2nd Workshop on Maritime Computer Vision (MaCVi) 2024: Challenge Results

Benjamin Kiefer[1], Lojze Žust[2], Matej Kristan[2], Janez Perš[2], Matija Teršek[3], Arnold Wiliem[4,5], Martin Messmer[1], Cheng-Yen Yang[6], Hsiang-Wei Huang[6], Zhongyu Jiang[6], Heng-Cheng Kuo[6], Jie Mei[6], Jenq-Neng Hwang[6], Daniel Stadler[7,15], Lars Sommer[7,15], Kaer Huang[8], Aiguo Zheng[9], Weitu Chong[10], Kanokphan Lertniphonphan[8], Jun Xie[8], Feng Chen[8], Jian Li[9], Zhepeng Wang[8], Luca Zedda[11], Andrea Loddo[11], Cecilia Di Ruberto[11], Tuan-Anh Vu[12], Hai Nguyen-Truong[12], Tan-Sang Ha[12], Quan-Dung Pham[12], Sai-Kit Yeung[12], Yuan Feng[13], Nguyen Thanh Thien[14], Lixin Tian[13], Sheng-Yao Kuan[6], Yuan-Hao Ho[6], Angel Bueno Rodriguez[16], Borja Carrillo-Perez[16], Alexander Klein[16], Antje Alex[16], Yannik Steiniger[16], Felix Sattler[16], Edgardo Solano-Carrillo[16], Matej Fabijanić[17], Magdalena Šumunec[17], Nadir Kapetanović[17], Andreas Michel[7], Wolfgang Gross[7], Martin Weinmann[18]

[1]University of Tuebingen, [2]University of Ljubljana, [3]Luxonis, [4]Sentient Vision Systems, [5]Queensland University of Technology, [6]University of Washington, [7]Fraunhofer IOSB, [8]Lenovo Research, [9]Lenovo, [10]Fudan University, [11]University of Cagliari, [12]The Hong Kong University of Science and Technology, [13]Dalian Maritime University, [14]University of Information Technology, [15]Fraunhofer Center for Machine Learning, [16]German Aerospace Center, [17]University of Zagreb, [18]Karlsruhe Institute of Technology

Abstract

The 2nd Workshop on Maritime Computer Vision (MaCVi) 2024 addresses maritime computer vision for Unmanned Aerial Vehicles (UAV) and Unmanned Surface Vehicles (USV). Three challenges categories are considered: (i) UAV-based Maritime Object Tracking with Re-identification, (ii) USV-based Maritime Obstacle Segmentation and Detection, (iii) USV-based Maritime Boat Tracking. The USV-based Maritime Obstacle Segmentation and Detection features three sub-challenges, including a new embedded challenge addressing efficicent inference on real-world embedded devices. This report offers a comprehensive overview of the findings from the challenges. We provide both statistical and qualitative analyses, evaluating trends from over 195 submissions. All datasets, evaluation code, and the leaderboard are available to the public at `https://macvi.org/workshop/macvi24`.

1. Introduction

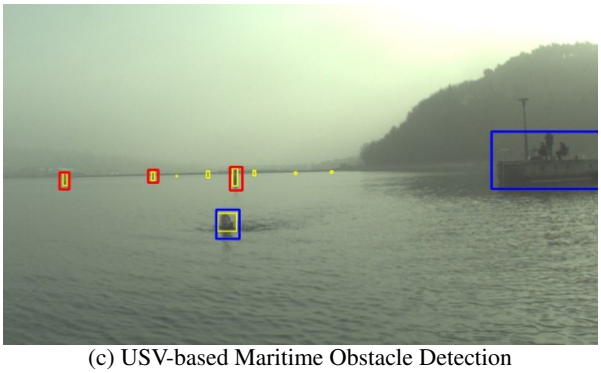
Maritime environments, encompassing both the vast open seas and intricate coastlines, have always been of paramount importance for global trade, exploration, and scientific research. Over the past years, there has been a rapid increase in the deployment of autonomous robotic platforms, particularly Unmanned Aerial Vehicles (UAVs) and Unmanned Surface Vehicles (USVs), to augment and sometimes even replace traditional human-driven operations in this domain. These technological marvels, while offering unparalleled advantages in terms of scalability, efficiency, and safety, heavily rely on state-of-the-art computer vision systems to navigate, detect, and interpret their maritime surroundings.

In response to the growing interest and the need for specialized computer vision solutions tailored for the maritime domain, the first Workshop on Maritime Computer Vision (MaCVi) [25] was organized in 2023. The workshop highlighted research challenges, introduced standardized benchmarks brought together researchers and practitioners worldwide, fostering collaboration and driving innovation. The 2nd Workshop on Maritime Computer Vision (MaCVi2024), organized in conjunction with WACV2024, is a continuation of these efforts. MaCVi2024 introduces the following challenges as shown in Figure 1: UAV-based Multi-Object Tracking (MOT) w/ Re-identification, USV-based Obstacle Segmentation, USV-based Embedded Semantic Segmentation, USV-based Obstacle Detection, and USV-based MOT.

These competitions either are extensions from the last workshop (UAV-based MOT w/ ReID), employ new datasets (USV-based Obstacle Detection and Segmentation), or are new (USV-based MOT). Moreover, in response to the feed-

(a) UAV-based Maritime MOT w/ re-identification

(b) USV-based Obstacle Segmentation &
USV-based Embedded Obstacle Segmentation



(c) USV-based Maritime Obstacle Detection

(d) USV-based Multi-Object Tracking

Figure 1: Overview of MaCVi 2024 challenges.

back and technological constraints highlighted in the previous MaCVi edition, this workshop features a new embedded sub-track within the Obstacle Segmentation challenge. The goal is to foster the development of lightweight, yet effective, computer vision algorithms suitable for deployment on devices with limited computational resources, commonly found on UAVs and USVs.

This report serves as a compass, guiding readers through the key findings, methodologies, and innovations presented during the workshop. The rest of the paper is organized as follows: First, we provide an overview of the challenge protocol (Section 2), before we review the outcomes of the individual challenge tracks (Sections 3 and 4) with their underlying benchmarks and datasets.

2. Challenge Participation Protocol

The challenge tracks were announced and opened for participation on the 20th of September 2023. Participants were granted access to download the datasets, evaluation, and visualization toolkits from the workshop homepage[1]. This allowed participants ample time to experiment with their methods on the provided data.

We opened the challenge server for submissions on the 20th of September 2023. For all challenge tracks except the embedded one, the participants were required to uploaded their predictions for evaluation. For the embedded challenge, participants were required to export their models to ONNX for on-device benchmarking. The BoaTrack upload period started on the 18th of October.

To prevent overfitting, the participants were restricted to a single upload per day per challenge. After evaluation, the submissions appeared on the public leaderboard. Participants were offered the option to withdraw their submissions at any time. The challenge closed on the 3rd of November 2023, while new submissions after the 27th of October were hidden from others and only revealed after the conclusion of the challenges.

To ensure a high standard of submissions, participants were informed that their submitted predictions would undergo further scrutiny. They were also mandated to provide comprehensive information about their methods. The pre-specified primary metrics for each challenge track determined the top-3 positions. Additionally, every participant was asked to provide information regarding the running time of their method, measured in frames per second wall clock time, their hardware specifications, and any extra datasets utilized (including those for pretraining).

In line with our commitment to advancing the state of maritime computer vision, top three ranked teams for all challenges were invited to submit a technical report detailing their methods and training configurations due by the 7th of November 2023. These insightful reports are attached in the appendix of this paper. The results were presented at the MaCVi2024 workshop on the 7th of January 2024.

[1] https://macvi.org/workshop/macvi24

2.1. Evaluation Server

Following the announcement of the Maritime Computer Vision initiative, the evaluation server transitioned to a new domain at `macvi.org`. This move was not just a change of address; the webpage underwent a comprehensive restructuring to better accommodate the integration of new challenge tracks, ensuring scalability and ease of expansion for future challenges.

The current server is an evolution of the original web server used for the SeaDronesSee benchmark and has been readily accessible online several months prior to the challenge commencement. Alongside the main challenge tracks, it also supports additional tracks such as Boat-MNIST, UAV-based Object Detection v1/v2 and UAV-based Single-Object Tracking.

3. UAV-based Object Tracking Challenge

The second iteration of the SeaDronesSee benchmark introduces an enhanced Multi-Object Tracking (MOT) track, now incorporating re-identification capabilities. This advancement builds on the initial focus of tracking objects in marine Search and Rescue (SaR) and surveillance scenarios; adding a critical layer of maintaining consistent subject identification. In this iteration, the challenge extends beyond tracking the movement and positions of people or boats over time to include re-identification, especially crucial for subjects that temporarily leave the field of view or become occluded.

The core challenges of tracking small, partly occluded subjects whose appearances change with movement and water-induced occlusion remain. Additionally, the complexities due to gimbal movement and altitude changes, causing rapid object motion within video frames, are addressed. The SeaDronesSee-MOT with re-identification track is designed to advance technologies in dynamic tracking and identification, essential for improving SaR and surveillance operations. The next section will explore the details and innovations of this updated challenge track.

3.1. Dataset

The updated SeaDronesSee-MOT dataset for the re-identification challenge encompasses enhancements, focusing primarily on the test set which now includes re-identification labels. This dataset comprises 21 train set clips, 17 in the validation set, and 19 in the test set, amounting to 54,105 frames.

As demonstrated in Table 1, the updated test set, enriched with re-identification labels, aims to track boats, swimmers, and floaters under a unified class setting, not distinguishing between these classes. This unified approach is pivotal for short-term tracking tasks, where objects that exit the scene are not tracked but are expected to be re-identified when they reappear. Naturally,the re-identification dataset often exhibits fewer unique object IDs than the standard MOT dataset. In instances where the re-identification dataset shows a greater number of unique IDs, this increase is attributed to the correction of previously mislabeled or unlabelled instances, enhancing the dataset's accuracy.

Table 1: Comparison of Unique Object IDs in SeaDronesSee-MOT and -MOT with Re-identification Datasets on the SeaDronesSee-MOT test set. The table shows the number of unique object IDs for each video ID in both datasets (video IDs 7, 8 and 20 do not exist).

Video ID	Unique IDs	Unique IDs (ReID)
0	11	12
1	15	15
2	5	5
3	10	10
4	120	34
5	11	11
6	11	10
9	14	11
10	6	7
11	3	3
12	8	8
13	6	6
14	4	4
15	10	4
16	10	10
17	5	5
18	28	27
19	1	1
21	139	23

Additionally, each frame in this dataset is complemented by comprehensive metadata, including UAV altitude, gimbal angles, GPS coordinates, and more. This detailed metadata may aid in precise object tracking and identification.

3.2. Evaluation Protocol

We evaluate the submissions by using the following metrics: HOTA, MOTA, IDF1, MOTP, MT, ML, FP, FN, Recall, Precision, ID Switches, Frag [28, 36]. The determining metric for winning is HOTA. In case of a tie, MOTA is the tiebreaker.

Furthermore, we require every participant to submit information on the computational runtime of their method measured in frames per second wall-clock time along their used hardware.

3.3. Submissions, Analysis and Trends

We received 49 submissions from 10 different teams. Additionally, we provided the same baseline as last time, i.e. a

Table 2: Multi-Object Tracking with Re-identification submissions overview. For brevity, we denoted all=train and val set.

Model name	Data	Detector	FPS	GPU
MG-MOT (UWIPL) (A.1) [48]	COCO, SeaDronesSee-MOT^{all}	YOLOv8-x	13	GV100
Tracking by Detection (Fraunhofer IOSB) (A.2)	COCO, SeaDronesSee-MOT^{all}	Varifocal-Net	1	V100
ReIDTracker-Sea (Lenovo) (A.3)	SeaDronesSee-MOT^{all}	Swin-Transformer	3	A100

Tracktor-based tracker using ECC with a Faster R-CNN ResNet-50 detector (A.4) without re-identification.

40 submitted trackers outperformed the baseline. However, we are going to restrict our analysis on the best tracker of each of the best three teams.

See an overview of the submitted methods in Table 2. Table 3 shows the results of the best submissions of the best five teams. Again, the winner submissions followed the tracking-by-detection paradigm. Interestingly, winning submissions each employed a different object detector backbone: a one-stage YOLOv8-x, a two-stage Varifocal Net (ResNet-50), and a Transformer.

MG-MOT (A.1) and *TBD* (A.2) performed almost on-par in both metrics, HOTA and MOTA. While *MG-MOT* has significantly fewer fragmentions, *TBD* has the lowest number of identity switches (only 16). The third place, *ReIDTracker-Sea* (A.3) is on-par in terms of MOTA, but considerably worse in HOTA and IDF1, indicating its inferiority associating objects. In fact, its number of ID switches and fragmentions are considerably higher, while precision and recall values are at the top. This is illustrated in Figure 2, showing a common fragmention and re-identification error caused by a sudden movement of the camera. In this particular case, we suspect the missing motion model of *ReIDTracker-Sea* to be the cause for this as both, *MG-MOT* and *TBD* don't do these errors.

While these errors where caused by camera movements in the *short-term*, a common source of errors was caused by very similar looking boats in the *long-term*. Figure 2 shows how a boat is assigned an incorrect ID after it has left the scene and re-entered it. This is a special case of long-term tracking where it is particularly hard for any metadata-agnostic re-identification module to re-identify the instance correctly. The winning model *MG-MOT* is the only submission that does not assign incorrect IDs in this specific example. This is likely due to its metadata-guided module that leverages metadata to obtain an understanding of the objects' topology.

3.4. Discussion

The UAV-based Multi-Object Tracking with Reidentification challenge highlighted challenges in differentiating similar objects from high altitudes during fast camera movements. The winning team, *MG-MOT*, successfully incorporated onboard metadata to address these issues. *Tracking by Detection*, in second place, also showcased strong performance with a well-tuned existing framework. The third-place team, *ReIDTracker-Sea*, utilized a Transformer for effective detection, though the absence of a motion model may have affected their association accuracy.

4. USV-based Perception Challenges

The following USV-oriented subchallenges were organized: (i) obstacle detection, (ii) obstacle segmentation, (iii) multi-object tracking, and (iv) the embedded segmentation challenge. The latter is the first of its kind, dedicated to promoting development of perception methods capable of running on embedded hardware. The challenges used two datasets: the LaRS benchmark [57] was used in segmentation and detection challenges, while the multi-object tracking challenge applied a dataset BoaTrack datast, captured by LOOKOUT [1]. Both of these datasets feature scenes captured from the viewpoint of USVs and boast a large scene and obstacle variety (see Figure 1).

4.1. Datasets

4.1.1 LaRS

The obstacle segmentation, embedded obstacle segmentation and obstacle detection challenges, used the recently released LaRS benchmark [57]. It contains over 4000 challenging and visually diverse maritime and inland scenes (see Figure 3) with panoptic obstacle annotations, featuring locations around the world. The panoptic annotations contain 3 stuff categories (sky, water and static obstacles) and 8 different dynamic obstacle instance categories (e.g. boats, buoys, swimmers). In addition, LaRS is annotated for semantic segmentation, where all static and dynamic obstacle categories are merged into a single obstacle segmentation mask.

LaRS features challenging scenarios such as scenes with object reflections, sun glitter and bad visibility. To this end, scenes have been additionally labeled with several scene-level attributes to allow for detailed analysis. LaRS is split into train, validation and test sets. The annotations for the train and validation sets are publicly available, while the annotations of the test set are withheld, to ensure fair comparison of methods. Instead, an online evaluation server is hosted to evaluate user-submitted predictions. For more details on the data acquisition and annotation processes refer to [57].

Table 3: Multi-Object Tracking results on the SeaDronesSee-MOT test set. The submissions are ranked based on HOTA.

Model name	HOTA	MOTA	IDF1	MOTP	MT	ML	FP	FN	Re	Pr	IDs	Frag
MG-MOT	0.695	0.771	0.859	0.208	153	27	9531	12351	0.871	0.897	18	783
TBD	0.693	0.780	0.844	0.205	165	20	10643	10391	0.891	0.889	16	984
ReIDTracker-Sea	0.624	0.781	0.713	0.204	150	32	9595	11166	0.883	0.898	178	1343

Figure 2: Common fragmentation and reidentification errors (here *ReIDTracker-Sea*). **Top:** The detected swimmer in the first frame is lost in one of the next frames due to a heading angle change and is assigned a new ID when it is re-detected in a later frame. **Bottom:** A boat is not re-detected re-entering the scene after having left it. This may be caused by the great similarity between the different boats.

Table 4: BoaTrack statistics. The average lifespan indicates how long an ID lives across frames before it leaves the scene.

Video	# Objects	Unique IDs	Avg. Lifespan
Video 1	36,116	130	277.82
Video 2	1,180	3	393.33
Video 3	66,072	94	702.89
Total	**103,368**	**227**	**455.37**

4.1.2 BoaTrack

BoaTrack is a new multi-object tracking (MOT) dataset recorded from the viewpoint of USVs. The goal of BoaTrack is to advance computer vision algorithms in autonomous boating and boating assistance systems. It is aimed at detecting and tracking boats and other objects (such as buoys) in open water and dock scenarios. For the purposes of the MaCVi challenge, only boats are to be tracked. Participants needed to submit object locations and IDs for each frame.

The focus is on short-term tracking, i.e., re-identification of objects that leave and re-enter the field of view is not required.

The test set contains three videos clips, totaling 10,656 frames. Please note, that we did not provide any train or val videos. This means, participants needed to train a detector on other datasets, such as the aforementioned LaRS, but were free to use any other publicly available dataset for training. Please see an overview of the dataset in Table 4. There are 8,255 frames for the first video and 901 and 1,500 for the second and third, respectively.

Figure 3: Examples of scenes in the LaRS benchmark.

4.2. USV-based Obstacle Segmentation Challenge

The methods participating in the USV-based Obstacle Segmentation Challenge were required to predict the scene segmentation (into obstacles, water and sky) for a given input image. The submitted methods have been evaluated on the recently released LaRS benchmark [57]. In addition to the publicly available training set, the authors were also allowed to use additional datasets (upon declaration) for training their methods.

4.2.1 Evaluation Protocol

To evaluate segmentation predictions, we employ the LaRS [57] semantic segmentation evaluation protocol. Segmentation methods provide per-pixel labels of semantic components (water, sky and obstacles). However, traditional approaches for segmentation evaluation (*e.g*. mIoU) do not consider the aspects of predictions that are relevant for USV navigation. Instead, the LaRS protocol evaluates the predicted segmentations with respect to the downstream tasks of navigation and obstacle avoidance and focuses on the detection of obstacles.

The detection of static obstacles (*e.g*. shoreline) is measured by the water-edge accuracy (μ), which evaluates the segmentation accuracy around the boundary between the water and static obstacles. On the other hand, the detection of dynamic obstacles (*e.g*. boats, buoys, swimmers) is evaluated by counting true-positive (TP), false-positive (FP) and false-negative (FN) detections, summarize by the F1 score. A ground-truth obstacle is counted as a TP if the intersection with the predicted obstacle segmentation is sufficient, otherwise it is counted as a FN. FPs are counted as the number of segmentation blobs (after connected components) on areas annotated as water in ground truth. For further details please see [57].

The detection F1 score is a great indicator of the quality of the method predictions as all obstacles have equal importance in the final score, regardless of their size. However, in the trivial case of predicting everything as an obstacle, all ground-truth obstacles will be counted as TP and there will be only one FP blob per image (albeit very large), which leads to a very large F1 score. On the other hand, mIoU measures the overall segmentation quality on a per-pixel level, but does not reflect the detection of smaller obstacles very well. We thus combine the two measures into a single quality measure ($\text{Q} = \text{F1} \cdot \text{mIoU}$) and use it as the primary performance measure in this challenge.

4.2.2 Submissions, Analysis and Trends

The USV-based obstacle segmentation challenge received 49 submissions from 7 different teams. As per the rules of the challenges, only the best-performing method from each team is considered in the following analysis. Results of the remaining submitted methods are available on the public leaderboards of the challenge on the MaCVi website [26]. Table 5 presents the overall standings of the teams participating in the challenge and the results of their best performing methods. In addition to the competing teams, we also analyze two baselines provided by the MaCVi 2024 committee, namely DeepLabv3 and K-Net, which was the previous state-of-the-art on the LaRS benchmark [57]. In this section we analyze all the contributed methods, with a special focus on the top three approaches. We refer to teams and their methods by their overall ranking in Table 5 with the notation (#n), where n is the place of the approach. Short descriptions of the top three methods are available in the Appendix B.

Overall, four teams, (#1) UniCa, (#2) HKUST, (#3) DLMU and (#4) HSU, improved over the previous state-of-the-art performance of K-Net [53] by quite a large margin. Namely, the top performing method SWIM^2 outperforms it by 6.8% Q. Interestingly, all these methods are based on the popular Mask2Former architecture [16] with slight alterations. Specifically, SWIM^2 adapts the training strategy and insights from biomedical computer vision to improve the recognition of small objects, which are common in LaRS and Mari-Mask2Former utilizes the dice loss, to address the issue of class imbalance. The top-two methods, SWIM^2 and TransMari, achieved quite a similar performance, with SWIM^2 outperforming TransMari by only 0.3% Q. TransMari performed slighly better in F1 detection score, but ranked lower in overall segmentation quality (mIoU).

Real-time semantic segmentation methods eWaSR [42] and PidNet [47] were also considered by two teams, but have been outperformed by computationally more intensive methods. (#6) utilized the recent OneFormer [24] architecture for universal semantic segmentation, but opted to use a lower processing resolution compared to other methods. None of analysed methods utilized additional data during training. In the following, we analyze the best-performing methods of each of the teams in more detail.

Detection by scene attributes: Figure 4 shows the performance with respect to scene attributes, including environment type, illumination, amount of reflections and scene conditions. Overall, the top four methods outperform the baseline (K-Net [53]) in most categories, particularly in challenging scenarios such as heavy reflections, foggy scenes and plants or debris presence in water. SWIM^2 demonstrates remarkable stability across different scenarios and is the only method that consistently outperforms the baseline in all categories.

Performance by obstacle size: Figure 5 compares the detection performance of the top three methods with the K-Net [53] baseline across different obstacle sizes. The largest differences between methods are revealed on small obstacles.

Table 5: Performance of the submitted segmentation methods and baselines (denoted in gray) on the LaRS test set. Performance is reported in terms of water-edge accuracy (μ), precision (Pr), recall (Re), F1 score, segmentation mIoU and overall quality (Q = mIoU $\times$ F1).

Place	Method	Institution	Q ↓	μ	Pr	Re	F1	mIoU
1	SWIM^2 [§B.1]	UniCa	**78.1**	**79.7**	76.9	**83.0**	79.9	97.8
2	TransMari [§B.2]	HKUST	77.8	79.6	78.5	82.0	**80.2**	97.1
3	Mari-Mask2Former [§B.3]	DLMU	75.7	78.4	**79.7**	75.1	77.3	97.8
4th	Mask2Former	HSU	73.8	78.5	76.9	73.8	75.3	**98.0**
-	K-Net	MaCVi	71.3	78.8	67.6	80.4	73.4	97.2
-	DeepLabv3	MaCVi	62.9	77.5	61.1	72.0	66.1	95.2
5th	eWaSR	EAIC-UIT	56.5	67.8	55.5	62.7	58.9	96.0
6th	OneFormer	DLR-MI	52.0	68.3	47.4	62.7	54.0	96.2
7th	PidNet	XJTU	50.7	74.7	47.0	62.8	53.7	94.3

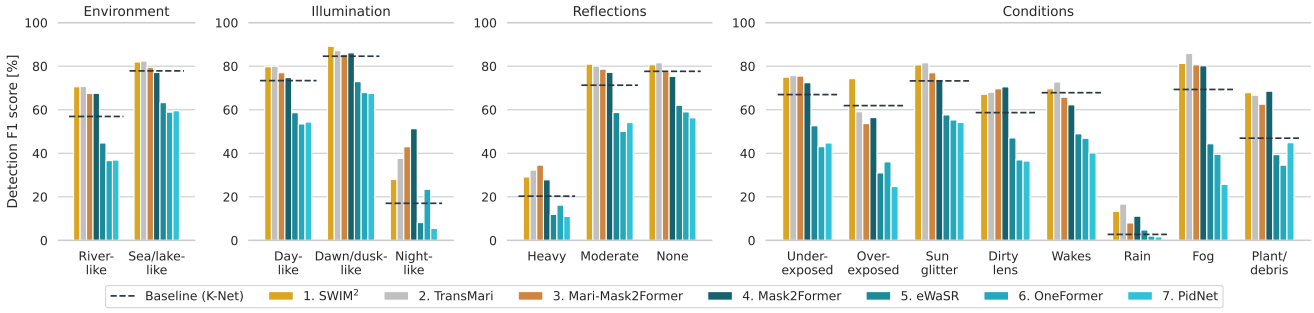


Figure 4: Performance of top segmentation methods with respect to different scene attributes. The performance of the baseline method K-Net is marked as a dotted line for reference.

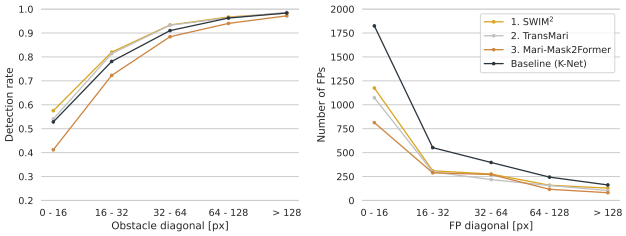


Figure 5: Obstacle detection rate and number of FPs across different sizes of obstacles for top performing methods in comparison to the baseline.

Specifically, we observe that all three methods significantly reduce the number of small false positives, while preserving high detection rates across the board. In particular, both SWIM^2 and TransMari achieve higher detection rates than the baseline across all obstacle sizes, while consistently reducing the number of false positives. Mari-Mask2Former, on the other hand, mainly outperforms the baseline through increased precision, as seen in a drastic reduction of FPs (especially small ones), but detects fewer obstacles overall.

Qualitative results: Examples of predicted scene segmentations are presented in Figure 6. In contrast to the baseline we observe remarkable robustness on night scenes (column 2) and increased segmentation accuracy on thin structures (columns 1 and 3) for methods SWIM^2, Mari-Mask2Former and Mask2Former. OneFormer, which employs a similar architecture, also performs well in night scenes, but is unable to accurately segment thin structures (columns 1 and 3) or smaller objects (column 5) due to operating at a lower resolution. Furthermore, all methods still have some trouble generalizing to varied reflections such as the example shown in column 4.

4.2.3 Discussion and Challenge Winners

The overall winners of the USV-based Obstacle Segmentation challenge are:

1st place: University of Cagliari with SWIM^2,

2nd place: Hong Kong University of Science and Technology (HKUST) with TransMari, and

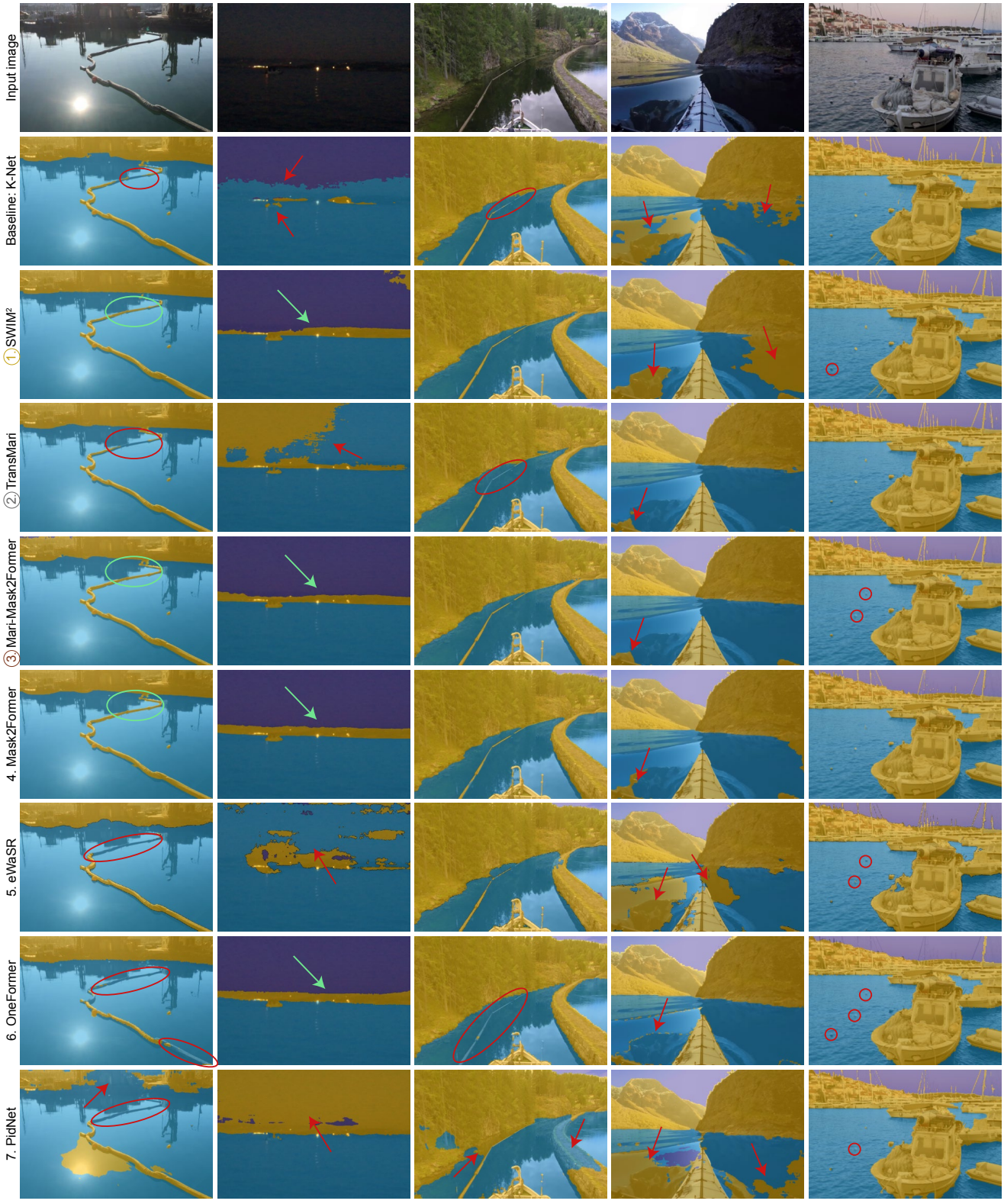


Figure 6: Qualitative comparison of methods for USV-based Obstacle Segmentation. Prominent errors and examples of good behaviour are highlighted.

3rd place: Dalian Maritime University (DLMU) with Mari-Mask2Former

All three methods significantly outperformed the previous state-of-the-art on the LaRS benchmark and demonstrate remarkable robustness to various challenging scenarios such as night scenes, small objects and thin structures. The 1st and 2nd placing SWIM^2 and TransMari methods set a new state-of-the-art for semantic segmentation on LaRS. The detection of tiny objects and robustness to novel water reflections remain open issues and we expect that the performance will continue to improve as new methods are developed to tackle these problems.

4.3. USV-based Embedded Obstacle Segmentation

Recent state-of-the-art obstacle detection methods typically utilize computationally expensive and power-hungry hardware, which makes them unsuitable for small-sized energy-constrained USVs [42]. To bridge this gap, the focus should be put on optimizing existing methods and developing new techniques. Thus, this challenge is an extension of the USV-based Obstacle Segmentation challenge described in Section 4.2, with the main goal of developing an obstacle segmentation method suitable for deployment on an embedded device. The methods are run, benchmarked, and evaluated on a real-world device – an upcoming next-gen device from Luxonis based on Robotic Vision Core 4 (RVC4).

4.3.1 Evaluation Protocol

Since the methods need to be suitable for deployment on an embedded device, we introduce additional constraints that need to be considered during development and submission. The following criteria and rules need to be respected:

- *Static graph* – the neural network must have a defined static graph and exported to ONNX so that it can be successfully compiled for the target platform.
- *Supported operations* – target platform supports a limited set of supported operations, which we provide before the beginning of the challenge.
- *Standardized inputs* – since models are evaluated on the device, we request that models expect input images of fixed size (768×384) and must be normalized with ImageNet [18] mean and standard deviation. We only allow methods with a single image input.
- *Throughput* – achieved throughput on the embedded device must be ≥ 30 FPS.

Submitted methods are first quantized to the INT8 format on the validation set of LaRS [57] and compiled to a binary that can be executed on the target device. Before inference, all images are resized, centered, and padded to 768×384 shape with preserved aspect ratio, and the outputs are resized to the original image resolution. Then, the same evaluation protocol as in 4.2.1 is used to derive the final scores. In addition to the sea measures, the average throughput achieved on the embedded device is reported.

4.3.2 Submissions, Analysis and Trends

35 submissions from 3 different teams, including 8 baseline models from the MaCVi2024 committee were evaluated. We show the baseline models and the best submission from each team in Table 6. As stated in the challenge rules, only models that are faster than the predetermined threshold of 30 frames per second were considered for the final rankings.

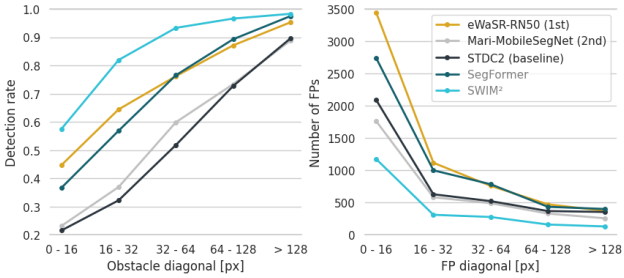


Figure 7: Detection rate and number of false positive predictions by diagonal size. We show the best three methods within FPS constraints and the best method regardless of the FPS from 4.3, and the best method from 4.2 for comparison.

The submitted methods explored various architecture changes and training regimes. eWaSR-RN50 (Section C.1) is based on eWaSR [42]. The authors fine-tuned various hyper-parameters, resulting in an increase of the batch size from 4 to 6, replacing the ResNet-18 [22] backbone with a heavier ResNet-50, and increasing the training duration. Furthermore, resizing all input images to a fixed size of 768×384 boosts performance, compared to center cropping during training.

Mari-MobileSegNet (Section C.3) utilizes the MobileNetV2 backbone [40], a typical architecture for mobile devices. The decoder employs the ASPP decoding head and is based on DeepLabV3 [14] architecture. FCNHead [34] is used as an auxiliary head during training.

Compared to the baselines (Section C.3) from the MaCVi2024 committee that satisfy the 30 FPS requirement, both, eWaSR-RN50 and Mari-MobileSegNet achieve higher throughput and Q score. While eWaSR-RN50 dominates in most metrics, Mari-MobileSegNet achieves higher precision ($+7.2\%$) at the cost of recall (-18.3%). Compared to the best baseline method SegFormer, which does not satisfy the FPS requirements, eWaSR-RN50 achieves only 0.5 lower Q score, while being almost $7\times$ faster on an embedded device.

Table 6: Overview of the submissions for the USV-based Embedded Obstacle Segmentation challenge. When determining the final placement of the teams, methods that achieve the required ≥ 30 FPS on the target device are considered (denoted in teal). For evaluation comparison, we include the winning method from the non-embedded challenge from Section 4.2 at the top, but do not consider it as part of the challenge. The best results from considered methods are denoted in bold.

Place	Institution	Method	Section	FPS	Q ↓	μ	Pr	Re	F1	mIoU
	UniCa	*SWIM²*	B.1	/	78.1	79.7	76.9	83.0	79.9	97.8
	UL	SegFormer (MiT-B2)	C.3	15.0	55.8	69.0	50.9	67.4	58.0	96.3
1	EAIC-UIT	eWaSR-RN50	C.1	**103.4**	**55.3**	**68.5**	48.5	**70.5**	**57.4**	**96.2**
	UL	DeepLabv3+ (RN-101)	C.3	16.6	54.9	68.7	56.2	58.6	57.4	95.7
	UL	FCN (RN-101)	C.3	16.8	51.5	68.6	53.5	54.1	53.8	95.6
2	DLMU	Mari-MobileSegNet	C.2	60.3	51.3	69.1	**55.7**	52.2	53.9	95.3
	UL	CN (RN-50)	C.3	19.7	50.9	68.0	53.1	55.0	54.0	94.2
	UL	STDC2	C.3	38.3	47.5	67.6	50.5	49.2	49.9	95.3
	UL	STDC1	C.3	45.8	45.1	66.8	47.2	48.5	47.9	94.3
	UL	BiSeNetv2	C.3	44.6	42.8	64.7	43.2	49.0	46.0	93.2
	UL	BiSeNetv1 (RN-50)	C.3	28.7	42.6	63.8	38.2	56.6	45.6	93.4

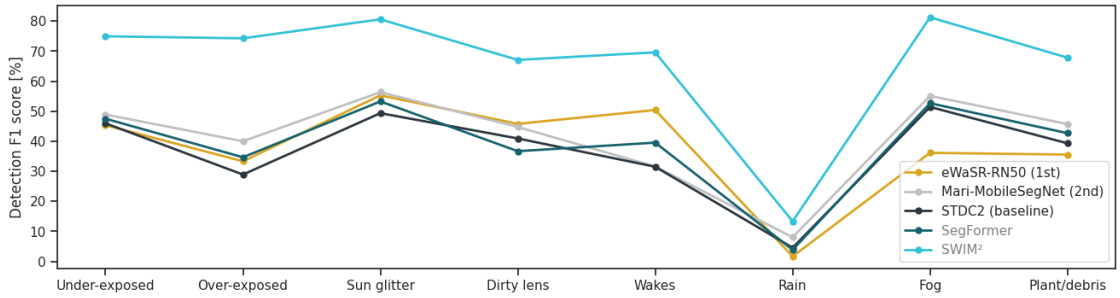


Figure 8: Detection F1 score under different conditions. We show the best three methods within FPS constraints and the best method regardless of the FPS from 4.3, and the best method from 4.2 for comparison.

We investigate how SegFormer, eWaSR-RN50, Mari-MobileSegNet, and the best baseline within FPS limits (STDC2) compare to the best method from the non-embedded segmentation challenge (SWIM^2) from Section 4.2. We observe that SWIM^2 is $+22.8$ Q better than eWaSR-RN50. This can be attributed to the smaller input shape that the embedded methods use at inference time, leading to poor performance on very small obstacles (see Figure 7). SWIM^2 archives almost $3\times$ better detection rate on obstacles with diagonal length between 0 and 16 pixels compared to Mari-MobileSegNet. This difference is less noticeable for SegFormer and Mari-MobileSegNet, but both produce more small-sized false positives.

In Figure 8, we show that the detection F1 score under different conditions follows a similar trend for the 5 analyzed models. While the detection performance gap is noticeable between the non-embedded method and the other embedded methods, all struggle in the presence of rain, but are most robust to the presence of sun glitter and fog. From the embedded methods, both SegFormer and eWaSR-RN50 perform significantly better in the presence of wakes, however the latter struggles the most in the fog compared to the others.

We show qualitative differences for the 5 methods in Figure 9. We can see that eWaSR-RN50 is the best embedded method for detecting small obstacles such as distant boats, but at the same time, it produces more false positive detections. SegFormer is better on ice and sky segmentation. Mari-MobileSegNet and STDC2 struggle the most with small obstacles. As expected, the non-embedded SWIM^2 performs the best on all examples.

4.3.3 Discussion and Challenge Winners

The overall winners of the USV-based Embedded Obstacle Segmentation are:

1st place: University of Information Technology, VNU-HCM (EAIC-UIT) with eWaSR-RN50, and

2nd place: Dalian Maritime University (DLMU) with Mari-MobileSegNet.

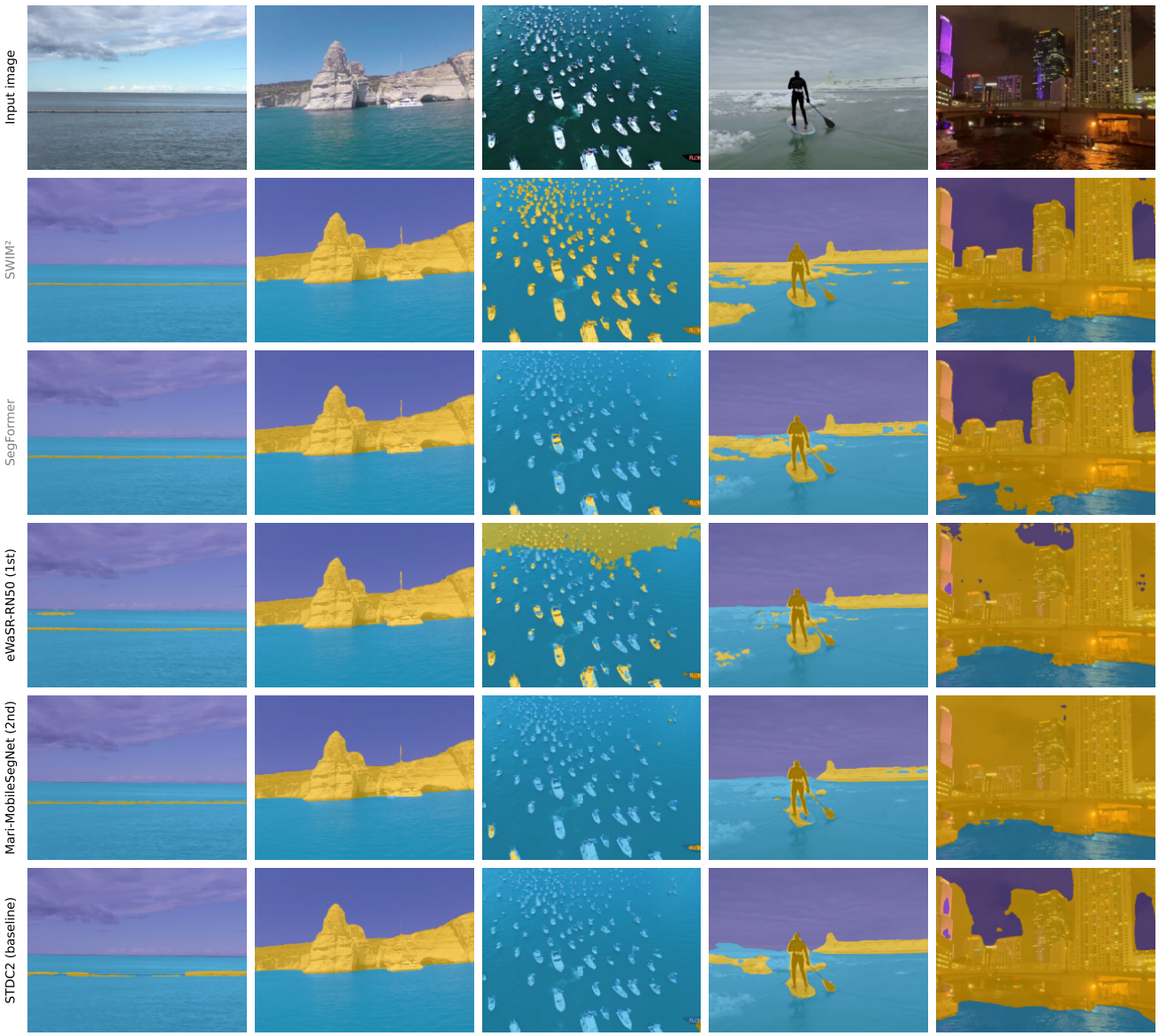


Figure 9: Qualitative comparison of methods for USV-based embedded obstacle segmentation.

The analysis indicates that the detection performance of the embedded-ready methods falls short from the non-embedded methods. This primarily stems from the fixed lower-resolution input, which makes it hard to achieve a good detection rate on small obstacles. However, the detection rate gap is significantly reduced for large obstacles.

4.4. USV-based Obstacle Detection Challenge

4.4.1 Dataset

This year's obstacle detection relies on the the newly released LaRS Dataset [57].LaRS is primarily segmentation dataset, however, with this challenge we recognize the fact that for early practical applications of USV collision avoidance and path planning, robust dynamic obstacle detection is beneficial, even if its pixel-wise accuracy is somewhat lacking.

In the Obstacle Detection track, the task was to develop an obstacle detection method capable of identifying obstacles in the input image and representing their location using rectangular bounding boxes. This was in accordance with the LaRS Dataset nomenclature, which identifies 8 classes of dynamic obstacles (boat, buoy, other, row boat, swimmer, animal, paddle board, float) and three classes of "stuff", referring to pixels in the image that do not correspond to any of the dynamic obstacle classes.

Table 7: Overview of the submissions for the USV Obstacle Detection challenge, with results. Ranking of the method on the leaderboard as well as the final placement of the teams are indicated. We also include the self-reported inference speeds, used hardware, and whether any other datasets were used in the training.

#	Author	Model Name	FPS	Hardware	Meta	F1
1	FER-LABUST, Croatia	YOLOv8x, pretrained	40	GTX 1080 TI	Yes	51.50
3	Fraunhofer IOSB	Co-DETR	4	A100	Yes	43.56
6	DLR MI-SIT	ScatYOLOv8CBAM+SAHI	4	A100		39.79
14	User #382	yolo_test2	0.1	1080		35.87
15	DLR MI-SIT	ModelSahi	1.5	A6000		34.21
18	mcprl	codetr_0.2	1	1080		22.13
20	Baseline UL	YOLO v7 baseline	10	1080 Ti		18.16

For the purpose of this challenge, all eight LaRS dynamic obstacle classes were treated as obstacles and given equal importance. A variety of processing backends can be utilized to infer the bounding boxes, with the stipulation that the results provided axis-aligned bounding boxes.

4.4.2 Evaluation protocol

The detection performance was evaluated using the standard Intersection over Union (IoU) metric for bounding box detections. A detection was classified as a true positive (TP) if it achieved an IoU score of at least 0.3 with the ground truth. Conversely, detections were considered false positives (FP) if they did not meet this threshold, unless 75% or more of the pixels in the submitted bounding box overlapped with the image area labeled as "static obstacle" as defined in the LaRS Dataset nomenclature. This approach effectively meant that false positive detections on land were not a concern. The final score was calculated as an average F1 score, derived from TP and FN metrics.

Additionally, every participant was required to submit information on the speed of their method, measured in frames per second (FPS). This also included an indication of the hardware used for benchmarking the speed. Lastly, participants were asked to disclose which datasets, including those used for pretraining, were utilized during the training process.

4.4.3 Submissions, Analysis and Trends

We received many submissions, the challenge was far more popular than MaCVi 2023. The top performing submissions are sorted in Table 7. Sorted by $F1$ metric, the top of the list is dominated by three teams, FER LABUST, Fraunhofer IOSB and DLR MI-SIT, who were invited to submit descriptions of their approach, which are provided in sections D.1, D.2 and D.3, respectively.

4.4.4 Discussion and Challenge Winners

The winners of the USV object detection challenge are as follows:

1st place: FER LABUST with YOLOv8x,

2nd place: Fraunhofer IOSB with PRBNet,

3rd place: DLR MI-SIT with ScatYOLOv8CBAM+SAHI

Important difference from last year is that Yolo-based method took the crown, helped only by additional training.

4.5. USV-based Multi-Object Tracking

Analogously to the SeaDronesSee-MOT challenge track, we evaluate the submissions on HOTA, MOTA, IDF1, MOTP, MT, ML, FP, FN, Recall, Precision, ID Switches, Frag [28, 36]. The determining metric for winning is HOTA, MOTA is the tiebreaker. We also required every participant to submit information on the computational runtime of their method measured in frames per second wall-clock time along their used hardware.

4.5.1 Submissions, Analysis and Trends

Despite the later release and start of the challenge, we received 23 submissions from 8 different teams. Interestingly, the performance in terms of the main metrics, HOTA, MOTA, and IDF1, is significantly worse compared to the UAV-based MOT with ReID challenge. We suspect this to be mainly due to the train-test-set domain gap as participants had to train on a dataset with likely significantly different characteristics. Also, the high number of objects per frame poses a difficulty (compare to Figure 4). Also see Figure 1 for a sample scene, showing that there are both, large foreground and small background objects.

Interestingly, ReIDTracker-Sea is in the top three of both competitions, UAV-based MOT with Reidentification and USV-based MOT, indicating its robustness to different domains.

Table 8: BoaTrack submissions overview.

Model name	Data	Detector	FPS	GPU
Detector Ensemble (UWIPL) (E.1)	COCO, LaRS	YOLOv8-x	6.6	V100
ReIDTracker-Sea (Lenovo) (E.2)	COCO, LaRS	Swin-Transformer	1	V100
DLR-BoaTrack (DLR) (E.3)	LaRS	ScatYOLOv8+CBAM	3	A100

Table 9: Multi-Object Tracking results on the BoaTrack test set. The submissions are ranked based on HOTA.

Model name	HOTA	MOTA	IDF1	MOTP	MT	ML	FP	FN	Re	Pr	IDs	Frag
D. Ens. (E.1	0.215	0.094	0.247	0.232	9	162	13231	80383	0.222	0.635	83	407
ReIDT. (E.2)	0.214	0.105	0.232	0.214	17	131	14476	76651	0.258	0.649	1404	4185
DLR (E.3)	0.193	0.057	0.202	-1.000	7	173	12275	85058	0.177	0.599	183	541

5. Conclusion

In this summary paper, we analyzed the challenges of the 2nd Workshop on Maritime Computer Vision. Specifically, MaCVi 2024 hosted one aerial and four different surface domain maritime challenges, some of which featured new exciting benchmarks for maritime vision.

The UAV-based Multi-Object Tracking with Reidentification challenge showed that there are still challenging conditions when there are fast camera movements or similar looking object that are hardly distinguishable from high altitudes, making reidentification difficult without considering the topology of the scene. Using the onboard's metadata, this can be incorporated and this year's winner, *MG-MOT*, did so. The second place *Tracking by Detection* performed almost equally well with a well-known tracking framework, indicating that clean tuning of existing methods is still very promising as well. The third place, *ReIDTracker-Sea* leverages a Transformer that has a good detection capability, but their method is missing a motion model, which likely has a negative impact on association accuracy.

The USV-based challenges featured two new datasets, LaRS and BoaTrack, each with unique challenges. Winners of the *USV Obstacle Segmentation Challenge* were (1st) SWIM^2 from University of Cagliari (2nd) TransMari from Hong Kong University of Science and Technology, and (3rd) Mari-Mask2Former from Dalian Maritime University. All methods significantly outperformed the baseline, dramatically improving on issues such as night scenes and thin structures, with SWIM^2 becoming the new state-of-the-art for semantic segmentation on LaRS.

MaCVi 2024 also hosted an embedded segmentation challenge for the first time, with the goal of developing and assessing methods for obstacle segmentation suitable for use on real world embedded hardware. Winners of the *USV-based Embedded Obstacle Segmentation Challenge* were (1st) eWaSR-RN50 University of Information Technology, and (2nd) Mari-MobileSegNet from Dalian Maritime University. Both methods achieve a high Q score compared to other embedded baseline models while running faster than 60 FPS on an embedded device, while the 1st method boasts superior detection of small obstacles. However, we still observe a large performance gap on all methods compared to state-of-the-art methods for obstacle segmentation and more effort needs to be invested into developing robust embedded-ready segmentation methods.

MaCVi hosted USV object detection as well, it was simplified and used new LaRS dataset in object detection scenario. Important difference from last year is that Yolo-based method took the crown, helped only by additional training.

The accuracy for the USV-based Multi-Object Tracking challenge is still very low compared to other MOT benchmarks. We believe this to be credited to the crowded nature of harbor scenes as indicated by the low precision and recall values. The winner, *Detection Ensembe* employed an ensemble model for higher performance. The second place, *ReIDTracker-Sea* performs well in this and the UAV-based competition, while the third place *DLR-BoaTrack* leverages an improved YOLOv8 model.

In summary, the outcomes of the MaCVi 2024 challenges underscore the advancing maturity of the maritime computer vision field. The increasing resilience of methods to prevalent issues like reflection reflects significant progress. However, real-world hardware limitations remain a notable constraint, revealing performance gap of embedded methods. Developing efficient methods for addressing this challenge is becoming an important frontier for enabling real world applications of autonomous maritime technology.

Acknowledgments. This work was supported by Conservation, Protection and Use joint call, Slovenian Research Agency (ARRS) project J2-2506 and programs P2-0214 and P2-0095, Sentient Vision Systems for sponsoring prizes for the UAV-based Multi-Object Tracking challenge, and LOOKOUT for providing testing videos for the USV-based Multi-Object Tracking challenge.

Appendix - Submitted Methods

A. UAV-based Tracking

A.1. MG-MOT: Metadata-Guided Long-Term Re-Identification for UAV-Based Multi-Object Tracking

Cheng-Yen Yang, Hsiang-Wei Huang, Zhongyu Jiang, Heng-Cheng Kuo, Jie Mei, Jenq-Neng Hwang
cycyang@uw.edu

1. Method: Metadata-Guided MOT Our proposed method Metadata-Guided MOT (MG-MOT) takes the metadata provided along with the data to do the long-term re-identification to recover and resulted in a much improved IDF1 of the MOT result. Most details can be found in our paper [48] and please feel free to use any of the figures in our work in the challenge report!
2. Dataset: We used the entire training and validation dataset of SeaDroneSee-MOT [44] to train our multi-class detector with the pretrained weight on COCO. Metadata of the training, validation, and testing are also used as a source to select a set of reasonable thresholds for our algorithm.
3. Environment: Our codebase is derived from ultralytics [2]. We trained our detector using 8 Nvidia V100 GPUs and the inference step including predicting the bounding boxes and online tracking is done on a single Nvidia GV100 GPU and i7- 10700K CPU @ 3.80GHz. The rough estimate of the latency is 13 FPS and given the tracks of the dataset are not much, the post-processing time is neglectable compared to the remaining latency.
4. Additional Findings & Thoughts: We tried using the altitude to fine-tune our detector but turned out not helping too much. We also found some problems in the dataset (e.g., annotations and metadata): (1) There exists a certain amount of ID labeling error in training and validation despite the tracks not leaving the image at all. (2) There exists some synchronization problems and some data errors in the metadata of the drone. It is also quite surprising that there are still great num- bers of FP and FN on the testing data, it will be nice if the organizer can visualize some of our and other participants results to show the error. *Acknowledged by the authors of the dataset. Hope this is resolved with this paper or the actual workshop event.*

A.2. Tracking-by-Detection (TBD)

Daniel Stadler, Lars Sommer
{daniel.stadler, lars.sommer}@iosb.fraunhofer.de

Tracking-by-Detection (TBD)
Eponym of our approach is the *tracking-by-detection* (TBD) paradigm which separates the multi-object tracking task into two sub-tasks: detection and tracking. Details of the applied methods for detection, tracking, and additional post-processing are given in the following.

Detector: To generate our detections, we used VarifocalNet [52] and ResNet-50 [23] as backbone architecture. For initialization, we used weights pre-trained on MS COCO [30]. SGD was used as optimizer with an initial learning rate of 0.02, a momentum of 0.9, and a weight decay of 0.0001. The model was trained for 12 epochs. We employed the SeaDronesSee Object Detection v2 train and validation set as training data. For images with dimensions less than 3840x2160 pixels, we used multiple scales (1920x1080, 2376x1296, 2688x1512, and 3360x1890). Otherwise, we set the input scale to 3360x1890 pixels. For inference, we applied multi-scale testing (2688x1512, 3360x1890, and 4032x2268). We considered all five classes during training and inference. The implementation provided by MMDetection [12] – an open source object detection toolbox based on PyTorch – was used to train our detector. We used 2 Tesla V100 GPUs (CPU: Intel Xeon E5-2698 v4 @ 2.20GHz). The inference speed of the detector was about 1 FPS.

Tracker: We followed a two-stage IoU-based association scheme in which first, high-confident detections are matched to all tracks, and second, low-confident detections are matched to the remaining unassigned tracks similar as in [54]. To prevent the start of FP tracks, detection boxes with aspect ratio larger than 5 were removed. Furthermore, an occlusion-aware initialization method was applied that filters detections with high overlaps to already tracked targets. We used the NSA Kalman filter [19] as motion model. Moreover, ORB features [38] were extracted and matched in consecutive frames to estimate transformation matrices that were used for camera motion compensation.

Post-processing: The resulting tracks were linearly interpolated and short tracks with less than 13 detections were removed. To merge tracks of the same object that couldn't be tracked at once because the object temporarily left the scene, we developed a hierarchical clustering based on appearance features extracted with the model from [3]. For each track, a mean feature vector was computed with the appearance features extracted from the tracks' detections. The cosine similarity between all tracks was calculated and two tracks were merged if their similarity score was above a threshold, where most similar tracks were merged first. Also, temporal constraints were enforced to prevent infeasible merges. The overall pipeline inference speed was less than 1 FPS.

A.3. ReIDTracker Sea: BoaTrack & SeaDronesSee

Kaer Huang, Aiguo Zheng, Weitu Chong, Kanokphan Lertniphonphan, Jun Xie, Feng Chen, Jian Li, Zhepeng Wang
{huangke1, zhengag, klertniphonp, xiejun, chenfeng13, lijian30, wangzpb}@lenovo.com, wtzhong22@m.fudan.edu.cn

Our maritime Multi-Object Tracking (MOT) solution for UAVs and USVs simplifies traditional complex MOT systems by using unsupervised learning with self-supervision on ImageNet and high-quality detectors. This approach, less reliant on costly video annotations, achieved top performances in major UAV and USV-based MOT competitions.

In our introduction, we explore achieving state-of-the-art (SOTA) results in Multi-Object Tracking (MOT) using only high-performance detection and appearance models. Our approach employs CBNetV2 Swin-B [29] for detection and MoCo-v2 [21] for a self-supervised appearance model. We omit traditional motion information like the Kalman filter and IoU mapping and utilize ByteTrack [54] to associate detection boxes of varying scores. For UAV datasets with low object overlap, we adjusted the NMS threshold accordingly.

Our MOT framework includes Detection, Appearance Model, and Data Association (see Fig. 10). The architecture comprises detection (providing high-quality instance boxes), an appearance model (offering quality embedding features), and a data association tracker that stabilizes trajectory outputs. The detection strategy uses a Swin-based transformer backbone [33] with CBNetV2 architecture, integrating multiple backbone features. The CBNetV2 integrates high and low-level features of multiple backbones which connected in parallel. The Feature Pyramid Network (FPN) [31] neck and Hybrid Task Cascade (HTC) [10] detector are attached and trained in each backbone as a main branch and an assistant branch. Only the main branch is used in the inference process. We use more weight to bound box regression than classification in Loss Function for a more compact detection box which will benefit appearance model performance.

We use unsupervised appearance models to address the high cost of video trajectory annotation. Our base appearance model for this framework is MoCo-v2 with ResNet50. The model extracts feature representations from detected boxes. MoCo-v2 model training by imagenet 1K dataset and then finetuning on MOT dataset. We also compare with model training by other contrastive learning methods (SimCLR, SimCLRv2 [15], MoCo-v2, etc). We also make a comparison between supervised learning and self-supervised learning - we draw the conclusion that MoCo-v2 has better generalization capacity in the maritime dataset. Because the last convolution module of ResNet50 is more related to classification type, not the general features we want, we finally removed the network in the final integration (Fig. 11).

We adopt the Bytetrack concept, a simple but strong method for **matching object id across frames**. The detected boxes in each frame are grouped based on their detection score into the high score and low score. Firstly, the method finds the association between the high score box and the tracklet. Then, the rest of the high score and low score boxes are used to find the association from the remained tracklet. The association method can be different in each association step. Our method uses only the appearance feature to associate both high and low score boxes with tracklet. In addition, we add a weighted score to tracklet to keep the tracklet representation from the higher detection score since the detection score tends to get lower when the occluded part gets bigger. The tracklet features are weighted by the detection score and combined within τ frames to maintain the object representation during occlusion. The weighted feature $\hat{e}_j$ combined tracklet feature e_j which is weighted by the detection score s_j from the previous τ frames.

$$\hat{e}_j = \frac{\sum_{t=1}^{\tau} e_j^t \times s_j^t}{\sum_{t=1}^{\tau} s_j^t}$$

$\hat{e}_j$ is further used for finding the matched box in the data association. We apply the same association method with [45]. A ReId similarity matrix between tracklet and detection box is computed and used to find matching pairs by the Hungarian algorithm [27].

The Swin-B backbone was initiated by a model pre-trained on ImageNet-22K. CBNetV2 was trained on the SeaDronesSee-MOT train and val dataset. We applied multi-scale augmentation to scale the shortest side of images to between 640 and 1280 pixels and applied random flip augmentation during training. Adam optimizer was set with an initial learning rate of 1e-6 and weight decay of 0.05. We trained the model on 4 A100 GPUs with 1 image per GPU for 10 epochs. During inference, we resize an image to 2880x1920 to better detect the small objects. For the detection task, we use a combination of classification Cross-Entropy loss and the generalized IoU regression loss [37]. Loss weights λ_1 and λ_2 are set to 1.0 and 10.0 by default, which drives the model output more compact box

$$\mathcal{L} = \lambda_1 \mathcal{L}_{cls} + \lambda_2 \mathcal{L}_{box}$$

The backbone of the appearance model is pre-trained on ImageNet-1K. Then, we fine-tuned the backbone by using MoCo-v2 on the SeaDronesSeeMOT dataset. The training dataset contains cropped object images according to bounding box labels from MOT dataset. The optimizer is SGD with a weight decay of 1e-4, a momentum factor of 0.9, and an initial learning rate of 0.12. We trained the model on 4 A100 GPUs with 256 images per GPU. Our method is generally similar to ByteTrack, but we used ReID to match high and low detection boxes. We set the high detection score threshold to 0.84 and the low detection score threshold to 0.3. for both challenges (SeaDroneSee-MOT and BoaTrack), We use the sample ReID module which training on ImageNet 1K. **SeaDronesSee-MOT with Reid:** We just train the detector using the SeaDronesSee-MOT train and val set. but we grouped "swimmer" and "swimmer with life jacket" and "life jacket" as one class. **BoaTrack:** we just train the detector using LaRS train set on "boat" labels.

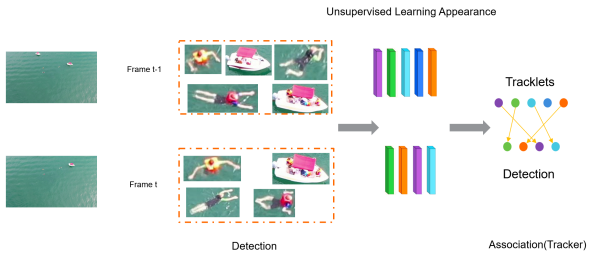


Figure 10: Overall architecture of ReIDTracker Sea.

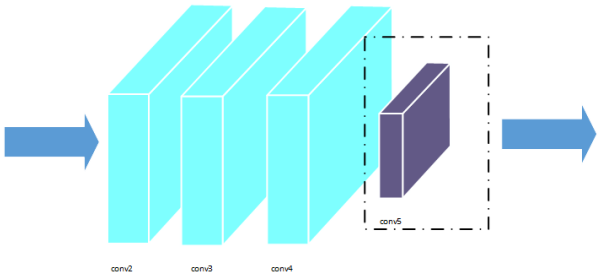


Figure 11: Appearance model of ReIDTracker Sea.

A.4. Tracktor Baseline

MaCVi Organizers

The baseline is the last as in the last workshop iteration [25]. We provided a Tracktor-based tracker using Camera Motion Compensation with a Faster R-CNN ResNet-50 detector. We used the tracking implementations from mmdetection [12] with default hyperparameters.

B. USV-based Obstacle Segmentation

B.1. ① SWIM^2

Luca Zedda, Andrea Loddo, Cecilia Di Ruberto
{luca.zedda, andrea.loddo, cecilia.dir}@unica.it,
Department of Mathematics and Computer Science, University of Cagliari

Contributions: Conceptualization: LZ, AL, CDR; Implementation: LZ; Experiments and; Analysis: LZ; Supervision: AL, CDR

The submissions related to this report are known as Snarciv3, Snarciv2, and Snarci, with Snarciv3 being the 1^{st} place winner of the competition. All architectures share the same backbone and model structure but vary in size and number of parameters. This technical report pertains to the winning solution: Snarciv3.

Our method is named SWIM^2, which stands for SWIn-based Maritime Mask2former. Our approach is based on the Mask2Former architecture [16]. We adapted the training strategy to use images of size 768x1536 to improve the recognition of small objects such as small boats and to better segment and recognize challenging objects found in real-life scenarios, such as those in the challenge dataset (e.g., algae and weeds growing on the borders of canals). We employed the Mask2Former base size model, which uses Swin as its backbone. For this downstream task, we chose to fine-tune the model that was pretrained on the Cityscapes dataset with a size of 512×1024. The original configuration file can be accessed at the following Link.

The preprocessing and augmentation pipeline includes the following steps: random resizing, cropping, flipping, and photometric distortion. These steps enhance data diversity and prepare it for robust model training. More details are available at the following link: Link.

Additionally, we trained the model on a single Nvidia RTX 3060 with 12GB VRAM GPU, which led us to limit our batch size to 1. We trained our model for 90,000 iterations using a PolyLR scheduler and saved the best model based on the mIOU metric every 2000 iterations. The training took approximately 2 days, and the inference time averaged 3.1 images per second.

Our team primarily works with high-resolution images, specifically blood-smear images. These images share common issues with maritime images, such as the low pixel count of small parasitized cells (e.g., malaria-infected RBC) and their placement within low-entropy backgrounds. In addition, we also encounter variations in acquisition equipment, lighting conditions, factors such as object deformation, blurred boundaries, and color differences. These additional issues can make image analysis and processing even more challenging.

B.2. ② TransMari

Tuan-Anh Vu, Hai Nguyen-Truong, Tan-Sang Ha, Quan-Dung Pham, Sai-Kit Yeung
{tavu, thnguyenab, tsha, qdpham}@connect.ust.hk, saikit@ust.hk,
The Hong Kong University of Science and Technology

Contributions: Conceptualization: TAVu, NTHai; Implementation: TAVu, TSHa; Experiments and Analysis: TAVu, QDPham; Supervision: SKYeung

The proposed maritime obstacle segmentation is based on the transformer-based architecture, namely Mask2Former [16]. Mask2Former utilizes the identical meta-architecture as MaskFormer, except it substitutes the standard decoder with the new Transformer decoder with masked attention operator. It functions to extract localized features by restricting cross-attention to the foreground region of the predicted mask for each query rather than focusing on the entire feature map. A multi-scale approach is suggested as an effective means

of managing small objects by leveraging high-resolution characteristics. Finally, they integrate optimization enhancements that augment the model's efficacy while preventing the need for additional computation. Our implementation based on MMSegmentation. The training configuration is adopted from the setting for ADE20K dataset [55] with pretrained Swin-L backbone on ImageNet22k dataset [39]. The image resolution is set to 512×512, and the training schedule is set to "schedule 80k". The optimizer is the AdamW optimizer, and the learning rate is 0.0001 with a momentum of 0.05. The data augmentation includes random crop and resize, random flipping, photometric distortion, noise transform, and normalization. We only use the LaRS dataset [57] for training, and no additional images are used. The training device is NVIDIA RTX 3090 with 24G memory. The inference speed is about 5.2 frames per second under the original image resolution of the dataset. The best submission of this method with test time augmentation achieved an average score of 77.8% in the Q metric.

B.3. ③ Mari-Mask2Former

Yuan Feng
fengyuan@dlmu.edu.cn,
Dalian Maritime University

Our method is named Mari-Mask2former and utilizes the relevant configuration files provided by MMSegmentation. The utilized model for USV obstacle segmentation is based on the transformer-based neural architecture Mask2former, which has shown outstanding performance in panoramic segmentation tasks. Inspired by this, we adapted the model in MMSegmentation by configuring it specifically for obstacle segmentation. The training configuration file is named `mask2former_swin-l-in22k-384×384-pre_8xb2--160k_ade20k-640×640` and it employs the Swin Transformer-Large as the backbone network.
Backbone: Swin-L pretrained on ImageNet-22k dataset with a resolution of 384×384
Decode head: Mask2Former Head

Training

Batchsize: During the training process, the batch size used was 2.
Optimizer: Adamw, learning rate: 0.0001, eps: 1e-8, decay: 0.05, betas: (0.9 $\sim$ 0.999)
Schedule: We adopted a warm-up strategy and the "poly" learning rate scheduler during training. The training process lasted for a total of 160,000 iterations, starting from iteration 0 and ending at iteration 160,000.
Augmentations: including random flipping, random cropping, random brightness adjustment, and random saturation adjustment, among others.
Loss functions: We employed the following three loss functions: loss_cls, loss_dice, and loss_mask. For loss_cls, we used CrossEntropyLoss with a class weight of [1.0, 1.0, 1.0, 0.1], a loss weight of 2.0, a mean reduction and no sigmoid activation. For loss_dice, we used DiceLoss with an activated True, an epsilon value of 1.0, a loss weight of 5.0, a naive_dice calculation, a mean reduction, and sigmoid activation. Finally, for loss_mask, we used CrossEntropyLoss with a loss weight of 5.0, a mean reduction, and sigmoid activation.
Datasets used: Backbone pretraining - ImageNet22k, Model pretraining - ADE20K 640×640, Finetuning - LaRs train
Hardware: The training device is NVIDIA RTX 3090 with 24G memory
Inference Speed: We tested LaRS's validation set using the code provided by MMSegmentation and obtained an average inference speed of 8.29 images per second.

Observations

During training, we set up validation every 2400 iterations and output the Intersection over Union (IoU) and accuracy for each class. Upon analyzing all the validation results, we observed that the class of obstacles exhibited slow growth in both of these metrics. We hypothesized that this may be due to an imbalance in the distribution of pixels across obstacles, sky, and water in the images. Obstacles occupy fewer pixels, with the majority of them being located in the foreground of the images.

Inspired by this finding, we considered employing Dice loss to address the issue of sample imbalance. Dice loss effectively leverages foreground information. Additionally, we can also explore assigning higher weights to the obstacle class when using the cross-entropy loss function, which would allow the network to focus more on difficult-to-classify samples. However, it is important to be mindful that convergence issues, such as slow convergence or difficulty in convergence, may arise when using multiple loss functions simultaneously.

When using the Mask2former method, we observed that the timestamp in the testing images was recognized as an obstacle. We also tried using Deelplabv3+ with an R101 backbone network and found the same phenomenon. However, we did not observe this issue when using the Segformer method. Intuitively, algorithms that classify timestamps as obstacles may not perform well. Surprisingly, the predictions from Mask2former were more accurate than Segformer and Deelplabv3+. Although we have not conducted an extensive investigation into this matter, we find this observation to be quite interesting.

B.4. Baselines

MaCVi Organizers

We provided two baselines for the challenge, DeepLabv3 [13] with the ResNet-101 [22] backbone and K-Net [53] with the Swin-T [33] backbone. We use the model implementations in the MMSegmentation framework [17]. The methods were trained for 80k iterations on 2× NVIDIA V100 GPUs with a batch size of 8. We employ random resizing, flipping, photo-metric distortions for image augmentation. We apply random cropping with 1024×512 crop size during training. During inference all input images are scaled to the 2048×1024 resolution.

C. USV-based Embedded Obstacle Segmentation

C.1. ① eWaSR-RN50

Nguyen Thanh Thien
thiennt@uit.edu.vn
University of Information Technology

This report describes our solution for the USV Embedded Obstacle Segmentation Challenge. We achieve first place with 57.4 F1 and 96.2 mIoU, which results in a Q (Quality) score of 55.3 on the leaderboard.

Method: Our submission is based on eWaSR [42]. Its architecture includes three parts: backbone, decoder, and segmentation head. The backbone is the only component that we change: the original backbone ResNet-18 [22] is replaced by larger version ResNet-50. With the new backbone, features needed for the decoder must be changed. Specially, features from layers 10, 19, 31 and 49, which are selected because their output features have the same size as output from layers 6, 10, 14, 18 of ResNet-18 backbone, are used.

Training: eWaSR [42] official repository is used for training and exporting the model. We trained the model for 100 epochs with batch size 6. Albumentations [7] is applied to resize training images to 384x768 (see more explaination in Observations section below). Other settings are kept as default.

Datasets: We use LaRS dataset [57] for training, no other dataset is utilized.

Hardware: A P100 16GB GPU is used for training.

Observations:

- Longer training improves the model's performance. The original eWaSR model (with ResNet-18 backbone) achieves 41.6 Q with only 25 epochs of training. The score increases to 46.6 and 47.1 Q with 50 and 100 epochs of training, respectively.
- Larger backbone enhances the result. With 100 epochs of training, model with ResNet-50 backbone obtains the best result with 54.1 Q, while model with ResNet-34 backbone is slightly worser with 52.8 Q. Although using large backbone reduces the inference speed (from 117.3 to 103.8 FPS), in this case, it is very worthwhile because the result is still good compared to the required threshold (30 FPS).
- Augmentations may give an undetermined impact. For most of submissions, our team resizes the training images by combining two Albumentations [7] transforms: LongestMaxSize and CenterCrop. Using this combination, we get the result 54.1 Q with ResNet-50 backbone. Later, we use normal Resize transform to resize all training images to 384x768 without the two transforms above. That leads to an improved score (55.3 Q), which is the final score of our team.

C.2. ② Mari-MobileSegNet

Yuan Feng, Lixin Tian
{fengyuan, 112023116tlx}@dlmu.edu.cn
Dalian Maritime University

Contributions: Conceptualization: YF; Implementation: LT; Experiments and Analysis: YF; Supervision: YF

Method: MobileNetV2 [40] is a novel mobile architecture that significantly improves the performance of mobile models across various tasks and model sizes. It introduces an inverted residual structure and lightweight depthwise convolutions to enhance model representation capabilities and reduce computational complexity. The approach emphasizes the importance of removing non-linearities in narrow layers and decoupling input/output domains from the expressiveness of the transformation. The submitted method utilizes a configuration file named *mobilenet-v2-d8_deeplabv3_4xb4-160k_ade20k-512x512* from the MMSegmentation [17] code repository. In this configuration, the encoder utilizes MobileNetV2, while the decoder employs the ASPP decoding head from the DeeplabV3 [14] architecture.

Backbone: MobileNetV2 [40] pretrained on ImageNet-1k dataset with a resolution of 256x256

Decode head: ASPPHead [14]

Auxiliary head: FCNHead [34]

Training:

- BatchSize: During the training process, the batch size used was 4.
- Optimizer: SGD, learning rate: 0.01, momentum: 0.9, decay: 0.0005.
- Schedule: We adopted a warm-up strategy and the "poly" learning rate scheduler during training. The training process lasted for a total of 320,000 iterations, starting from iteration 1500 and ending at iteration 320,000.
- Augmentations: including random flipping, random cropping, random brightness adjustment, and random saturation adjustment, among others.

- Loss Functions: In this method, only the cross-entropy loss function is utilized.

Datasets used:

- Backbone pretraining: ImageNet-1k
- Model pretraining: ADE20K 512X512
- Finetuning: LaRS train

Hardware: The training device is NVIDIA RTX 2080Ti with 11G memory
Observations: In order to reduce computational costs and enable deployment on embedded devices, we utilized a lightweight backbone network, MobileNetV2 [40], which is consistent with the approach taken in eWaSR [42]. The substitution of a more lightweight backbone network has effectively resolved computational cost issues, however, it may negatively impact detection performance, and strike a balance between detection performance and real-time processing is necessary when building a segmentation network. Additionally, during our attempts to export ONNX formats using other methods, we encountered errors caused by unsupported operators within the Pytorch-related library, such as global average pooling. Inspired by this, we explored the replacement of computationally intensive modules in the network with equivalent, simplified, and more efficient operators to also reduce computational costs when converting to ONNX formats.

C.3. Baseline methods

MaCVi Organizers

We train the SegFormer [46] with MiT-B2 backbone, DeepLabv3+ [14] and FCN [34] with ResNet-101 [22] backbones, FCN [34] and BiSeNetv1 [50] with ResNet-50 [22] backbones, STDC2 [20], STDC1 [20], and BiSeNetv2 [49] models on LaRS [57] dataset. We use MMSegmentation [17] framework with default settings for training.

D. USV-based Obstacle Detection

D.1. YOLOv8 Extra Large

Matej Fabijanić, Magdalena Šimunec, Nadir Kapetanović
{MatejFabijanic, MagdalenaSimunec}@fer.hr
University of Zagreb, Faculty of Electrical Engineering and Computing
Contributions: Conceptualization: MF, NK; Implementation: MF, MŠ; Experiments and Analysis: MF, MŠ, NK (analysis); Supervision: NK

The neural network architecture used for the challenge of obstacle detection was"YOLOv8 Extra Large" model [2] with 68.2 million parameters. We have not made any adaptations to the architecture, seeing how we decided to just test achievable detection accuracy for the architecture as-is before trying to make changes in order to enhance the model for the specific use-case of detecting objects on the surface of water.

The architecture and its documentation can be found at [2]. We have used the default training parameters found at [43], with the exceptions of lowering the number of epochs to wait for no observable improvement for early stopping of training from 50 to 15 due to long training time, and lowering the number of images per batch from 16 to 8 due to experienced memory errors when using a higher batch size. We have not made any augmentations to the images.

Datasets Used for Pretraining

Datasets used for pretraining [dataset name (train images/test images)]: Open Images Dataset V7 (15981/4019), COCO 2017 (621/155), CalTech256 + Boat Types Recognition dataset from Kaggle (376/93)[2], Šibenik Croatia 2023 CCTV footage (2034/299) [3].

For the Open Images dataset, we have used a randomly selected subset of 20000 images containing boats. For the COCO dataset, we have used a hand-picked dataset of 776 images containing boats which we deemed were useful for the purpose of the challenge. From CalTech256 + Boat Types Recognition dataset from Kaggle, we hand-picked and manually annotated 469 images of different boats, trying to have row boats and other classes also present.

Machine Specifications Used for Training

CPU: Intel i9-9900K @3.60 GHz, GPU: NVIDIA GeForce GTX 1080Ti 11GB, RAM: 64 GB.
Inference Speed: 34ms for 2208x1242 pixels

D.2. Co-DETR

[1]*Andreas Michel,* [1]*Wolfgang Gross,* [2]*Martin Weinmann*
{andreas.michel,wolfgang.gross}@iosb.fraunhofer.de, martin.weinmann@kit.edu
[1]*Fraunhofer IOSB,* [2]*Karlsruhe Institute of Technology*
Contributions: Conceptualization: AM; Implementation: AM,WG; Experiments and Analysis: AM,MW; Supervision: WG,MW

The used detection pipeline is based on the Co-DETR [56] object detector. Co-DETR exploits multiple parallel auxiliary heads in order to increase the learning effectiveness of detection transformer. In our case, the underlying detector is DINO [51] with six encoder- and decoder-layers with sinusoidal positional encoding. We used the implementation

[2]https://mega.nz/file/UNFllTII#fCood0Zd1QyfxQJOmvhycii2dx0mt-QZdanDaZhCr74
[3]https://mega.nz/file/wAk3zbyK#YBz-njyY5v8QTs93aId5CtXDblPqkUmbO8xCAET_4fw

provided in the MMDetection toolbox version 3.2 [11] to train our detector. As a backbone, we utilize an ImageNet-22K [18] pre-trained Swin-L [33] vision transformer. Our training is conducted on two NVIDIA A100 80GB graphics cards. The detector is pre-trained on the Objects365 [41] and the MS COCO [32] datasets. In the next stage, we combine all images from MS COCO, including the category ship, with the LARS [57] dataset and train for twelve epochs. The last training stage includes only the LARS dataset with the dynamic obstacle class and is only two epochs long. While training, we apply data augmentation in the form of random vertical image flipping, random multi-scale resizing, and random crop operations. AdamW [35] was used as an optimizer with an initial learning rate of 0.0002 and a weight decay of 0.0001. Furthermore, we applied gradient clipping and apply a 0.1 learning rate multiplier to the backbone. Testing was performed on an NVIDIA A100 80GB graphics card. The inference speed was about four FPS. The test pipeline includes resizing to a 2048x1280 scale but no test time augmentation (TTA). In our experiments, TTA with multi-scale resizing leads not only to an expected large decrease of 3 FPS but also to an unexpected loss in detection performance. Furthermore, skipping the second training stage and training directly on the LARS dataset for twelve epochs decreases the F1 score by over three points on the test dataset. Varying the confidence score threshold also caused a large fluctuation in detection performance. The empirically determined score threshold of 0.25 results in the best F1 score for the chosen method.

D.3. ScatYOLOv8+CBAM

Borja Carrillo-Perez, Alexander Klein, Antje Alex, Edgardo Solano-Carrillo, Felix Sattler, Yannik Steiniger, Angel Bueno Rodriguez
{Borja.CarrilloPerez, Alexander.Klein, Antje.Alex, Edgardo.SolanoCarrillo, Yannik.Steiniger, Angel.Bueno}@dlr.de
German Aerospace Center (DLR), Institute for the Protection of Maritime Infrastructures, Bremerhaven, Germany
Contributions: Conceptualization: BCP, AK, AA, ESC, FS, YS, ABR; Implementation: BCP, AK; Experiments and Analysis: BCP, AK; Supervision: BCP, AK, AA, ESC, ABR

For the USV-based Obstacle Detection challenge, we chose the ScatYOLOv8+CBAM architecture described in [8]. This object detector improves ship recognition in maritime contexts by synergizing their global and local features for a better perceptibility, with more prominent delineation (by scattering transform) and leveraging their location (by attention mechanism). We combined the detector with Slicing Aided Hyper Inference (SAHI) [6] for data augmentation in training and inference for an improved small object detection.

The implementation of ScatYOLOv8+CBAM, compared to the standard YOLOv8 [2], adds the ScatBlock, a 2D scattering-transform-based block at the beginning backbone and Convolutional Block Attention Modules (CBAM) to the prediction heads.

In the submission we adopted YOLOv8x, the largest YOLOv8, as it provided the best results during the preselection, and to which we added an extra prediction head to conform ScatYOLOv8x+CBAM+P.

Prior to training, we used the procedure described in the SAHI method [6] for data augmentation. We created the new dataset, based on LaRS dataset [57], with slices of size 640×640 pixels with 20% overlap. The augmented training and validation set contain 19913 and 1597 sliced images respectively.

We trained the model, with random weight initialization, on the augmented dataset for 40 epochs using the default YOLOv8 training parameters. The $F1$ score achieved by our method on the test set is 39.79. The SAHI parameters during inference include 640×640 pixel slices with no overlap, Intersection over Union (IoU) as post-process match metric with 0.3 threshold, and a confidence score threshold of 0.1. The training, validation and test of our approach use a single NVIDIA A100 GPU (CPU AMD EPYC 7713 64-Core Processor).

E. USV-based Tracking

E.1. Detectors Ensemble

Hsiang-Wei Huang, Cheng-Yen Yang, Zhongyu Jiang, Sheng-Yao Kuan, Yuan-Hao Ho, Jenq-Neng Hwang
University of Washington (UWIPL)
{*hwhuang, cycyang, zyjiang, shengyao, yuanhh2, hwang*}*@uw.edu*

We use YOLOv8x coco-pretrained weight and further trained on Lars dataset (https://lojzezust.github.io/lars-dataset/) for 100 epochs with 0.001 learning rate, all the other training parameters follow the default setting of YOLOv8 repository (https://github.com/ultralytics/ultralytics). Since only objects that are boat class in the test set are annotated, only images of "boat" class from Lars dataset are used for training. We run BoTSORT on the test set following the default tracking parameter of Ultralytics BoTSORT implementation. We do not record the FPS but according to the original paper of BoTSORT, the FPS is roughly 6.6. The Lars-trained YOLOv8 + BoTSORT serves as our baseline and resulted in a HOTA of 0.191.

We found that the model trained on Lars dataset cannot accurately detect those boats when the boats are close to the USV, this is mainly because the boats in Lars dataset are relatively smaller, while some of the boats in the BoaTrack

test set is much larger. To overcome this, we found that using COCO pre-trained weights can yield better results for those boats that are closer, so we implemented the Lars-trained YOLOv8x for all the test video except for sequence 75 after 3450 frames, when the boats in the video are closer and larger, we use the detection from COCO pre-trained YOLOv8x to conduct tracking. This detectors ensemble trick resulted in 0.215 in HOTA, which is the 1st place of the USV Multi-object tracking competition.

All the experiments are conducted on one V100.

E.2. ReIDTracker Sea: BoaTrack & SeaDronesSee

Kaer Huang, Aiguo Zheng, Weitu Chong, Kanokphan Lertniphonphan, Jun Xie, Feng Chen, Jian Li, Zhepeng Wang
{huangke1, zhengag, klertniphonp, xiejun, chenfeng13, lijian30, wangzpb}@lenovo.com, wtzhong22@m.fudan.edu.cn
See Section A.3.

E.3. DLR-BoaTrack

Angel Bueno Rodriguez, Borja Carrillo-Perez, Alexander Klein, Antje Alex, Yannik Steiniger, Felix Sattler, Edgardo Solano-Carrillo
angel.bueno, borja.carrilloperez, alexander.klein, antje.alex, yannik.steiniger, felix.sattler, edgardo.solanocarrillo@dlr.de

Our methodology employs the tracking-by-detection paradigm: an object detection model identifies objects, and a multiple object tracking algorithm links these detections into coherent trajectories. All our experiments were performed on an NVIDIA A100 GPU (CPU AMD EPYC 7713 64-Core Processor). Detailed methodology is outlined below: Object Detection: As object detector, we selected the ScatYOLOv8+CBAM architecture [9]. This object detector builds upon the YOLOv8 framework to enhance ship recognition in maritime environments, focusing on refining meaningful visual features for accurate vessel detection and localization. To this end, we adapt YOLOv8x and add an extra prediction head to conform ScatYOLOv8x+CBAM+P. For pre-training data augmentation, we utilized the SAHI workflow [5], slicing the LaRS dataset training and validation partitions into 640×640 pixel segments with a 20slicing, the training and validation partitions of the LaRS dataset produced an augmented dataset of 19913 sliced training images and 1597 sliced validation images. This augmented dataset was used to train our ScatYOLOv8x+CBAM+P model over 40 epochs, starting with random weights and the default YOLOv8 training parameters. We excluded SAHI during inference to prioritize processing efficiency. Tracking: The tracker employs the BYTE association mechanism [54] with the camera motion compensation developed in BoT-SORT [4]. This amounts to using BoT-SORT with no re-identification module. The selection of confidence thresholds for the first and second associations is performed empirically by looking at the statistics of confidences generated by the detector on each video. The default parameter configurations of BoT-SORT were adjusted for video 366.avi, for which a high-score threshold of 0.2, a threshold of 0.4 for new tracks, and a threshold of 0.5 for confirming were used. For the rest of the videos, the parameter configurations remain close to the default values: high-score threshold of 0.5, new track threshold of 0.6, and confirmation threshold of 0.7. After inference, the results for all the videos are interpolated using the standard procedure in the MOT challenge. Our method achieved a HOTA of 0.193, MOTA of 0.057, and IDF1 of 0.202 on the BoaTrack test set, running at an average of 12.6 FPS, including the object detection timings.

References

[1] LOOKOUT AI System. `https://www.getalookout.com/`. Accessed: 2023-11-18.

[2] YOLOv8 Github. `https://github.com/ultralytics/ultralytics`. Accessed: 2023-11-18.

[3] N. Aharon, R. Orfaig, and B.-Z. Bobrovsky. Bot-sort: Robust associations multi-pedestrian tracking. *arXiv:2206.14651*, 2022.

[4] Nir Aharon, Roy Orfaig, and Ben-Zion Bobrovsky. Bot-sort: Robust associations multi-pedestrian tracking. *arXiv preprint arXiv:2206.14651*, 2022.

[5] Fatih Cagatay Akyon, Sinan Onur Altinuc, and Alptekin Temizel. Slicing aided hyper inference and fine-tuning for small object detection. In *2022 IEEE International Conference on Image Processing (ICIP)*, pages 966–970. IEEE, 2022.

[6] Fatih Cagatay Akyon, Sinan Onur Altinuc, and Alptekin Temizel. Slicing aided hyper inference and fine-tuning for small object detection. In *2022 IEEE International Conference on Image Processing (ICIP)*, pages 966–970, 2022.

[7] Alexander Buslaev, Vladimir I. Iglovikov, Eugene Khvedchenya, Alex Parinov, Mikhail Druzhinin, and Alexandr A. Kalinin. Albumentations: Fast and flexible image augmentations. *Information*, 11(2), 2020.

[8] Borja Carrillo-Perez, Angel Bueno Rodriguez, Sarah Barnes, and Maurice Stephan. Improving yolov8 with scattering transform and attention for maritime awareness. In *2023 International Symposium on Image and Signal Processing and Analysis (ISPA)*, pages 1–6, 2023.

[9] Borja Carrillo-Perez, Angel Bueno Rodriguez, Sarah Barnes, and Maurice Stephan. Improving yolov8 with scattering transform and attention for maritime awareness. In *2023 International Symposium on Image and Signal Processing and Analysis (ISPA)*, pages 1–6. IEEE, 2023.

[10] Kai Chen, Jiangmiao Pang, Jiaqi Wang, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jianping Shi, Wanli Ouyang, et al. Hybrid task cascade for instance segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4974–4983, 2019.

[11] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, et al. Mmdetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019.

[12] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, Zheng Zhang, Dazhi Cheng, Chenchen Zhu, Tianheng Cheng, Qijie Zhao, Buyu Li, Xin Lu, Rui Zhu, Yue Wu, Jifeng Dai, Jingdong Wang, Jianping Shi, Wanli Ouyang, Chen Change Loy, and Dahua Lin. MMDetection: Open MMLab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019.

[13] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(4):834–848, 2018.

[14] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017.

[15] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

[16] Bowen Cheng, Ishan Misra, Alexander G. Schwing, Alexander Kirillov, and Rohit Girdhar. Masked-attention mask transformer for universal image segmentation. 2022.

[17] MMSegmentation Contributors. MMSegmentation: Openmmlab semantic segmentation toolbox and benchmark. `https://github.com/open-mmlab/mmsegmentation`, 2020.

[18] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[19] Y. Du, J. Wan, Y. Zhao, B. Zhang, Z. Tong, and J. Dong. Giaotracker: A comprehensive framework for mcmot with global information and optimizing strategies in visdrone 2021. In *Int. Conf. Comput. Vis. Worksh.*, pages 2809–2819, 2021.

[20] Mingyuan Fan, Shenqi Lai, Junshi Huang, Xiaoming Wei, Zhenhua Chai, Junfeng Luo, and Xiaolin Wei. Rethinking bisenet for real-time semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9716–9725, 2021.

[21] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738, 2020.

[22] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[23] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 770–778, 2016.

[24] Jitesh Jain, Jiachen Li, MangTik Chiu, Ali Hassani, Nikita Orlov, and Humphrey Shi. OneFormer: One Transformer to Rule Universal Image Segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

[25] Benjamin Kiefer, Matej Kristan, Janez Perš, Lojze Žust, Fabio Poiesi, Fabio Andrade, Alexandre Bernardino, Matthew Dawkins, Jenni Raitoharju, Yitong Quan, Adem Atmaca, Timon Höfer, Qiming Zhang, Yufei Xu, Jing Zhang, Dacheng Tao, Lars Sommer, Raphael Spraul, Hangyue Zhao, Hongpu Zhang, Yanyun Zhao, Jan Lukas Augustin, Eui-ik Jeon, Impyeong Lee, Luca Zedda, Andrea Loddo, Cecilia Di Ruberto, Sagar Verma, Siddharth Gupta, Shishir Muralidhara, Niharika Hegde, Daitao Xing, Nikolaos Evangeliou, Anthony Tzes, Vojtěch Bartl, Jakub Špaňhel, Adam Herout, Neelanjan Bhowmik, Toby P. Breckon, Shivanand Kundargi, Tejas Anvekar, Ramesh Ashok Tabib, Uma Mudenagudi, Arpita Vats, Yang Song, Delong Liu, Yonglin Li, Shuman Li, Chenhao Tan, Long Lan, Vladimir Somers, Christophe De Vleeschouwer, Alexandre Alahi, Hsiang-Wei Huang, Cheng-Yen Yang, Jenq-Neng Hwang, Pyong-Kun Kim, Kwangju Kim, Kyoungoh Lee, Shuai Jiang, Haiwen Li, Zheng Ziqiang, Tuan-Anh Vu, Hai Nguyen-Truong, Sai-Kit Yeung, Zhuang Jia, Sophia Yang, Chih-Chung Hsu, Xiu-Yu Hou, Yu-An Jhang, Simon Yang, and Mau-Tsuen Yang. 1st workshop on maritime computer vision (macvi) 2023: Challenge results. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV) Workshops*, pages 265–302, January 2023.

[26] Benjamin Kiefer and Lojze Žust. Macvi website.

[27] Harold W Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955.

[28] Laura Leal-Taixé, Anton Milan, Ian Reid, Stefan Roth, and Konrad Schindler. Motchallenge 2015: Towards a benchmark for multi-target tracking. *arXiv preprint arXiv:1504.01942*, 2015.

[29] Tingting Liang, Xiaojie Chu, Yudong Liu, Yongtao Wang, Zhi Tang, Wei Chu, Jingdong Chen, and Haibin Ling. Cbnet: A composite backbone network architecture for object detection. *IEEE Transactions on Image Processing*, 31:6893–6906, 2022.

[30] T. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *Eur. Conf. Comput. Vis.*, pages 740–755, 2014.

[31] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017.

[32] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[33] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10012–10022, 2021.

[34] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.

[35] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

[36] Jonathon Luiten, Aljosa Osep, Patrick Dendorfer, Philip Torr, Andreas Geiger, Laura Leal-Taixé, and Bastian Leibe. Hota: A higher order metric for evaluating multi-object tracking. *International journal of computer vision*, 129(2):548–578, 2021.

[37] Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, and Silvio Savarese. Generalized intersection over union: A metric and a loss for bounding box regression. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 658–666, 2019.

[38] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. Orb: An efficient alternative to sift or surf. In *Int. Conf. Comput. Vis.*, pages 2564–2571, 2011.

[39] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *Int. J. Comput. Vis.*, 2015.

[40] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018.

[41] Shuai Shao, Zeming Li, Tianyuan Zhang, Chao Peng, Gang Yu, Xiangyu Zhang, Jing Li, and Jian Sun. Objects365: A large-scale, high-quality dataset for object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8430–8439, 2019.

[42] Matija Teršek, Lojze Žust, and Matej Kristan. ewasr—an embedded-compute-ready maritime obstacle detection network. *Sensors*, 23(12):5386, 2023.

[43] Ultralytics. Training arguments. `https://docs.ultralytics.com/modes/train/#arguments`. Accessed: 19.11.2023.

[44] Leon Amadeus Varga, Benjamin Kiefer, Martin Messmer, and Andreas Zell. Seadronessee: A maritime benchmark for detecting humans in open water. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2260–2270, 2022.

[45] Zhongdao Wang, Hengshuang Zhao, Ya-Li Li, Shengjin Wang, Philip Torr, and Luca Bertinetto. Do different tracking tasks require different appearance models? *Advances in Neural Information Processing Systems*, 34:726–738, 2021.

[46] Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M Alvarez, and Ping Luo. Segformer: Simple and efficient design for semantic segmentation with transformers. *Advances in Neural Information Processing Systems*, 34:12077–12090, 2021.

[47] Jiacong Xu, Zixiang Xiong, and Shankar P. Bhattacharyya. Pidnet: A real-time semantic segmentation network inspired from pid controller, 2022.

[48] Cheng-Yen Yang, Hsiang-Wei Huang, Zhongyu Jiang, Heng-Cheng Kuo, Jie Mei, Chung-I Huang, and Jenq-Neng Hwang. Sea you later: Metadata-guided long-term re-identification for uav-based multi-object tracking. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV) Workshops*, January 2024.

[49] Changqian Yu, Changxin Gao, Jingbo Wang, Gang Yu, Chunhua Shen, and Nong Sang. Bisenet v2: Bilateral network with guided aggregation for real-time semantic segmentation. *International Journal of Computer Vision*, 129:3051–3068, 2021.

[50] Changqian Yu, Jingbo Wang, Chao Peng, Changxin Gao, Gang Yu, and Nong Sang. Bisenet: Bilateral segmentation network for real-time semantic segmentation. In *Proceedings of the European conference on computer vision (ECCV)*, pages 325–341, 2018.

[51] Hao Zhang, Feng Li, Shilong Liu, Lei Zhang, Hang Su, Jun Zhu, Lionel M Ni, and Heung-Yeung Shum. Dino: Detr with improved denoising anchor boxes for end-to-end object detection. *arXiv preprint arXiv:2203.03605*, 2022.

[52] H. Zhang, Y. Wang, F. Dayoub, and N. Sunderhauf. Varifocalnet: An iou-aware dense object detector. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 8514–8523, 2021.

[53] Wenwei Zhang, Jiangmiao Pang, Kai Chen, and Chen Change Loy. K-Net: Towards Unified Image Segmentation. In *Advances in Neural Information Processing Systems*, Oct. 2021.

[54] Y. Zhang, P. Sun, Y. Jiang, D. Yu, F. Weng, Z. Yuan, P. Luo, W. Liu, and X. Wang. Bytetrack: Multi-object tracking by associating every detection box. In *Eur. Conf. Comput. Vis.*, pages 1–21, 2022.

[55] Bolei Zhou, Hang Zhao, Xavier Puig, Tete Xiao, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Semantic understanding of scenes through the ade20k dataset. *Int. J. Comput. Vis.*, 2019.

[56] Zhuofan Zong, Guanglu Song, and Yu Liu. Detrs with collaborative hybrid assignments training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6748–6758, 2023.

[57] Lojze Žust, Janez Perš, and Matej Kristan. Lars: A diverse panoptic maritime obstacle detection dataset and benchmark. In *International Conference on Computer Vision (ICCV)*, 2023.