliged to send out AIS messages, and several categories of leisure crafts do not have radar reflectors. This paper therefore addresses the problem of using solely visual input to track objects at sea, an illustration of which is shown in Figure 1. We argue that the latest advancements in neural networks (NN) can provide both the means for reliable object detection, and visual appearance descriptions that make robust tracking possible. To this end, this paper suggests a method for tracking based on object features that have already been found by the neural network as part of the object detection. This paper shows

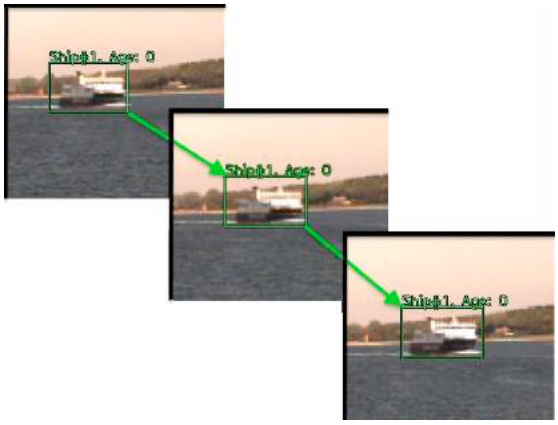

Fig. 1. Vision-based tracking in marine environment.

that, by reusing features, our approach ensures robust and discriminative representations of objects. This method will save computation time by avoiding extraction of new features.

The suggested approach is meant to be deployed in marine environments, thus this work features a number of appealing characteristics. First, it is implemented to run in real-time and thus with high execution speed (*FPS*), which is mainly due to the dual use of extracted features. In addition, we only need to train a single NN, which makes training and deployment of our approach easy and fast. The suggested method is evaluated on data that were acquired in Danish near-coastal waters and the paper shows the method's efficacy in such environments. The evaluation results obtained show that our approach is able

10.1016/j.ifacol.2020.12.1455

to detect relevant objects and track them over long periods of time with few switches of object identity. Finally, in some cases where the tracking of an object is lost for any given reason, our method is shown able to resume tracking of lost target and reassign the appropriate object ID. The contribution of this paper is hence twofold: (i) we propose a real-time tracking algorithm for marine environments and (ii) we show that the features extracted for object detection can be reused also for tracking, thus removing the need for a second feature extraction step.

Table 1. Abbreviations and Acronyms

acronym	explanation
NN	Neural Network
DL	Deep Learning
FPS	Frames per Second
IoU	Intersection over Union
MOTP	Multiple Object Tracking Precision
MOTA	Multiple Object Tracking Accuracy

2. PROBLEM DEFINITION

Our purpose is to track multiple objects from a sequence of images, using detections provided by a NN. The problem is as follows:

Given: A list of predicted objects with the attributes:

- Pixel coordinates for upper left and lower right corner of the bounding rectangle surrounding the object (bounding box)
- A feature vector that describes the appearance of the contents of the bounding box

Derive: An algorithm that can track each of the objects contained in the bounding boxes over consecutive frames.

2.1 Problem breakdown

From corner coordinates of the bounding rectangle for object i, we calculate its centre as $[x_c^i, y_c^i]$, the ratio r^i between height and width, the aspect ratio, and the area of the rectangle, s^i. With velocity vector of the centre, $[\dot{x}_c^i, \dot{y}_c^i]$, and rate of change of area, $\dot{s}^i$, prediction of bounding box motion in picture coordinates is governed by a state space equation. Using $\mathbf{x} = [x_c^i, y_c^i, s^i, r^i, \dot{x}_c^i, \dot{y}_c^i, \dot{s}^i]^T$. The following predictor assumes unknown centre velocity and bounding box area. With $k = 1, ...$ denoting discrete time (frame number being processed) and t_k the time stamp, the predictor for motion of the bounding box for object i is,

$$\begin{aligned} \hat{\mathbf{x}}_{k+1}^i &= \mathbf{A}_d(t_{k+1} - t_k)\hat{\mathbf{x}}_k^i + \mathbf{K}_k^i \left(\mathbf{y}_k^i - \hat{\mathbf{y}}_k^i\right) \\ \hat{\mathbf{y}}_k^i &= \mathbf{C}_d \hat{\mathbf{x}}_k^i \end{aligned} \tag{1}$$

Where the linear state space model $\{\mathbf{A}_d, \mathbf{C}_d\}$ is the discrete prediction model, mapping observations $\mathbf{y}_k^i = [x_c^i, y_c^i, s^i, r^i]_k^T$ to the states $\mathbf{x}_k^i$ and $\mathbf{K}_k^i$ being the predictor gain matrix. The predictor gain is calculated using a Kalman filter design (Lappe, 2015) with measurement covariance $\mathbf{R}_d$ and process covariance $\mathbf{Q}_d$ chosen to give adequate tracking performance. With $\mathbf{y}_k^i$ being calculated for all bounding boxes of objects estimated in image frame k, the predictor in Equation 1 is used to track development in positions of detected objects in an image. The crucial point is that i shall refer to the same object in consecutive frames.

2.2 Proposed solution

Solving this allocation puzzle would answer the problem formulated above. The paper shows how the combined allocation and prediction challenge is resolved in a sequence using four steps: in the first step, objects are detected using a NN; in the second, the expected location of all tracked objects is predicted for the new image; in the third, the features derived during object detection are used together with the predicted location of the tracked object to determine which objects correspond to detections in the new frame; in the fourth step, the bounding boxes found by the NN are used as input to the predictor in Equation 1. The overall structure of the proposed tracking algorithm is illustrated by the overall flowchart in Figure 2.

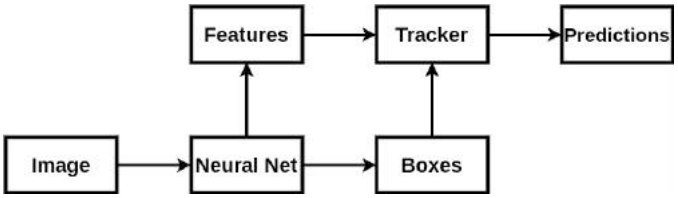

Fig. 2. Block diagram showing the structure of the tracking algorithm

3. RELATED WORK

Algorithms based on Deep Learning (DL) have proven to be very effective tools for visual object detection (Chatfield et al., 2014), as also attested by the vast plethora of relevant literature. The strength of DL algorithms originates from their inherent ability to learn and extract robust and discriminative object features automatically (Dara and Tumma, 2018), as opposed to human-crafted feature extractors. The use of NN-generated features can provide robust object detection results, even in challenging environmental conditions, such as in marine environments, which can be subsequently used for object tracking.

Object tracking for real applications often requires image data processing. A real-time tracking implementation has been proposed in the work of Bewley et al. (2016) that uses image streams as input to the Faster R-CNN (Ren et al., 2017) object detection network for real-time target tracking. The proposed implementation is able to track the bounding boxes of multiple objects in real-time using a simple combination of a Kalman Filter and the Hungarian algorithm with a simple association metric for inter-frame data association. The paper shows that the dominant factor influencing the tracking results is the accuracy and robustness of object detection.

However, a weakness of the work of Bewley et al. (2016) is that it results in a significant number of ID-switches during multiple object tracking, as object appearance is not included. To accommodate this, the work in Wojke et al. (2017) extended the previous implementation to include a new metric for inter-frame association, which includes appearance cues. This implementation results in

fewer ID-switches, while retaining its simplicity and real-time operation. In principle, this work is similar to our approach, but Wojke et al. (2017) resizes the detections to a fixed size and applies a second NN to create a feature vector describing the objects. However, in a marine environment, there is an abundance of very small objects due to the large distances found at sea. Resizing small objects for feature extraction might give rise to problems, as local features become very sparse. Zou et al. (2019) use a similar approach, where the appearance of two detections is compared using a siamese network that decides whether two detections are of the same object. while overcoming the resizing problem, this approach will also affect computational speed as a separate network is used to extract features once more. The method presented in this paper reuses the features extracted during detection; this is an advantage, as the features for a given object must be describing for the detection to occur. Thus, the overall implementation is simpler and more efficient.

The algorithm of Henriques et al. (2015) is among the state of the art, real-time performing tracking algorithms. This algorithm relies on human-crafted features and focuses on detection of objects in consecutive frames. In contrast, the algorithm proposed in this paper uses a NN to automatically learn relevant features for each case. Furthermore, the work of Henriques et al. (2015) requires an initial bounding box indicating the object to be tracked, while our approach employs automatic detection and then tracking of objects detected.

Tracking multiple objects in a tracking-by-detection manner, as introduced by Wojke et al. (2017), was recently extended and adapted for use in various specific applications. Vehicle tracking and re-identification was addressed in the works of Tang et al. (2019) and Hou et al. (2019), where an additional low confidence track filtering step was incorporated. Furthermore, tracking of objects from drone-mounted cameras was the focus of Kapania et al. (2020) and Jadhav et al. (2020), providing reliable tracking solutions for specific application scenarios.

The analysis of related work indicates a literature gap with respect to efficient and reliable visual tracking implementations in challenging environmental conditions, in marine environments. All the above mentioned algorithms focus on very different application or test scenarios that do not suffer from problematic environment conditions to the same degree as the vessel navigation scenario considered in this paper.

4. METHOD

As described in the problem statement in Sec. 2, the algorithm proposed in this paper is used to perform tracking of objects detected by a NN. Following the findings in (Schöller et al., 2019), an appropriate neural network for object detection in a marine environment is *RetinaNet* (Lin et al., 2017). In short, *RetinaNet* works by extracting features from an image and shaping these into four so called feature maps, each containing features of different scales, by using a convolutional neural network in conjunction with a bounding box regression network, and a classification network. This means that large, more global, features will be present in one feature map, while the more local features of smaller objects will be present in another. Each feature map is divided into a set of areas denoted *anchors*. The anchors are sent to the classification and regression network where the content of the anchor is classified. If the classification network determines that an object is present within the anchor area, a detection is made. The classification network classifies the content of the anchor box into a set of labels, e.g. ship or buoy, while the regression network optimises the fit of the anchor box to the contained object. The output of *RetinaNet* is therefore the coordinates of the bounding boxes containing the found objects together with their respective labels.

In order to have an algorithm providing situation awareness of the motion of surrounding objects, these need to be tracked and identified correctly from frame to frame after the objects in each frame are identified. The prediction step has to be incorporated such that the tracking algorithm is able to estimate the position of an object in case the NN does not find a previously tracked object due to e.g. occlusion.

The first step of the tracking algorithm is to match the detections in a newly acquired image with those already tracked from previous frames. A cost function for objects i and j is defined as,

$$J_{i,j} = \alpha d_{s_{i,j}} + \beta d_{p_{i,j}} + \gamma(1 - IoU_{i,j}) \tag{2}$$

where $d_{s_{i,j}}$ is a similarity distance (see below), $d_{p_{i,j}}$ is the Euclidean distance in image coordinates, normalised with relation to object diagonal and $IoU_{i,j}$ is the IoU between object i and j. The α, β and γ terms are weighs of the three. The cost is only computed between objects of the same class, i.e. a ship does not suddenly become a buoy and vice versa. A new detection is matched with a previously tracked object by minimizing this cost function. If no match is found for a given detection, it is considered a new object, and is added to a list of tracked objects.

The similarity between two objects is found using feature matching. In this paper, the similarity between two objects is computed by comparing feature maps from the object detector. As the area obtained by the two objects in the feature maps seldom is the same, the feature maps of each object are converted to a fixed size before matching. The reshaped feature map is denoted the feature vector of the object.

The feature vector of an object is extracted by first locating an object in each of four feature maps, illustrated in Figure 3.

The four feature maps are illustrated in Figure 3. The width of each feature map is given as a fraction of the input image width, such that the widths of the four feature maps are $\frac{1}{16}$, $\frac{1}{32}$, $\frac{1}{63}$ and $\frac{1}{120}$ of the input image. Each feature map can be thought of as a 256 channel image, with each channel corresponding to the activation of a convolutional filter in the NN. The feature vector of a certain object is constructed by finding the average intensity of each channel in the feature maps, in the area corresponding to where the object is present, and then concatenating these into a vector, using global average pooling and concatenating. This action is described by Equation 3.

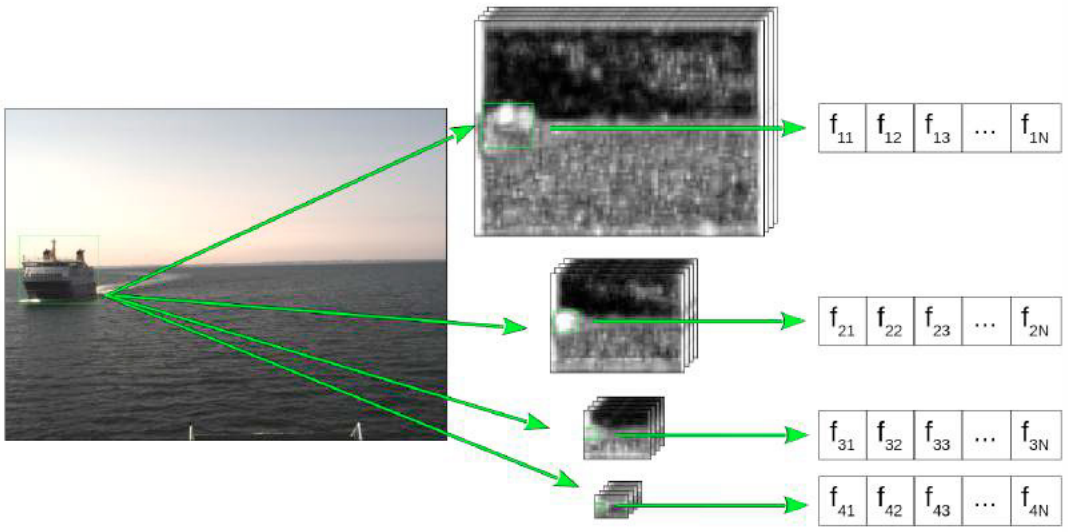

Fig. 3. Image of a ship along with the respective feature maps. The ship position in each feature map is highlighted by a green box

$$f_{i,j,k} = \frac{1}{N_{i,j}} \sum_{\lfloor x_{i_0} FM_{j_w} \rfloor}^{\lceil x_{i_1} FM_{j_w} \rceil} \sum_{\lfloor y_{i_0} FM_{j_h} \rfloor}^{\lceil y_{i_1} FM_{j_h} \rceil} FM_{j,k,xy} \tag{3}$$

$$\boldsymbol{f_{i,j}} = [f_{i,j,1}, f_{i,j,2} \dots f_{i,j,K}] \tag{4}$$

$$\boldsymbol{f_i} = [\boldsymbol{f_{i,1}}, \boldsymbol{f_{i,2}}, \boldsymbol{f_{i,3}}, \boldsymbol{f_{i,4}}] \tag{5}$$

where $f_{i,j,k}$ is the average feature of object i from feature map j in depth k, (x_{i_0}, y_{i_0}) is the upper left corner of the bounding box of object i indexed by fraction of image width and height respectively, (x_{i_1}, y_{i_1}) is the lower right corner of the bounding box of object i indexed by fraction of image width and height respectively, FM_{j_w} is the pixel width of feature map j, FM_{j_h} if the pixel height of feature map j, $N_{i,j} = (\lceil x_{i_1} FM_{j_w} \rceil - \lfloor x_{i_0} FM_{j_w} \rfloor)(\lceil y_{i_1} FM_{j_h} \rceil - \lfloor y_{i_0} FM_{j_h} \rfloor)$ is the pixels occupied by object i in feature map j, $FM_{j,k,xy}$ is the feature value of feature map j at depth k in position (x, y). $\boldsymbol{f_{i,j}}$ is the feature vector of object i composed from the features of feature map j up until the maximum depth K. $\boldsymbol{f_i}$ is the general feature vector of object i composed from all feature maps.

The similarity between two objects a and b with feature vectors $\mathbf{a}$ and $\mathbf{b}$ is expressed by the scalar product between unit vectors in directions of the two feature vectors. In Equation 6, $d_{s_{a,b}}$ is often referred to as the cosine distance, although it is not a measure of distance, it does not meet the triangle inequality for example, but is a measure of similarity in direction of the two feature vectors.

$$d_{s_{a,b}} = 1 - \frac{\boldsymbol{f_a} \cdot \boldsymbol{f_b}}{||\boldsymbol{f_a}||\ ||\boldsymbol{f_b}||} \tag{6}$$

A measure $d_{p_{i,j}}$ is found using the Euclidean distance in image plane between centre of a detected object and the predicted centre position of a tracked object. In order to calculate this, we need to predict the position of all tracked objects in the next image to come. This is done using the predictor presented in Equation 1. The $\mathbf{K}$ matrix is implemented as a conventional linear Kalman filter. The implementation adopted in this work is that of (Bewley et al., 2016). This predictor estimates the bounding box velocity allowing the bounding box velocity to change as an object changes distance to the viewpoint, in this case own vessel.

Finally, a measure $(1 - IoU_{i,j})$ is calculated from the IoU between objects i and j. This means that if the bounding box of objects i and j has overlap, this measure contributes with a low cost to the matching.

If a detection is matched with a tracked object, the detected bounding box is used to update the Kalman filter. If no detection is matched with an object, the Kalman filter will keep predicting the position of the object for a set number of frames before it is deleted from the list of objects in view.

After the cost is computed for each set of tracked objects and detections, the matching pairs are found by solving the linear assignment problem.

5. RESULTS

This section presents an evaluation of the proposed tracking algorithm. The method will be evaluated against the *SORT* tracker of Bewley et al. (2016), as the Kalman filter used in the proposed method is based on the *SORT* tracker.

The suggested tracker will be evaluated using the metrics MOTA and MOTP (Bernardin and Stiefelhagen, 2008), number of ID switches, precision and recall.

MOTA is defined as follows:

$$MOTA = 1 - \frac{\sum_t (m_t + fp_t + mme_t)}{\sum_t g_t} \tag{7}$$

where m_t is the number of missed objects at time t, fp_t is the false positives at time t, mme_t is the number of missed detections at time t and g_t is the number of ground truth objects at time t. MOTA is a general measure of how able a tracker is at keeping accurate trajectories, independent of its ability to estimate object position compared to ground truth.

Fig. 4. Cropped example of an annotated image from the data set. Objects are far away and are therefore rather small in most images.

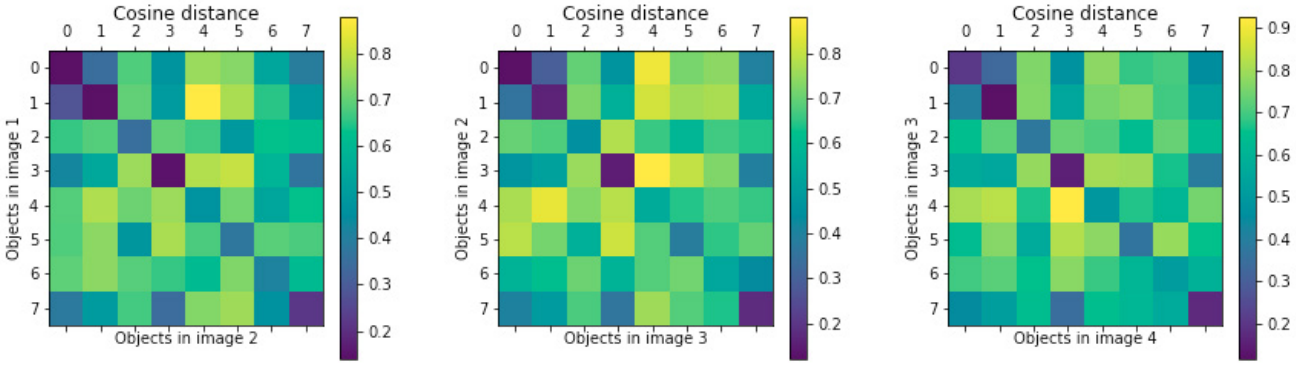

Fig. 5. The cosine distance between feature vectors in four consecutive images.

MOTP is defined as:

$$MOTP = \frac{\sum_{i,t} d_{i,t}}{\sum_t c_t} \tag{8}$$

where $d_{i,t}$ is the distance between the object o_i and its match and c_t is the number of matches at time t. MOTP is therefore the total positioning error averaged by the number of matches. MOTP describes a trackers ability to estimate the position of an object regardless of trajectory accuracy.

Finally, precision and recall are defined as:

$$\begin{aligned} precision &= \sum_t \frac{c_t}{c_t + fp_t} \\ recall &= \sum_t \frac{c_t}{c_t + mme_t}, \end{aligned} \tag{9}$$

Precision describes the ratio between true predictions and all predictions, while recall describes the probability to detect an object.

The suggested method is evaluated on a sequence of self-annotated images acquired on-board a ferry in the South Funen archipelago in Denmark. The sequence consists of 337 images with a resolution of $1440 \times 1080px$, containing 2208 instances of objects. The sequence also includes multiple occlusions, e.g. a ship being occluded by another ship for 9 frames. The NN used for object detection was trained on similar data, with the testing sequence not being included in the training. An example image from the sequence is shown in Figure 4.

A predicted tracking is determined to be true if the IoU between the predicted object and a ground truth box is above 0.2. The relatively low threshold is set as such because of the abundance of small objects in the evaluation sequence due to objects being far away. During evaluation, the proposed method will use a weighting of $\alpha = 0.2$, $\beta = 1$ and $\gamma = 0.4$ for the cost function presented in Eq. 2.

Table 2. Results for (Bewley et al., 2016) (*SORT*), a features-only variant (Features) and the full proposed algorithm (Proposed).

	MOTA	MOTP	Switch	P	R	*FPS*
SORT	0.58	**0.36**	48	0.79	0.81	**14.3**
Features	0.62	0.37	39	0.804	0.85	13.7
Proposed	**0.70**	0.37	**35**	**0.83**	**0.89**	13.7

Table 2 shows that the proposed method outperforms the *SORT* tracker in all measures but speed and MOTP, however the method proposed in this paper is only slightly slower. Using only feature matching, i.e setting $\alpha = 1$, $\beta = 0$ and $\gamma = 0$ in Equation 2, the algorithm still outperforms the *SORT* tracker, in terms of MOTA, number of ID-switches, precision and recall. This indicates that feature matching alone is a relatively good matching scheme for tracking.

In the above table, *FPS* is calculated after the object detection step, thereby only measuring the time taken for

tracking. Note that there is only a slight drop in *FPS* when including feature matching. This is a result of the features already being computed from the object detection. Therefore, it is only necessary to form the feature vector, which was found to take 0.49 ms. This method is significantly faster than other methods utilizing feature matching for tracking, e.g. the *deepSORT* tracker in which feature extracting takes 30 ms on a Nvidia GeForce GTX 1050 GPU Wojke et al. (2017).

It is furthermore relevant to look into the percentage of lifetime, i.e. an objects appearance in the data set, that was successfully tracked by the proposed algorithm. Such an analysis elucidates the tracker's ability to keep long tracks. Table 3 shows, for 12 indicative objects, that most of them are tracked for more than 80% of their lifetime, and few objects are tracked for less than 20%. The lost objects are mostly very distant vessels in the sequence which are only within view for a short period of time.

Table 3. Percentage of lifetime tracked by the proposed algorithm.

	TOTAL	>80%	20-80%	<20%
# Objects	12	8	2	2

Another relevant measure is the number of fragmentations. This is the amount of times a track is lost. For the proposed method, the number of fragmentations is 44. As it was found that the number of ID-switches was 35, the percentage of lost tracks that are picked up again with the right ID is 20%.

The similarity distance between all objects in one image and the next are illustrated in Figure 5, where it is seen that feature matching indeed contributes to the tracking, as a low cosine distance is apparent diagonally. The figure also illustrate the need for a Kalman filter to utilise position, as some objects have a low similarity distance to more than one other object.

One advantage of a tracking algorithm is that when looking at a sequence of images, the backbone object detection network might miss a target that the tracker is still able to follow, thereby increasing the recall for a sequence. To assess this increase, the object detection network is evaluated on the sequence, disabling tracking. This yields a recall of $r = 0.821$ or a missed detection probability of 17.9%. With the proposed tracking method, this is reduced to 11% missed detections.

Finally, it is noted that tracking results depend heavily on the quality of the object detection step. Evaluating tracking using a detection network that was trained on data, which include the tracking sequence, yields the results presented in Table 4.

Table 4. Results using a network that has been trained on the evaluation sequence

	MOTA	MOTP	Switch	P	R	*FPS*
SORT	0.9	0.29	16	0.93	0.98	**14.3**
Features	0.9	0.29	10	0.92	0.98	13.7
Proposed	**0.93**	**0.28**	**5**	**0.94**	**0.99**	13.7

6. CONCLUSIONS

This paper proposed a novel method for tracking of objects that have been detected by a convolutional neural network. Using feature matching together with a Kalman filter predictor, the method showed promising results. This was due to the reuse of the features computed during the detection step of the NN, thereby increasing throughput compared to other methods, which compute new features during tracking. Evaluating the tracking method in a marine environment showed that the proposed method performed as desired with a low number of ID-switches and a high frame rate.

ACKNOWLEDGEMENTS

This research was sponsored by the The Danish Maritime Fund, Orients Fund and the Lauritzen Foundation through the ASAN project and bny the same foundations and the Danish Innovation Fund through the ShippingLab project, grant 8090-00063A, (the Work Package on Autonomy).

REFERENCES

Bernardin, K. and Stiefelhagen, R. (2008). Evaluating multiple object tracking performance:the clear mot metrics. *EURASIP Journal on Image and Video Processing*. doi:doi:10.1155/2008/246309.

Bewley, A., Ge, Z., Ott, L., Ramos, F., and Upcroft, B. (2016). Simple online and realtime tracking. *2016 IEEE International Conference on Image Processing (ICIP)*. doi:10.1109/icip.2016.7533003.

Blanke, M., Hansen, S., Stets, J.D., Koester, T., Brøsted, J.E., Llopart Maurin, A., Nykvist, N., and Bang, J. (2019). Outlook for navigation – comparing human performance with a robotic solution. In *Proc. of ICMASS'2018*. SINTEF Academic Press. URL `http://hdl.handle.net/11250/2599009`.

Chatfield, K., Simonyan, K., Vedaldi, A., and Zisserman, A. (2014). Return of the Devil in the Details: Delving Deep into Convolutional Nets. In *British Machine Vision Conference*. doi:10.5244/C.28.6.

Dara, S. and Tumma, P. (2018). Feature extraction by using deep learning: A survey. In *Second International Conference on Electronics, Communication and Aerospace Technology (ICECA)*, 1795–1801. doi:10.1109/ICECA.2018.8474912.

Henriques, J.F., Caseiro, R., Martins, P., and Batista, J. (2015). High-speed tracking with kernelized correlation filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37, 583–596.

Hou, X., Wang, Y., and Chau, L. (2019). Vehicle tracking using deep sort with low confidence track filtering. In *2019 16th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, 1–6.

Jadhav, A., Mukherjee, P., Kaushik, V., and Lall, B. (2020). Aerial multi-object tracking by detection using deep association networks. In *2020 National Conference on Communications (NCC)*, 1–6.

Kapania, S., Saini, D., Goyal, S., Thakur, N., Jain, R., and Nagrath, P. (2020). Multi object tracking with uavs using deep sort and yolov3 retinanet detection framework. In *ACM Workshop on Autonomous and Intelligent Mobile Systems*. Association for Computing Machinery (ACM). doi:10.1145/3377283.3377284.

Lappe, R.R. (2015). *Kalman and Bayesian Filters in Python*. URL `https://github.com/rlabbe/Kalman-and-Bayesian-Filters-in-Python`.

Lin, T.Y., Goyal, P., Girshick, R., He, K., and Dollár, P. (2017). Focal loss for dense object detection. In *Proc. 2017 IEEE Int. Conf. on Computer Vision (ICCV)*. IEEE. doi:10.1109/iccv.2017.324.

Porathe, T., Prison, J., and Man, Y. (2014). Situation awareness in remote control centres for unmanned ships. In *Proceedings of Human Factors in Ship Design & Operation, 26-27 February 2014, London, UK p. 93-101*.

Ren, S., He, K., Girshick, R., and Sun, J. (2017). Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(6), 1137–1149. doi:10.1109/TPAMI.2016.2577031.

Schöller, F.E.T., Plenge-Feidenhans'l, M.K., Stets, J.D., and Blanke, M. (2019). Assessing deep-learning methods for object detection at sea from LWIR images. *Proc. CAMS'2019, IFAC-PapersOnLine*, 52(21), 64–72. doi:10.1016/j.ifacol.2019.12.284.

Tang, Z., Naphade, M., Liu, M.Y., Yang, X., Birchfield, S., Wang, S., Kumar, R., Anastasiu, D., and Hwang, J.N. (2019). Cityflow: A city-scale benchmark for multi-target multi-camera vehicle tracking and re-identification. In *The IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Wojke, N., Bewley, A., and Paulus, D. (2017). Simple online and realtime tracking with a deep association metric. *2017 IEEE Int. Conf. on Image Processing (ICIP)*. doi:10.1109/icip.2017.8296962.

Zou, Y., Zhang, W., Weng, W., and Meng, Z. (2019). Multi-vehicle tracking via real-time detection probes and a markov decision process policy. *Sensors*, 19(6), 1309. doi:10.3390/s19061309.