Czech Technical University in Prague

Faculty of Electrical Engineering
Department of Computer Science

Fusion of data from dual RGB sensors and thermal camera

Master's Thesis

Ondřej Kafka

Prague, May 2024

Study programme: Open Informatics
Branch of study: Artificial Intelligence

Supervisor: Ing. Petr Štěpán, Ph.D.

Acknowledgments

Firstly, I would like to express my gratitude to the SEA.AI team for the opportunity to work on this project. Namely, I would like to thank Josef Kaufmann, Kevin Serrano, and David Moser for their support and guidance. I would also like to thank my supervisor, Ing. Petr Štěpán, Ph.D., for his insights and feedback. I would also like to thank my family, partner, and friends for their support and encouragement. Lastly, I would like to thank all my colleagues and friends who have contributed to this work in any way.

MASTER‘S THESIS ASSIGNMENT

I. Personal and study details

Student's name:	**Kafka Ondřej**	Personal ID number:	**483475**
Faculty / Institute:	**Faculty of Electrical Engineering**		
Department / Institute:	**Department of Computer Science**		
Study program:	**Open Informatics**		
Specialisation:	**Artificial Intelligence**		

II. Master’s thesis details

Master’s thesis title in English:

Fusion of data from dual RGB sensors and thermal camera

Master’s thesis title in Czech:

Fúze dat ze duálních senzorů RGB a termo kamery

Guidelines:

1) Explore sea.ai's sensors that provide data from an RGB camera and one or more thermal cameras.
2) Study the algorithms for data fusion and analysis of RGB and thermal images
3) Verify the camera calibration matrices and time synchronicity of the given datasets from the sea.ai sensors. Design algorithms to refine the determination of the relative position of the RGB and thermal camera images
4) Develop algorithms to detect object data from the fusion of all available cameras. Evaluate the proposed algorithms on the provided datasets.

Bibliography / sources:

Žižka Ivan, "Stereo termokamera", Bakalářská práce FEL, ČVUT 2022
Brenner, Martin, et al. "RGB-D And Thermal Sensor Fusion: A Systematic Literature Review." arXiv preprint arXiv:2305.11427 (2023).
Y. Sun, W. Zuo and M. Liu, "RTFNet: RGB-Thermal Fusion Network for Semantic Segmentation of Urban Scenes," in IEEE Robotics and Automation Letters, vol. 4, no. 3, pp. 2576-2583, July 2019, doi: 10.1109/LRA.2019.2904733.

Name and workplace of master’s thesis supervisor:

RNDr. Petr Štěpán, Ph.D. Multi-robot Systems FEE

Name and workplace of second master’s thesis supervisor or consultant:

Date of master’s thesis assignment: **01.02.2024** Deadline for master's thesis submission: ___________

Assignment valid until: **21.09.2025**

RNDr. Petr Štěpán, Ph.D. Supervisor’s signature	Head of department’s signature	prof. Mgr. Petr Páta, Ph.D. Dean’s signature

III. Assignment receipt

The student acknowledges that the master’s thesis is an individual work. The student must produce his thesis without the assistance of others, with the exception of provided consultations. Within the master’s thesis, the author must state the names of consultants and include a list of references.

.

Date of assignment receipt	Student’s signature

Declaration

I declare that presented work was developed independently, and that I have listed all sources of information used within, in accordance with the Methodical instructions for observing ethical principles in preparation of university theses.

Date ....

Abstract

In the maritime domain, navigation and collision avoidance traditionally rely on human vision, which is limited, especially under adverse environmental conditions. This thesis addresses the integration of multimodal object detection using thermal and RGB images to enhance situational awareness in these environments. A deep learning-based approach is introduced for correspondence search and homography estimation, enabling precise alignment and annotation propagation. This alignment is subsequently used to create a high-quality dataset. Several state-of-the-art architectures and our proposed architectures are trained and evaluated on this dataset, revealing that leveraging a second modality leads to improved performance across various metrics. Notably, transformer-based architectures like CMX show the highest performance but at the cost of increased complexity and inference times. Simpler models, such as our proposed WNet-S, demonstrate competitive results with better efficiency, indicating that complexity does not always correlate with better performance.

Keywords Object Detection, Sensor Fusion, Semantic Segmentation, Naval Computer Vision

Abstrakt

V oblasti námořní navigace je při vyhýbání se kolizím tradičně spoléháno na lidský zrak, který je ale omezený především v nepříznivých podmínkách. Tato práce se zabývá integrací multimodální detekce objektů pomocí termální a RGB obrazové kamery pro zlepšení situačního povědomí v těchto podmínkách. Je představen přístup založený na hlubokém učení pro vyhledávání korespondencí a odhad homografie, který umožňuje přesné zarovnání a propagaci anotací skrz modality. Toto zarovnání je následně použito k vytvoření datasetu. Na tomto datasetu je natrénováno a vyhodnoceno několik state-of-the-art architektur a architektur navržených v této práci. Výsledky benchmarků ukázaly, že využití druhé modality zlepšuje výkon napříč různými metrikami. Zvláště architektury založené na transformerech, jako je CMX, dosahují nejvyššího výkonu, avšak za cenu zvýšené složitosti a delších časů inferencí. Jednodušší modely, jako náš navržený WNet-S, vykazují konkurenceschopné výsledky s lepší efektivitou, což naznačuje, že složitost ne vždy koreluje s lepším výkonem.

Klíčová slova Detekce objektů, Senzorová fúze, Sematická segmentace, Námořní Počítačové Vidění

Contents

Chapter 1

Introduction

In the maritime domain, the reliance on human vision for navigation and collision avoidance has long been a cornerstone of seafaring tradition. However, this dependence poses inherent limitations on situational awareness, particularly in adverse environmental conditions characterized by reduced illumination or unfavorable weather. Despite advancements in computer vision in other sectors, the integration of such technologies into maritime practices remained relatively underexplored until recently. Moreover, autonomous navigation builds solely on GPS- or radar-based planning with little attention to computer vision. However, leveraging the visual information the intermediate surroundings offer would serve for immediate collision avoidance with objects that are not visible using GPS or radar. Such a device is attached to the ship directly and can recognize collision threats and warn the conductor or navigation system in time. Such a system can be of crucial importance for smaller vessels, on whom a barrel in the water may pose an existential danger. These systems have recently been brought to market by companies such as SEA.AI, in cooperation with which this thesis was created. Images of their systems can be seen in Figure 1.1.

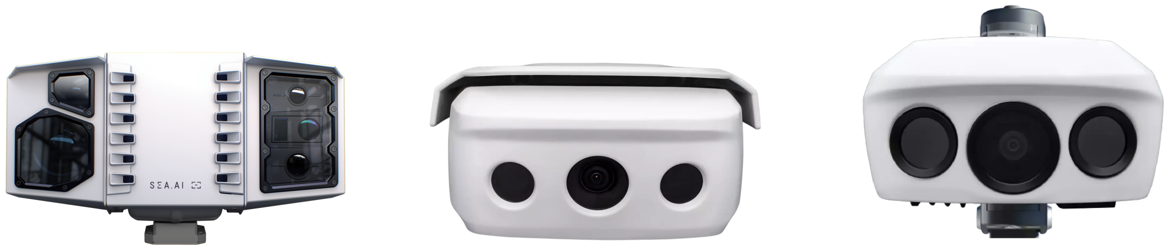

Figure 1.1: Multimodal naval devices offered by SEA.AI. Sentry, Offshore, and Competition left to right.

For image processing with this purpose in mind, object detection or prospective instance segmentation, essentially a version of semantic segmentation, are suitable algorithms. In recent years, there has been a notable shift in the approach to object detection and semantic segmentation, transitioning from traditional computer vision algorithms to methods based on deep learning. Among these, convolutional neural networks (CNNs) have demonstrated remarkable results in addressing the challenges posed not only by semantic segmentation and object detection [44] but also in image classification [48] or more complex methods such as correspondence search among pairs of images. The segmentation and detection tasks are essential in computer vision applications and find diverse uses in areas such as robotic sensing, video surveillance, and autonomous driving. The evolution towards deep learning-based techniques has marked a significant trend, represented by higher usage of various neural network architectures to achieve more accurate outcomes in a variety of computer vision tasks.

However, as the prevailing standard in object detection or semantic segmentation is the utilization of neural networks designed exclusively for processing RGB images captured by visible light cameras, the performance is susceptible to degradation under unfavorable lighting or weather conditions. For instance, many algorithms struggle to detect objects in near-total darkness accurately. In contrast, thermal imaging cameras can operate effectively across diverse lighting scenarios. Unlike visible light cameras, which function within the wavelengths from 0.4μm to 0.7μm, the visible light spectrum, thermal imaging cameras generate images from the longwave infrared (LWIR) part of the spectrum, which consists of wavelengths from 8μm to 12μm. This radiation is emitted by all objects with normal outdoor temperatures and thus is naturally independent of the visible light [2]. All these features combined suggest that the information provided by thermal cameras is a valuable complement to RGB information and is more robust to adversarial environmental conditions. Moreover, integrating thermal images can enhance RGB image segmentation algorithms by providing an alternative scene representation, which is often a plus for deep-learning methods.

After all, the concept of multimodal fusion with the aim of deep-learning algorithm accuracy improvement is nothing new [2]. These methods have already found their usage in areas such as autonomous vehicles, aviation, surveillance, and even the movie or gaming industry, where these methods are combined with depth sensors to create more accurate 3D models. Generally, the additional sensor typically used provides thermal, depth, or polarization data.

Therefore, integrating thermal and RGB images presents a promising avenue for advancing object detection and semantic segmentation techniques, particularly in scenarios where conventional RGB-based methods or human vision falter due to challenging environmental conditions. Analogous applications in various other fields support this approach.

While building on the abovementioned, this thesis aims to propose and implement an algorithm for segmentation and object detection using multimodal RGB and thermal marine visual data provided by SEA.AI. However, this task firstly requires dataset preprocessing as the initial data is unsuitable for this purpose right off the shelf. Additionally, the outcomes of this study should include a comparison with both state-of-the-art methods and the methods currently employed in SEA.AI devices, considering performance and resource demands, given that current algorithms are deployed on edge devices with limited computational capabilities.

Thesis outline

Chapter one introducs the motivation behind the research, emphasizing the potential benefits of integrating thermal and RGB images for semantic segmentation in maritime applications. The goals of the thesis were outlined, focusing on proposing an algorithm for object detection using multimodal visual data and exploring its applications in improving naval navigation along with appropriate data preprocessing pipelines.

Chapter two reviews existing literature on semantic segmentation, object detection, and multimodal image processing. It defines key concepts and challenges in semantic segmentation using multimodal visual data, providing a foundation for subsequent discussions.

The third chapter provides an overview of the dataset used in the study, detailing its features, acquisition process, sensor description and details, and preprocessing methods, such as alignment and subsampling. This chapter sets the stage for implementing the proposed algorithm by ensuring the dataset's suitability for multimodal semantic segmentation and object detection tasks.

Chapter four outlines the proposed algorithms for object detection using multimodal RGB and thermal data. It also discusses additional approaches sourced from existing literature for comparison purposes, highlighting similarities and differences with the proposed algorithm as well as combinations of these methods.

The fifth chapter presents the results of the implemented methods and details the benchmarking experiments conducted to evaluate their performance. All the metrics used for the evaluation are also introduced there. This chapter provides insights into the effectiveness and efficiency of the proposed algorithm compared to existing methods as well.

The final chapter summarizes the findings of the study and discusses their implications for maritime navigation and safety. It also identifies potential areas for future research and concludes the thesis with reflections on the contributions made to the field of semantic segmentation in maritime applications.

Chapter 2

Related work

It was highlighted above that there is a marine industry's demand for robust and prompt potential threat and object detection on collision course capabilities. However, these can be compromised in low-light or adverse weather conditions. To address this potential detection vulnerability, the fusion of thermal and RGB images emerges as a compelling solution. Thermal imaging technology, operating beyond the constraints of visible light, offers distinct advantages in capturing scene details regardless of lighting variations. Deployed on top of such technologies are deep learning algorithms, such as object detection, instance segmentation, or main semantic segmentation.

Semantic segmentation is an image processing method that involves densely labeling pixels of predefined categories with the label of their respective category. The primary algorithm used in this realm for RGB image processing is UNet [42]. However, much evolution succeeded this initial architecture, such as state-of-the-art methods DeeplabV3 [28] introducing atrous convolution, or transformer-based architecture ViT-Adapter [9]. The work dedicated to multimodal semantic segmentation is outlined in the first section of this chapter, Chapter 2.

Object detection is a crucial image processing technique that involves assigning bounding boxes to objects of interest within an image, thereby determining their locations. Historically, this task has been predominantly tackled using Convolutional Neural Networks (CNNs), usually coupled with some region proposal part, an example of which is Fast-RCNN [39]. Another approach replaces the region proposal with a predefined grid with predefined box shapes, where the bounding boxes are estimated in a single forward pass, an approach introduced in YOLO [38]. These methods are further extended into instance segmentation, where individual masks for each detected object are generated, bridging the gap between the aforementioned techniques. Notable advancements in this field include subsequent iterations of YOLO [27] models and the development of MaskRCNN [31]. However, these architectures are not adopted in this work for multimodal sensor fusion, as more of the multimodal salient object detection architectures and benchmarks are based on the same principles as the semantic segmentation architectures [18]. Moreover, also the reference architecture from SEA.AI is built this way. The instances are separated later from the segmentation outputs in a postprocessing phase.

Multimodal semantic segmentation

In the realm of RGB and thermal image (RGBT) joint semantic segmentation, the cornerstone was set by MFNet [29] by publishing an aligned multimodal dataset with pixel-precise annotations of street scenes intended for autonomous vehicles (RS RGBT Dataset). The publication also proposed a symmetric double encoder architecture for segmentation resembling U-Net [42], with a notable departure: feature summation at each level instead of concatenation. The encoder MFNet employs is VGG16 [45] with slight modifications to allow for the processing of higher-resolution images. This architecture was built upon in other works, starting with RTFNet [24] and FuseSeg [22], preserving the double encoder architecture while altering the feature fusion strategies. Two-stage architecture PST900 [21] followed, achieving worse results in the RS Dataset, but was published with another dataset from the environment of the DARPA Subterranean Challenge. The two mentioned datasets are used as the architecture benchmarks by the community in most of the published papers on the RGBT segmentation topic.

Following these initial publications, other architectures were typically built upon the same basis of the symmetric double encoder using mainly ResNet [37], MobileNet [32], or VGG as the encoder. However, the main difference again came with fusion strategies. These fusion strategies are discussed further in Section 4.1 This division is, however, only a coarse one; each of the publications usually comes with a unique approach to fusion itself, replacing the original concatenation and summing strategies in the initial publications. However, after the initial publications, the number of published architectures increased significantly with minimal changes in mean accuracy on the RS RGBT dataset, often bringing a much higher computation cost as a tradeoff. As the explored task of detection needs to run at a considerable frame rate to be helpful in such near real-time perception edge devices, the networks with high inference times, which often stem from complex network architecture, are infeasible. Therefore, not all of them need to be mentioned due to marginal accuracy improvements over the older implementations, too complex architecture, or lack of implementation or implementation details.

Graded-Feature Multilabel-Learning Network (GMNet) [14] achieved the first considerable leap forward from the initial publications by introducing custom Shallow and Deep Feature Fusion Modules and a triple loss computed on boundaries, binary masks, and semantic masks. Even though it was published in 2021, this model has remained among the state-of-the-art models until today. What is more, it is also implemented with a lighter backbone. Another mention should go to MMSMCNet [8], which outperformed GMNet even with ResNet50 backbone, considered lightweight in this context. The final mentioned CNN-based RGBT architecture is Context-Aware Interaction Network [1], a recent publication with Mobilenet backbone, outperforming the previously mentioned architectures and preserving reasonable complexity for our usage while lowering the number of parameters in comparison to the previously mentioned methods.

With the recent rise of transformer architectures in natural language processing (NLP) [35], they also spread into most of the computer vision tasks such as object detection [17], image segmentation [13] or classification. Transformers thus also inevitably appeared in multimodal sensor fusion. Recognizable architecture based on this technology is Cross-modal fusion for RGB-X semantic segmentation with transformers (CMX) [7], which targeted multiple RGB-X modalities at once and became state-of-the-art in all of them, including RGBT or RGB and depth (RGBD). Even though this architecture is too memory-demanding for our use case, it is worth mentioning and using as a baseline. Let us use this network as a transition to a different additional sensor to RGB. Prevalent modalities explored in 2D sensor fusion and consequent semantic segmentation are thermal, depth, and RGB images. Even though this thesis operates solely with RGBT, a significant amount of work has been published on RGBD combination with promising results. Since the architectures are very similar to those published in the RGBT domain, they are very well utilizable for RGBT, which was among other results shown in CMX. Therefore, some of those methods should be mentioned as well.

The prevalent architecture feature for RGBD segmentation is a symmetric double encoder, too. The cornerstone representative for RGBD is FuseNet [30], with architecture very similar to MFNet, also using VGG16. This architecture was extended by RDFNet [34], using however Resnet152 as the backbone, making it more complex. However, RDFNet stays among SOTA until today. A notable architecture is also ACNet [23], which adds a third encoder used for merged feature maps. Moreover, the maps are merged using attention modules. Again, most of the other architectures bring only a small improvement to the results on the benchmarking datasets, so only a few of them are mentioned here. The first mention here goes to SGNet [4], bringing focus on a high number of frames per second (FPS), and CENNet [10], which is a more efficient update of the previous architecture RefineNet [33], allowing for higher FPS, too. EMSANet [11] is another mention here, which is one of the RGBT semantic segmentation architectures with the fewest parameters. The current state-of-the-art in non-transformer architectures is SA-Gate [16] using DeeplabV3+ decoder. The most notable in the field of transformers are CMX, as mentioned above, and its more efficient successor, DFormer [6].

Some of the architectures mentioned here for both modalities were trained on the dataset created in this work and are introduced more in-depth in Chapter 4, and the benchmark result details are shown in Chapter 5.

Chapter 3

Dataset

In the first part of this chapter, Section 3.1, the devices used for data collection are described. Then in Section 3.2 is shown the calibration and preprocessing of data from Sentry, and the preprocessing consisting of calibration and homography construction for data from Offshore is then discussed in Section 3.3. These preprocessing steps for both devices are then followed by the same subsequent alignment process to obtain homography suitable for the dataset needs outlined at the beginning of this chapter. This process is discussed in Section 3.4. The next part of this chapter Section 3.5 copes with the abundance and repetitiveness of the data in the provided dataset. The chapter is closed with an overview of the dataset that was created and a recapitulation of the steps performed in Section 3.6.

3.1 Devices description

Since the thesis focuses on RGBT sensor fusion and object detection in marine environments, it is built on a dataset comprising data obtained from RGB and thermal sensors collected in open waters provided by SEA.AI. Two types of devices were used to produce the data. The first device is SEA.AI Sentry, and it generates four data streams: narrow field of view (FoV) RGB, narrow FoV thermal, wide FoV RGB, and wide FoV thermal. The FoV values, along with the focal lengths of all the cameras, are shown in the device schematic in Figure 3.1. The second device, SEA.AI Offshore, produces three image streams –one fisheye RGB camera in landscape ratio and two slightly overlapping portrait ratio thermal streams with narrower FoV than RGB camera. However, even the fused thermal streams have a FoV much smaller than the RGB one. Device details are shown in the schematic in Figure 3.2. Both datasets are divided into collections called trips, where one trip is a sequence collected by one device throughout a voyage of a variable length.

The final dataset for the work is to be made of pairs of images (RGB and thermal images) with a single ground-truth target. The target can be a mask, bounding boxes, or both, depending on the computer vision task to be performed. However, the modalities are annotated separately, and the sensors always have different fields of view, resolution, and aspect ratio (the strength of these effects depends on the device and stream used), and the annotations are always in the frame of the respective modal. To obtain a desired dataset, we need to find an alignment between each tuple of images with high precision so that the ground truth targets can be correctly projected into one frame, which can be used to compute losses effectively after joint semantic segmentation or another task. Also, the images themselves need to be, at minimum, approximately aligned. If the aspect ratios and FoVs of models were completely different, it would be challenging for the network to find the corresponding parts of the images so that they can be projected into a single output. It would also need to select with respect to which input the output would be framed. Moreover, it is necessary to highlight the importance of the alignment step through the lens of annotations. It allows

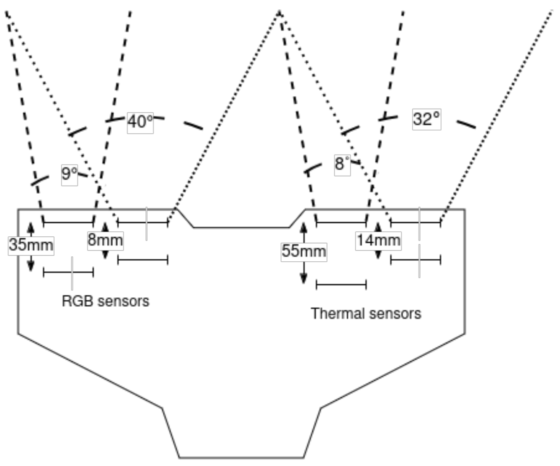

Figure 3.1: SEA.AI Sentry device chart with FoV angles and focal lengths used for data collection. Thermal cameras are on the right, and RGB cameras are on the left.

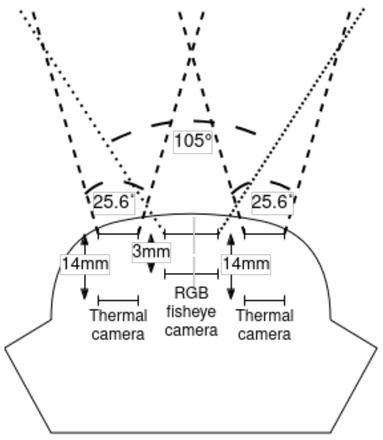

Figure 3.2: SEA.AI Offshore device chart used for data collection with Fov angles and focal lengths, thermal cameras are on the sides, and RGB camera is in the middle.

for the propagation of features even if available in only one of the channels in the training dataset. This unavailability can be due to the nature of the object or environmental conditions. Thus, a single ground truth frame is created, containing all the relevant information in both modalities and to which outputs are compared. The network also learns to leverage features that are available in a single sensor. One can argue that it would be advantageous if the detection network were robust to such slight misalignments between images. However, the perfectly aligned tuples can be slightly misaligned during augmentation before training to reach such an effect while still maintaining the advantages that aligned targets bring.

However, the misalignment is not caused solely by the factors mentioned above. The thermal and RGB cameras don't have synchronized clocks, so the closest timestamps pair the images. The desynchronization gives some space for misalignment of the frames from different channels. However, the main source of misalignment stems from the camera parameters itself. Even though cameras are calibrated, i.e., intrinsic and extrinsic parameters are provided, the limited calibration precision is visible after constructing the homography from the parameters and mapping from RGB to thermal space. An example of an image after such a mapping for Sentry narrowFoV is shown in Figure 3.3 and Offshore in Figure 3.5. A necessary remark is that the cameras have different camera origins. Namely, the cameras are mounted at different positions, the separation being in the orders of magnitude of 10 cm. This difference is not an issue for distant objects but would introduce some misalignment for close objects, such as parts of the ship visible in the camera's field of view or objects that are too close. Nevertheless, we typically do not deal with objects as close as to cause the misalignment, so we can rely on the image pairs preprocessed using the provided calibration parameters. Additionally, stemming from the distance assumption, translation of the camera origin is not included in the extrinsic parameters, so all the calibrations below work only with rotations.

3.2 Sentry dataset preparation

The part of the provided dataset produced by the Sentry device consists of videos. Still, as the temporal dependency is not leveraged in this work, the video datasets are converted to images, and these are grouped within modal and FoV to frame tuples, i.e., tuples of narrow FoV thermal and narrow FoV RGB and the same for wide FoV. In this work, only the narrow FoV data are used, as a simple pinhole model can be used for modeling the camera; also, no undistortion is needed, and what is more, the streams have the same viewing direction, so they capture similar information. As mentioned before, cameras are calibrated so that we can construct homography mapping RGB images to thermals or vice versa. If the reader would like to learn more about homographies and homogenous coordinates, these are introduced, e.g., in Ref. [52]. The projective mapping between RGB and thermal is built from calibration matrices for the pinhole camera model, which is of the form

$$K = \begin{bmatrix} f_x & 0 & x_0 \\ 0 & f_y & y_0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{3.1}$$

where f_x and f_y are a focal lengths and x_0 and y_0 are center offsets in x and y axes respectively. Their typical values are shown in Figure 3.1. For a more detailed explanation of calibration matrices and camera parameters, please refer to Ref. [52]. Along with the internal camera parameters, the extrinsic rotation parameters for the cameras, which are roll, yaw, and pitch angles, are also provided. From these angles, the rotation matrix can be constructed as

$$R = R_{yaw} \cdot R_{pitch} \cdot R_{roll} \tag{3.2}$$

for each of the sensors. The two matrices, R and K, are then composed into one homography matrix with the corresponding matrices for the other sensor, resulting in a projective transformation that maps the RGB image to the selected thermal one as

$$H_{R\to T} = K_T \cdot R_T \cdot R_R^{-1} \cdot K_R^{-1}, \tag{3.3}$$

where the subscript T corresponds to thermal-related matrices and R to RGB. The RGB image warped onto the thermal in this way is shown in Figure 3.3. The inverse of this matrix is then a mapping from thermal to RGB

$$H_{T\to R} = H_{R\to T}^{-1} = K_R \cdot R_R \cdot R_T^{-1} \cdot K_T^{-1}. \tag{3.4}$$

The thermal image warped onto the RGB one in this way is shown in Figure 3.4, and it shows the difference in the FoV, which is very slight in comparison to the difference for offshore in the next section.

3.3 Offshore dataset preparation

The preprocessing stage for the dataset collected using Offshore devices was performed similarly to that of Sentry. In this case, however, the dataset is provided already as frame tuples (three-membered in this case; RGB, thermal left, and thermal right), which are again grouped by timestamp, with the difference that the thermal channels are synchronized, as the two thermal cameras are using the same clock. Then, the RGB image is undistorted using

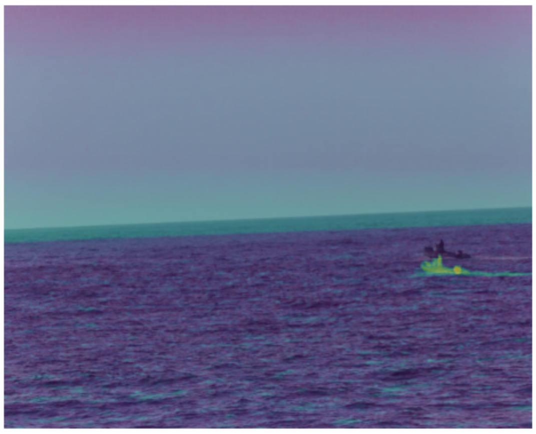
Figure 3.3: Overlapped RGB and thermal images after applying $H_{R\to T}$ constructed from camera parameters for Sentry.

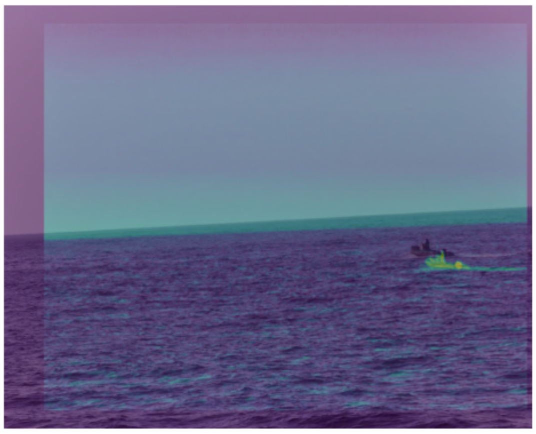
Figure 3.4: Overlapped RGB and thermal images after applying $H_{T\to R}$ constructed from camera parameters for Sentry.

calibration parameters obtained from checkerboard calibration. This results in the undistorted RGB image and a new calibration matrix for the camera that would virtually produce the undistorted image. This new camera matrix is of the format shown at Eq. 3.1. As undistortion is out of the scope of this work, for further details on checkerboard calibration and image undistortion, please refer to Ref. [52]. Moreover, there are different Offshore devices on which the dataset was collected, and as each was calibrated separately, there are also multiple undistorted calibration matrices. Therefore, only a single example of such an undistorted k-matrix is shown as

$$K_{new} = \begin{bmatrix} 539.2 & 0 & 766.1 \\ 0 & 540.0 & 587.6 \\ 0 & 0 & 1 \end{bmatrix}. \tag{3.5}$$

For the mapping from RGB to thermal image, we also need to consider external camera parameters, which are again yaw, pitch, and roll angles arranged into one matrix R as in Eq. 3.2 for each of the sensors. These two matrices are then composed into one homography for either the left or right camera as indicated at Eq. 3.3. Results after this initial homography warping can be seen in Figure 3.5. The inverse of this transformation is a mapping from thermal image to RGB image as indicated in Eq. 3.4. Results of such mapping using both thermal frames are illustrative of the differences in the FoV for thermal and RGB channels and can be seen in Figure 3.6.

3.4 Pixel-precise alignment

Even though the alignment pipeline starts with the homography warping based on calibration parameters, the modal images need to be further aligned for reasons outlined in previous sections. What was called modal alignment so far is also known in the literature as image registration [56]. Image registration is the process of aligning multiple pieces of data into one coordinated system. This can include various data types such as photographs, sensor data, or data from different times or viewpoints –in the case of this thesis, we are processing images from various sensors and views. There are two main approaches to image registration: spatial and frequency domain. The most used method in the frequency domain is the phase

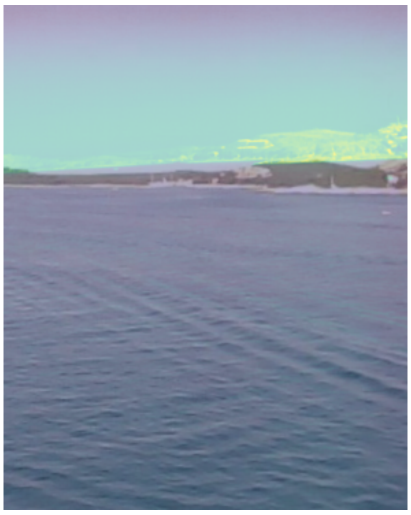

Figure 3.5: Overlapped RGB and thermal images after applying $H_{R \to TR}$ constructed from camera parameters for Offshore.

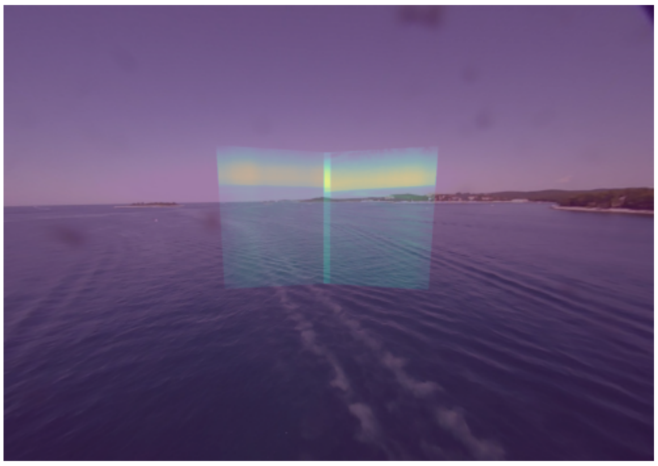

Figure 3.6: Overlapped RGB and both thermal images after applying $H_{T \to R}$ constructed from camera parameter for Offshore.

correlation –a translation estimation method that is resilient to occlusions or noise. This method is based on the fast Fourier transform (FFT). Phase correlation can also be used to determine rotation and scaling differences between two images by converting them to log-polar coordinates [54]. Since rotation estimation is translation-invariant, it is a common practice to estimate rotation first, rotate the image, and estimate translation.

In the spatial domain, a common approach is feature-based, meaning that the first step is feature detection and description [41] on both images, using handcrafted methods such as SIFT [53] or ORB [46] or deep-learning-based methods, e.g., SuperPoint [26]. The word handcrafted in the machine learning context means methods that are not learnable. The features are matched, again using handcrafted methods often based on nearest neighbor search such as in FLANN [49] or learned techniques such as SuperGlue [20]. Finally, a homography matrix is estimated based on the correspondences using mostly RANSAC-based approximates for relative pose camera estimation.

Note that all the images shown in this section are from the Sentry dataset unless said otherwise, simply to consistently outline the methods, but the results are shown at the end of the section, also on the data from Offshore, to illustrate the transferability of the developed approaches.

From now on, we assume that the homography provides the main projective warping, and the remaining differences can be parametrized as an affine transformation, i.e., transformation allowing only for scale, rotation, and translation. This allows us to use the frequency domain, as only scale, rotation, and translation between images can be estimated with FFT. Another assumption is that the initial homography aligns the images reasonably well, so the affine transformation will only have a certain effects-magnitude, i.e., the parameters of the transformation are small. We can use this to automatically register homography outliers or reduce the hypothesis space automatically.

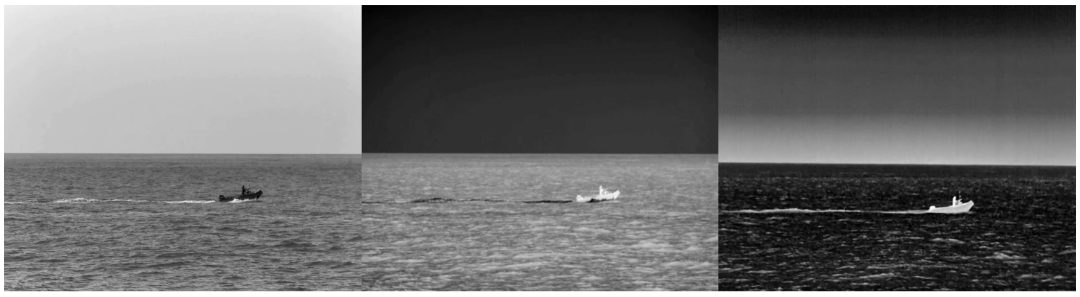

Figure 3.7: Original RGB image, preprocessed RGB image and thermal image.

3.4.1 Frequency domain methods

The experiments were conducted in the same order as outlined –starting in the frequency domain. The methods used in this section are described in Ref. [50], but the authors use a different notation. Firstly, the three-channel RGB data needs to be transformed into a single-channel grayscale, which can be easily achieved using, e.g., the OpenCV library. The next step is preprocessing. RGB image preprocessing is applied to compensate for the different nature of images. Methods tested are color inversion, Contrast-limited Adaptive Histogram Equalization (CLAHE) [55], Wallis filter [57] or Gaussian blur to remove noise. An example image preprocessed with CLAHE, color inversion, and blur is shown in Figure 3.7 along with the original RGB image and a corresponding thermal image. The experiments below are conducted on images with the preprocessing shown above since it outperformed other preprocessing methods in experiments. Moreover, results with no preprocessing were not alignable at all.

Let us continue with the outline of translation estimation first. Its core method, the phase correlation, is also the cornerstone of rotation estimation further in this section. The phase correlation produces a complex function whose maximum magnitude corresponds to the shift estimate between the two input images. The landscape of such a process is shown in Figure 3.8. The figure contains a cropped version to the window of size 120×120, as 30 is by far the maximum shift we assume between the images in the dataset. The landscape was created from twice the same image but once shifted by (-7,8) pixels. The displayed landscape is produced using

$$\mathsf{CSPD}(I_{th}, I_{RGB}) = \frac{\mathcal{F}(I_{th}) \circ \mathcal{F}(I_{RGB})^*}{|\mathcal{F}(I_{th}) \circ \mathcal{F}(I_{RGB})^*|} \tag{3.6}$$

for the computation of the cross power spectral density (CPSD), where $\mathcal{F}$ is a 2D fast Fourier transform (FFT), $(\cdot)^*$ symbolizes complex conjugate and $\circ$ and $|\cdot|$ are element-wise product and absolute value. The landsacape for shift estimation between thermal image I_{th} and preprocessed RGB image I_{RGB} is then computed from CSPD as

$$r = \mathcal{F}^{-1}(\mathsf{CSPD}(I_{th}, I_{RGB})). \tag{3.7}$$

This discrete function is then searched for its maximum, as explained before, resulting in the shift coordinates as

$$(s_x, s_y) = \mathsf{argmax}_{(x,y)}(r), \tag{3.8}$$

which yields the shift x and y direction estimate.

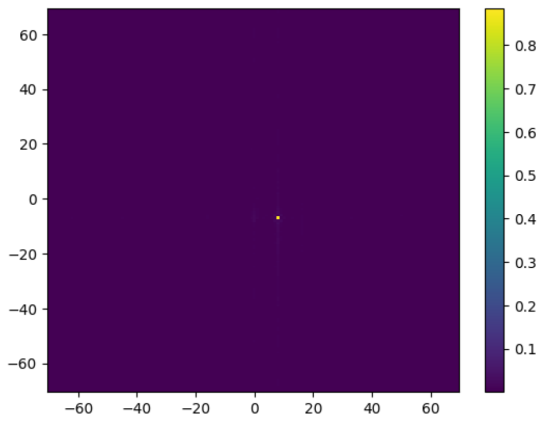

Figure 3.8: Ideal objective function landscape cropped to window size of 120 × 120 for FFT-based translation estimation.

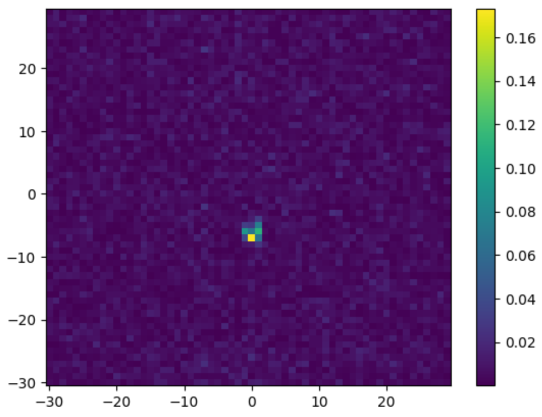

Figure 3.9: Ideal objective function landscape cropped to window size of 60 × 60 for FFT-based rotation and scale estimation.

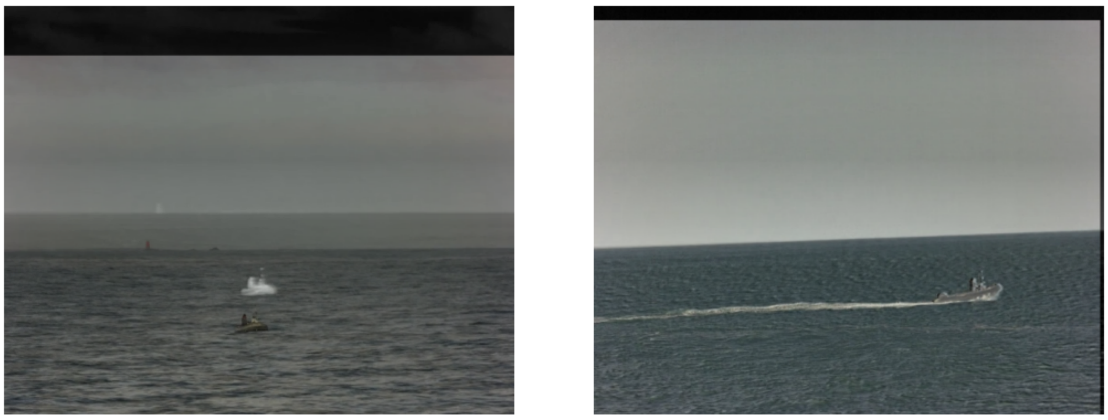

Figure 3.11: FFT-based alignment estimation result with a suitable pair from the dataset on the right and typical results on the left.

However, when we apply the same process to the pair of RGB and thermal images from the dataset, the typical result does not satisfy the alignment requirements for ground-truth propagation. We obtain a correct translation on a suitable pair of images, which can be seen in the right part of Figure 3.11, but the typical result can be seen on the left. A standard (normalized and cropped) landscape for a combination from the dataset can be seen in Figure 3.12. We can see that the results do not satisfy the pixel-precise criteria.

However, it still remains to tackle the rotation differences, as different rotations can cause the poor outcomes above. Moreover, FFT-based rotation estimation is translation invariant, which means it is a suitable first step. After the preprocessing, a windowing function is applied to avoid issues with the periodicity of the FFT –typically Hann window. Then, the images are transformed using FFT. The log-polar transform of the magnitude of the frequency-domain images is applied. For the polar transformed magnitudes, refer to Figure 3.14. Images transformed in such a way are then used in the cross-correlation described above, yielding a shift in the log-polar space. The landscape of the result of this process is shown in Figure 3.9. The figure contains a cropped version to the window of size 60 × 60. The landscape was created from twice the same image but once rotated by -5 degrees. From that shift in log-polar

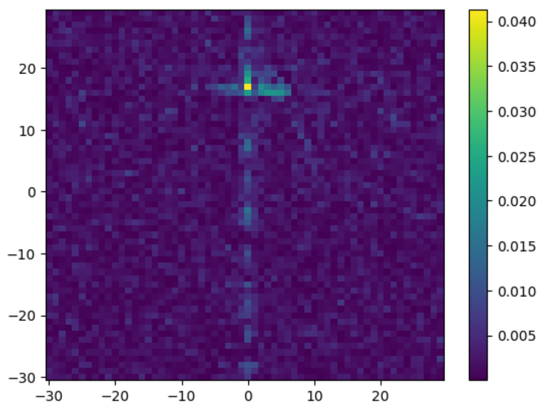

Figure 3.12: A typical FFT translation estimation objective function landscape.

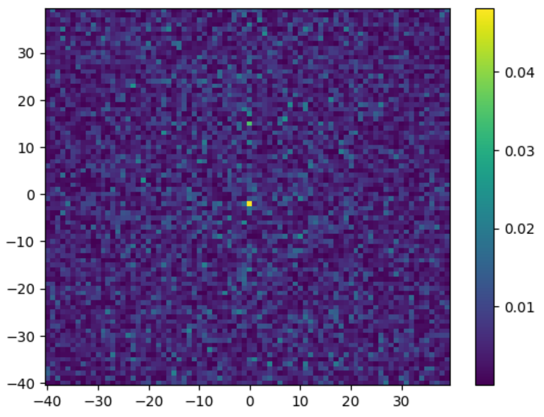

Figure 3.13: A typical FFT rotation and scale estimation objective function landscape.

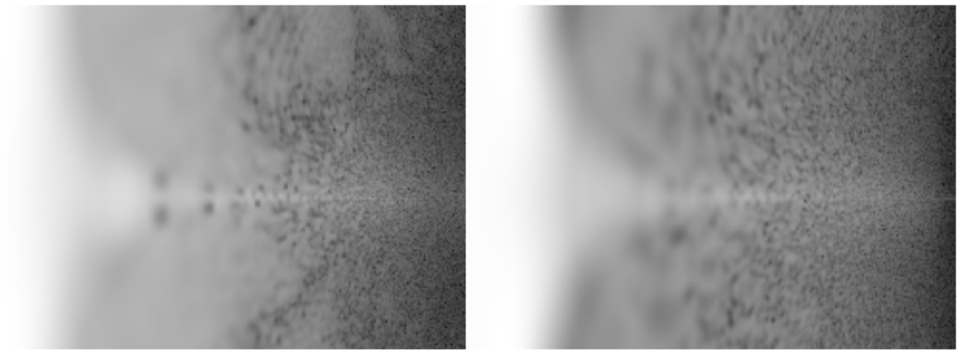

Figure 3.14: Log-polar transform of the magnitude of the frequency-domain images.

space, the rotation in degrees is computed as $\texttt{shift}_y/h * 360$, where h is the height of the image. Again, we can find some good results on suitable images, an example of what can be seen in the right part of Figure 3.16. Note that this result is after both correct rotation and translation estimation. However, most images are misaligned even after this step, as seen in the same Figure 3.16 on the left. The landscape of a rotation estimation function of a typical image pair from the dataset is in Figure 3.13.

The experiments have shown that these methods work perfectly on artificially transformed images. In the case of a cross-modal application, FFT-based alignment works only in some cases where the image pairs are suitable. It is challenging to provide any further specifications of suitability since it can't certainly be said from the experiments. Still, from the image pairs that were successfully aligned, it seems that a clear horizon is not enough, and some dominant, such as a big ship or long wave after a dinghy, needs to be present. From the CPSD landscapes, we can also deduce that the translation is generally more stable since its landscape seems smoother and with clearer peaks. However, the performance is not as good as needed for this application. It can be caused by the different perceptions of wave or sky patterns, and since this is the most prevalent feature of the images, such an apparent correspondence as a small ship can be then dominated by the noise from the environment. Another cause can also be the relative movement of waves and floating objects. One of the main issues is also the fact that there are not many options in terms of robustness or generalization-across-more-images

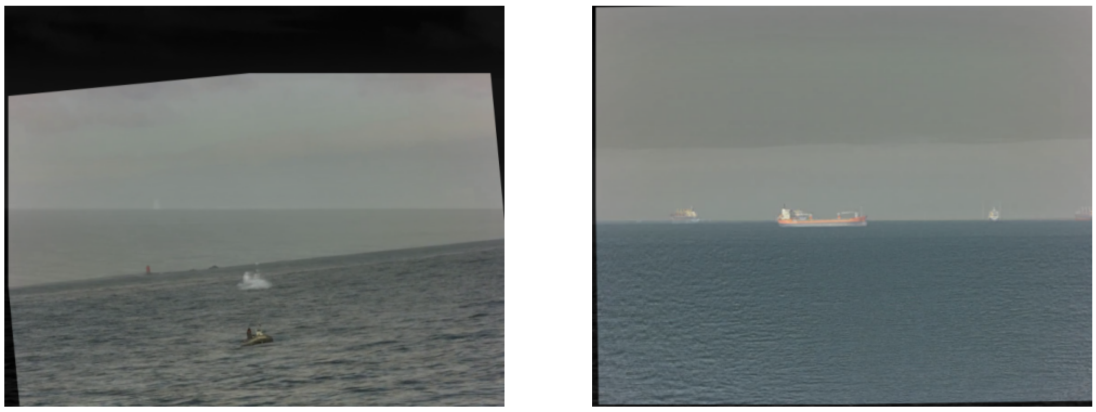

Figure 3.16: FFT-based rotation and scale estimation result with a suitable pair from the dataset on the right and typical results on the left.

improvements for these methods.

3.4.2 Spatial domain methods

The mentioned problems of frequency-domain methods naturally lead to spatially oriented approaches, as basing the transformation method on significant features, such as floating objects, seems promising.

In the correspondence-search-based approaches, we start with converting the RGB image to grayscale, too, as the thermal image is grayscale, and some methods also need the input to be grayscale. It is followed by preprocessing again to make the images as visually similar as possible, as the features to be extracted are based on the visual aspect. Color inversion has proved to be a crucial step here.

A typical pipeline for correspondence search starts with feature detection, followed by feature description, which results in vector-described features. This is precisely how Scale-invariant feature transform (SIFT) works. These vectors are then compared among images using a vector similarity with algorithms such as k-nearest neighbors, and tentative matches are thus made. SIFT is considered a baseline method in most computer vision experiments since it is robust and well-known by the community. In the experiments, SIFT delivered results similar to the other tested handcrafted methods, such as the ORB extractor (Oriented FAST and Rotated BRIEF). Correspondences obtained using ORB on a single image pair are shown in Figure 3.17. The other method shown in this chapter for comparison works similarly from a high-level point of view. Still, the first difference is that it is not handcrafted but built on two unsupervised deep-learning-based algorithms. The first part is SuperPoint (SP), which produces the features described as those mentioned for SIFT. However, matching is not based simply on vector similarity; it also takes into account some geometric restrictions. This is then a task of Graph Neural Network (GNN)-based SuperGlue (SG). The resulting correspondences for this method can be seen in Figure 3.18.

The next step is to estimate a projective transformation from the tentative matches. This is typically done using the RANSAC (Random Sample Consensus) algorithm, which randomly selects a number of points needed to compute the transformation. The transformation is computed using a closed-form solution, and then support from other correspondences is computed given some threshold (often in px) for a point to be considered a support

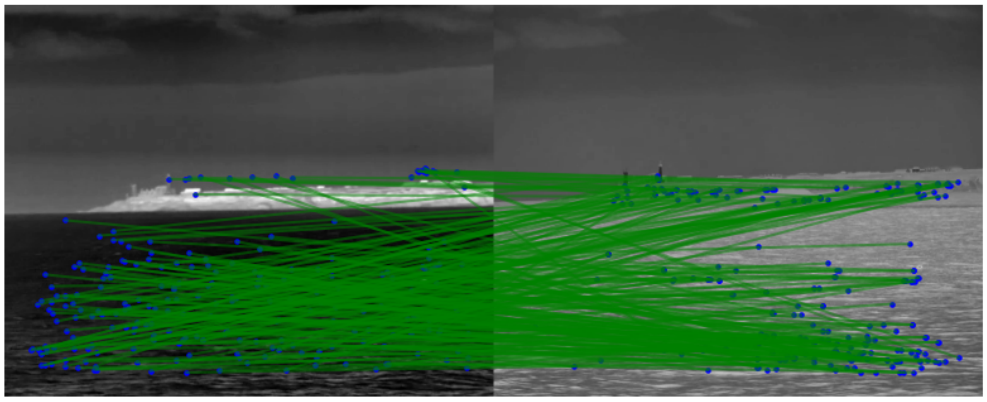
Figure 3.17: Correspondences found using ORB feature extractor and exhaustive matching on a single image pair.

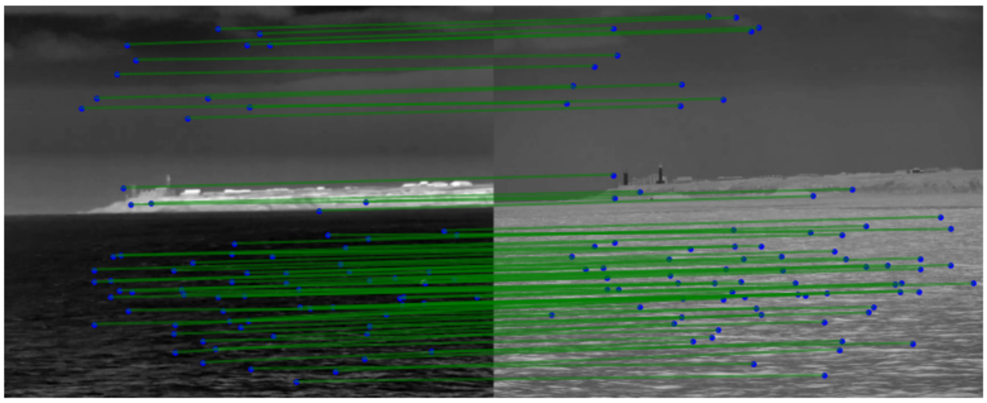
Figure 3.18: Correspondences found using SP feature extractor and SG matcher on a single image pair.

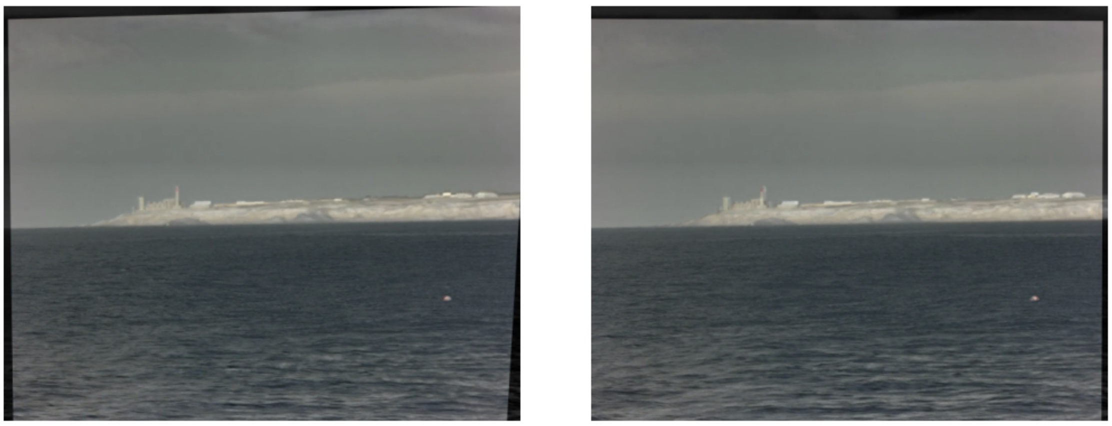

Figure 3.20: Transformations estimated form SP+SG correspondences. General projective transformation is on the left, affine on the right.

point. For more information on RANSAC, please refer to [52]. In these experiments, we used custom-implemented RANSAC for two reasons. Firstly, we want to be able to search for also different types of transformation than the general projective one, i.e., affine transformation (translation, scale, and rotation) or rigid transformation (no scale), since we assume that the pre-alignment tackles other deformations. For this, custom closed-form transformation solvers were implemented along with RANSAC implementation.

Moreover, since the images were pre-aligned, we can expect only a small magnitude of the transformation, and if the homography is transforming the image more than expected, we would like to keep the original pre-alignment instead of delivering some non-realistic deformation. This magnitude is easy to compute in the case of rigid transformation, where we can take an angle from the rotation part and the magnitude of the translation vector from the homography matrix and threshold these two values. In affine transformation, the scale also needs to be extracted as the determinant of the rotation part, which is also straightforward to the threshold. The decomposition of homography into affine transformation components is

$$H = \begin{bmatrix} \alpha R & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix}, \tag{3.9}$$

where α is the scale, R is the rotation matrix and $\mathbf{t}$ is the translation vector. However, in the case of general homography, it is not as straightforward to estimate the individual components. However, we can again use the abovementioned assumption and restrict ourselves to the affine transformation to overcome this problem. The experiments show that the results tend to be more robust and seem less overfitted to the given set of correspondences with the affine transformation. A comparison of these transformations can be seen in Figure 3.20.

Naturally, even this approach does not work for some image pairs, even quite frequently. Especially when images that are a bit hazy, the horizon is blurred, or images that do not contain any significant patterns, such as boats or waves, there are whole sequences where no correspondences are found. However, unlike in the previous FFT-based approach, there is a space for robustness and generalization improvement in the correspondence search. The space stems from the following observation. Once the transformation is estimated, it is usable throughout the whole sequence - in particular, when estimated as an affine transformation because the overfitted projective transformation has a higher error when generalizing to other images in the sequence. The images aligned and shown in Figure 3.22 were taken in the same sequence as the one in Figure 3.20 but using the transformation from those correspondences. With this observation, we can also eliminate the assumption that the non-perfect time synchronization is the cause of misalignment on our dataset and attribute it to the limited calibration.

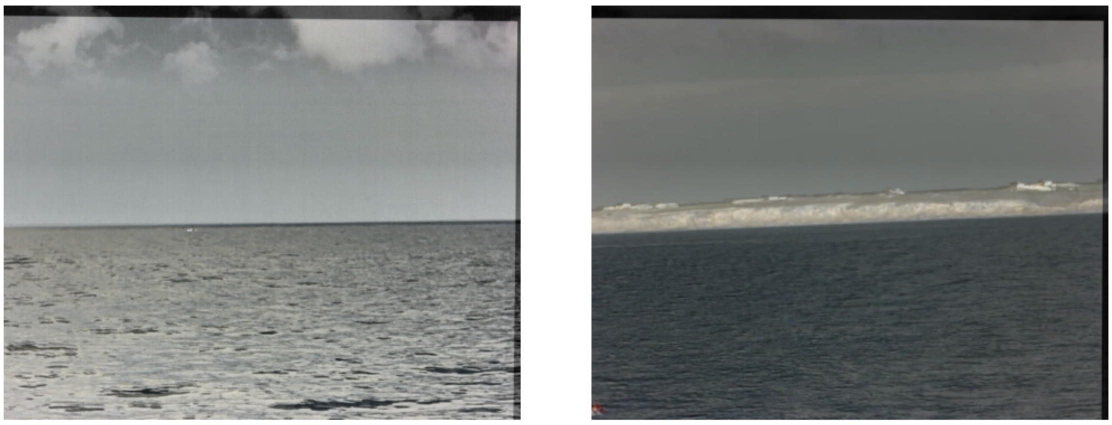

Figure 3.22: Image pairs transformed using affine transformation estimated on a different image pair in the same sequence.

This observation opens the door for a new method, which collects correspondences over more images in one trip, as long as the same device is used. Once enough correspondences are distributed all over the image, the estimate's uncertainty is sufficiently decreased, and the transformation parameters can be estimated. This estimate can then be used everywhere throughout the sequence and also align images, which would be unalignable if attempted separately. The example of correspondences collected throughout the whole sequence can be seen in Figure 3.23. Then, images aligned using transformation estimated from these collected correspondences are to be seen at Figure 3.25. On the left is a typical image that would be alignable even on its own, but the image on the right has no significant features and a hazy horizon, so its alignment using any of the explored approaches would be hardly possible, but with the correspondence-collecting method, we receive a sufficient alignment.

The approach developed in this section yields a homography H_F estimated from correspondences. This, in composition with the initial homography $H_{R \to T}$ estimated from calibration parameters creates the correct mapping

$$H^F_{R \to T} = H_F \cdot H_{R \to T}, \tag{3.10}$$

that can be used to warp RGB images to the thermal ones. Its inverse would again produce a transformation from thermal to RGB. Even though the development was shown solely on Sentry data, it was also tested on the Offshore datasets, leading to similarly satisfactory results. The result of RGB mapped to the thermal left image can be seen in Figure 3.26 and the other way round is shown in Figure 3.27.

This framework seems robust enough and sufficient for the needed ground-truth alignment. In need of higher precision, a method called GlueStick is also notable, which creates correspondences based not only on points but also on lines detected in both images. It would be natural to detect the horizon in the image and use it for more precise rotation estimation using such an algorithm. However, the horizon is often not clear enough since images can be hazy or hardly visible due to a shady border between land and water.

Now, after estimating the secondary homography we use after the pre-alignment from camera parameters, we can construct a full projective transformation from RGB to thermal

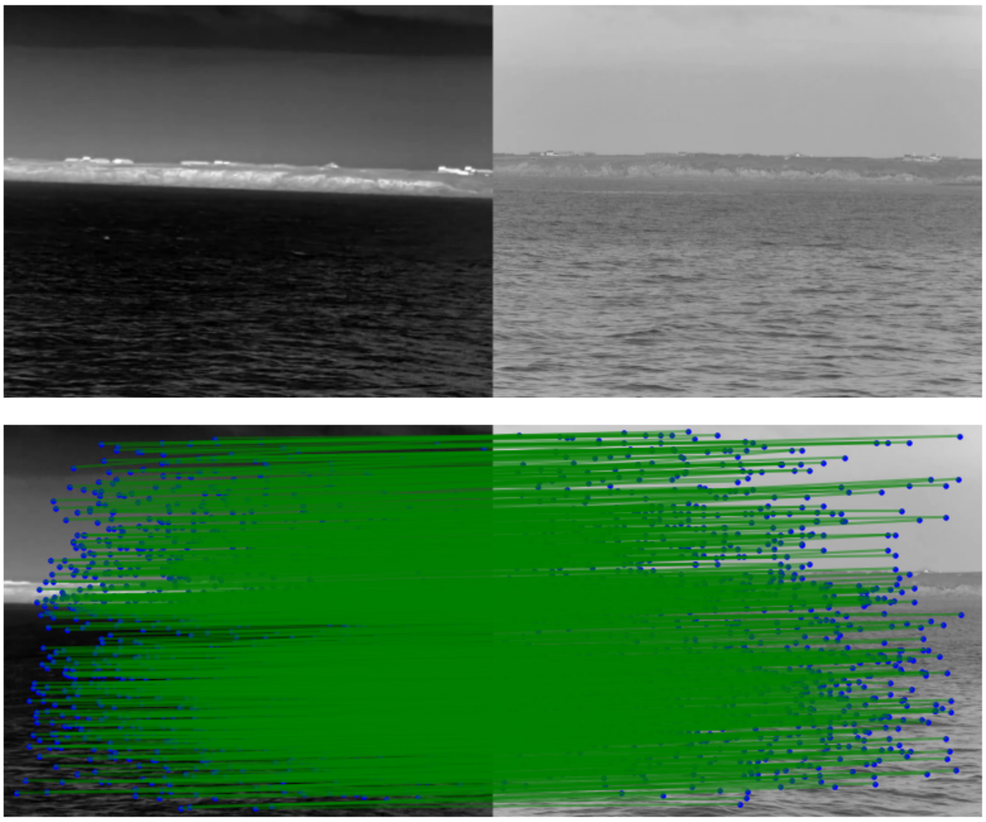
Figure 3.23: Correspondences found using SP+SG over a sequence of 40 images. The images are also shown at the top for better visibility.

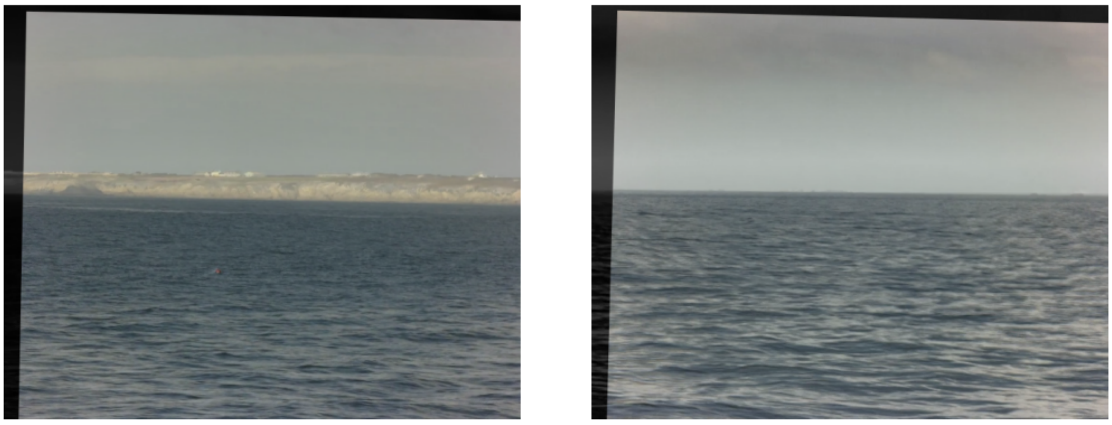
Figure 3.25: Image pairs aligned using transformation estimated from correspondences collected over 40 images.

image space (or vice versa) as shown in Eq. 3.10. This homography can be used to transform the ground-truth (GT) masks, such as bounding boxes, in the case of the SEA.AI dataset. Examples of GT annotations for RGB and thermal images, respectively, can be seen in Figure 3.29.

In our case, the annotations are bounding boxes. Therefore, the ground truth can be

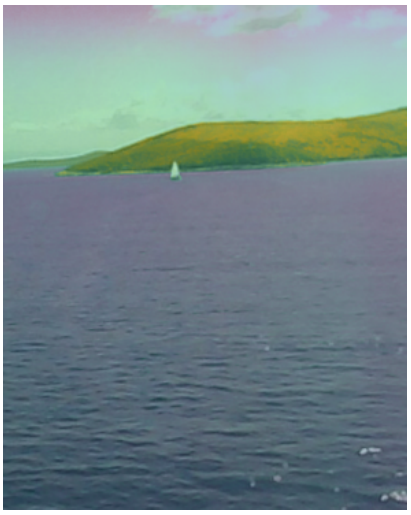

Figure 3.26: Overlapped images after applying finetuned $H^F_{R\to TR}$.

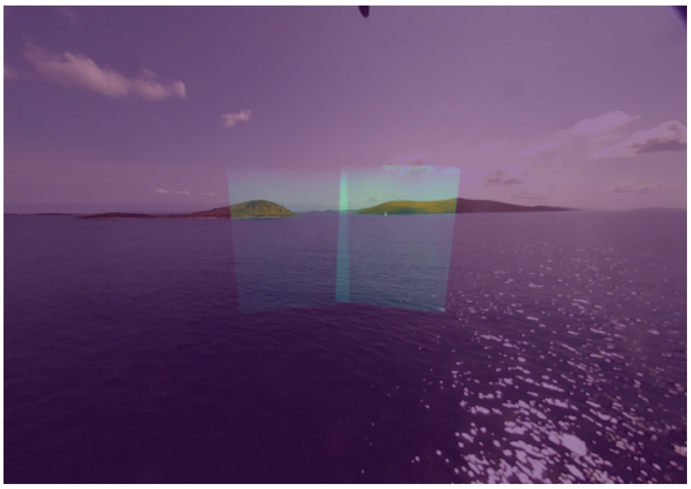

Figure 3.27: Overlapped RGB and both thermal images after applying finetuned $H^F_{T\to R}$.

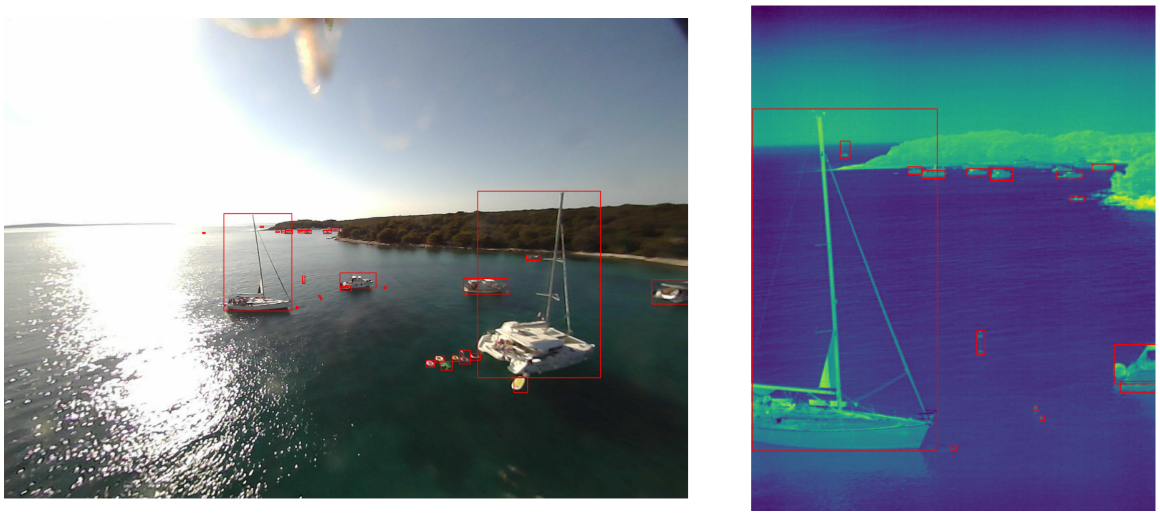

Figure 3.29: Example of ground truth bounding boxes for RGB and thermal images.

transformed from RGB to thermal by simply using the $H^F_{T\to R}$ for each corner of the bounding box $b = (b_x, b_y)$ and normalization as

$$\lambda \begin{bmatrix} b^t_x \\ b^t_y \\ 1 \end{bmatrix} = H^F_{R\to T} \cdot \begin{bmatrix} b_x \\ b_y \\ 1 \end{bmatrix}, \tag{3.11}$$

where λ is a normalization factor and $b^t = (b^t_x, b^t_y)$ are boudning box corner coordinates in the thermal image. It is only necessary to crop the bounding boxes so that they do not fall outside of the images, which is easily done for each corner as

$$\begin{bmatrix} b^f_x \\ b^f_y \end{bmatrix} = \texttt{min(max(} \begin{bmatrix} b^t_x \\ b^t_y \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \end{bmatrix}), \begin{bmatrix} W \\ H \end{bmatrix}), \tag{3.12}$$

where H and W are the height and width of the new image size, while `min()` and `max()` are element-wise vector functions. GT masks propagated from both modalities to both the RGB and thermal images are shown at Figure 3.31.

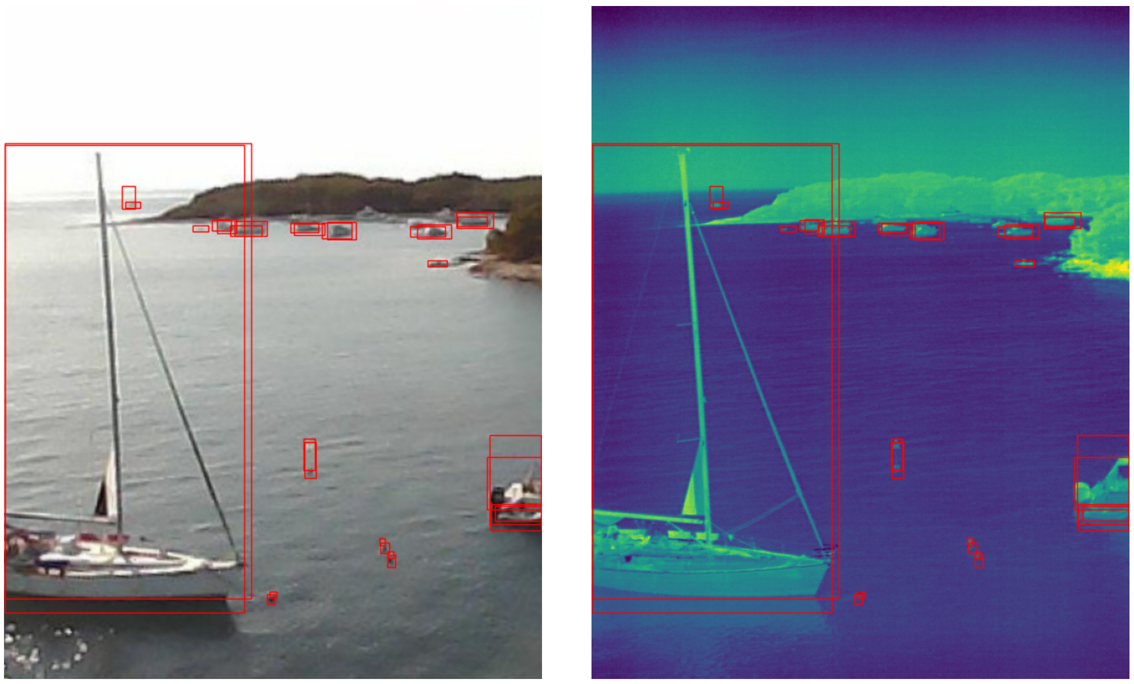

Figure 3.31: Ground truth boxes projected into RGB and thermal images respectively from both modalities.

3.5 Distilling the data

At this point, we have aligned data tuples with ground truth targets. However, the number of annotated images tuples created this way is close to 700,000. Training the number of models benchmarked in Chapter 5 would be infeasible with this amount of data. Moreover, the data mainly comprises sequences, meaning many very similar frames are in the dataset. Therefore, the next goal is to reduce the amount of available data while maintaining the information provided in the dataset. Because of the amount of sequences in the dataset, a natural approach that comes to mind is simply subsampling. In other words, select some sampling period N and then select every N-th sequence image. Subsampling would result in the desired size of the dataset provided N is chosen correctly. While this is a simple to implement and effective approach, it ignores the need for some more dynamic N, as some scenes are very short and might be easily missed. On the other hand, if the ship is motionless, it still collects many almost identical images. Dynamic N is also needed as the objects are moving at different speeds, and some of them might be missed completely or in a very unrepresentative position. In contrast, the slow objects would appear in many frames, thus depriving the dataset of its richness. A better approach would be to use some variant of box tracking algorithm, such as KCF [5] to track the annotated bounding boxes across images and, if some important change in annotated scenery occurs, to use that image in the training dataset. While this approach delivers more of what we seek, it still needs to consider the unannotated aspects of scenery. These caveats lead to the approach used in this work. The dataset images are processed using embeddings. Embeddings are vectorized representations of images (in our case, but are often used also in natural language processing (NLP) as vectorized versions of tokens) that exhibit certain properties, such as that embedding of visually similar images should be close (with respect to some metric defined in the space of the embeddings) to each other or not close for images that are different. Let's define an encoder as a function

$$f_e : [0,1]^{3\times H\times W} \to \mathbb{R}^N, \tag{3.13}$$

where $[0,1]^{3\times H\times W}$ is the RGB image space and R^N is the embedding space, with H being image height and W image with and N a dimension of the embedding space that depends on the encoder used. In this function, a visual transformer-based encoder trained on the cityscapes dataset with $N = 512$ was used as the encoder. After computing these embeddings for each image, we can perform some dimensionality reduction procedure to cluster the embeddings in some domains, where we can visualize the embeddings and verify the similarity of images in the clusters. Methods typically used for dimensionality reduction are based on SVD decomposition and can be Principal Component Analysis (PCA), Uniform Manifold Approximation and Projection (UMAP), or t-distributed Stochastic Neighbor Embedding (TSNE), with the latter used in our case for trips longer than 100 samples and UMAP otherwise. All the methods are introduced and explained in greater detail at [51]. The embeddings of one sequence from the offshore dataset projected into a plane are shown in Figure 3.32.

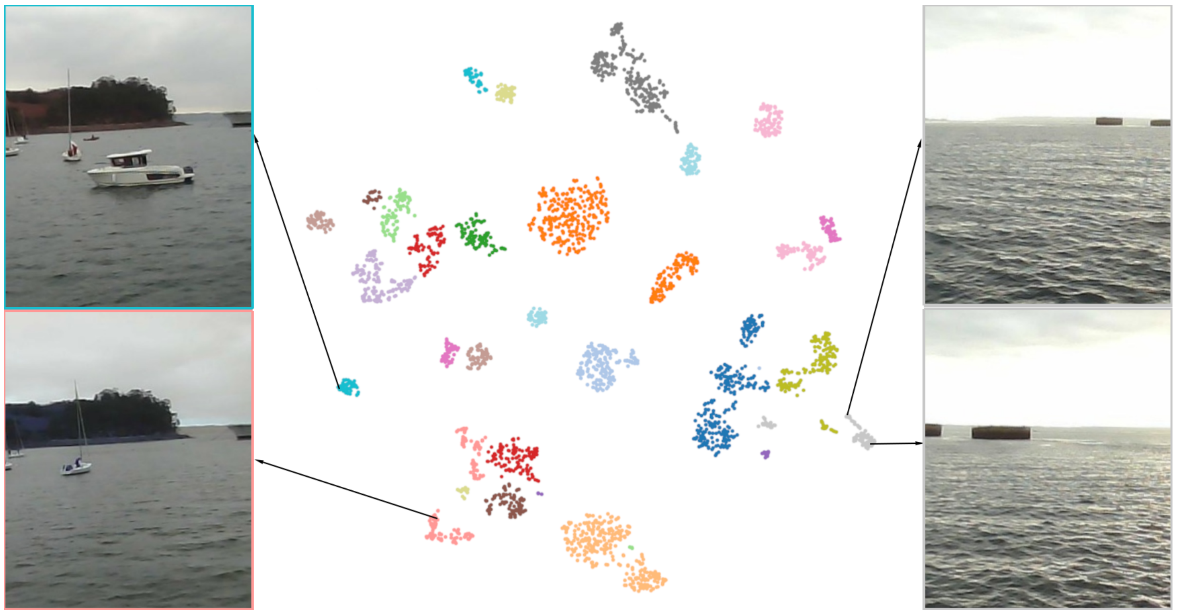

Figure 3.32: 512-dimensional embeddings of images from one offshore dataset trip projected into a plane and clustered using DBScan.

After projecting the embeddings into lower dimensional space, we can perform clustering (this is also more challenging in higher dimensions due to the sparseness of the vectors in that space and higher computational costs). Clustering is basically a method of grouping closeby data points together. Typically used methods are K-Means, DBScan, or AffinityPropagation, all described in Ref. [51]. In the case presented here, DBScan is selected as the main clustering algorithm due to its speed and the fact that the number of clusters does not need to be specified in advance and thus results in the natural clustering of a dataset depending on method parameters, which are ϵ and minima number of samples per cluster. Automatically selecting a number of clusters is also what is expected from an algorithm for automated dataset distillation. The clustering of the embeddings of images from one Offshore dataset trip is shown in Figure 3.32. Visualized are also image examples from some clusters to outline the algorithm's efficacy. However, this step leaves the user with very coarsely clustered embeddings, and if one were to select just a single image from each cluster, it would reduce the size of the dataset by a factor of hundreds. Moreover, how should one choose a single sample representative of all the hundreds of samples when the samples still vary inside a single cluster, as seen in Figure 3.32. Since we operate in a clustered vector space, choosing a centroid or an image closest to the centroid is a very natural answer. While this addresses the selection question, the too-big reduction factor and, from it, stemming limited representativeness of the selected sample is resolved with a second round of clustering. These main clusters can be divided into subclusters of virtually the same images, among which one representative is selected as the one closest to the centroid. The number of chosen subclusters in this stage can be enforced to ensure at least some specified reduction factor by using an algorithm such as KMeans, the results of which are shown in Figure 3.34 and where K is the input parameter that can be computed as $K = N_c/f_r$, where N_c is a number of images in a cluster, and f_r is a reduction factor. Alternatively, the number of subclusters can be defined automatically by a clustering algorithm that does not need it as an input, such as DBScan or Affinity Propagation. Subclusters obtained with Affinity Propagation are shown in Figure 3.33 with the visualizations capturing examples of the images to demonstrate the quality of the proposed method. Selection between the subclustering method is more of an empirical trait as it depends on the usage of the final dataset, the number of samples on the input, and the nature of the dataset.

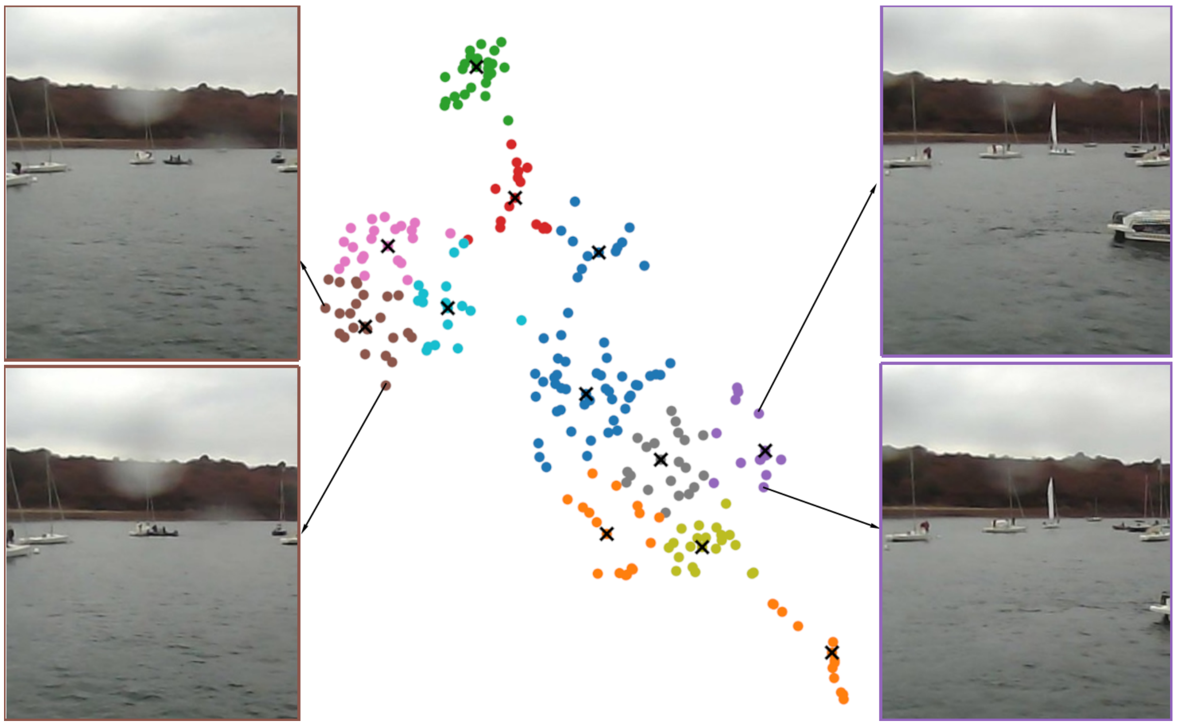

Figure 3.33: One cluster obtained by DBScan subclustered with Affinity Propagation and centroid samples marked with x.

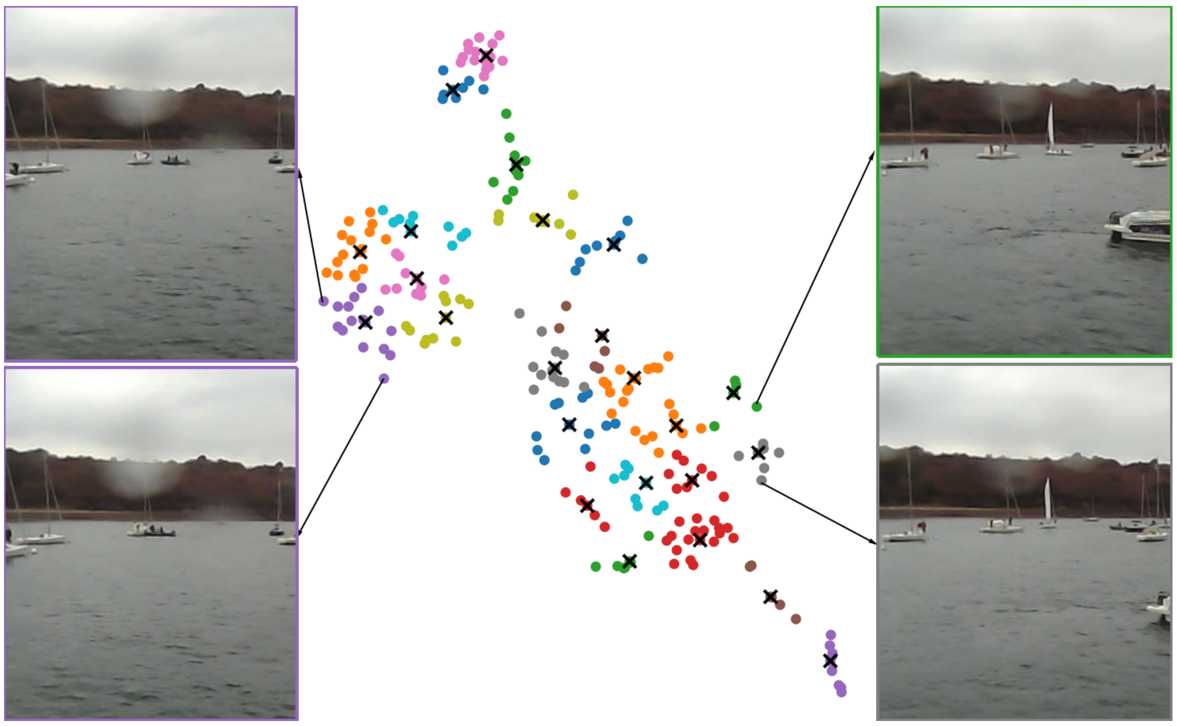
Figure 3.34: One cluster obtained by DBScan subclustered with K-Means and centroid samples marked with x.

3.6 Resulting dataset

There are only a few videos labeled in both thermal and RGB channels in the Sentry dataset. Therefore, these are reserved for testing purposes. The bulk of the data comes from the offshore dataset, where the initial count is 1M image triplets (RGB, thermal left, and thermal right). These were then filtered to the ones captured on devices with calibration data available. The filtered triplets were then all pre-aligned using these calibration parameters. Then, correspondences were collected for images in every trip separately, one trip typically consisting of a few thousand triplets. After gathering at least 3500 correspondences for each of the two pairs (RGB left projection with thermal left image and the same for right), the finetuning affine transformation H^F was constructed and used to transform every second RGB into left and right along with its annotation, thus resulting in two RGB and thermal image tuples with merged ground truth. For each trip, a visual sanity check was performed to make sure the transformation was successful (it was not always the case, as some trips were not properly synchronized or too textureless to collect any correspondences). All these steps combined resulted in approximately 350,000 annotated image pairs. RGB-only embeddings were computed using a visual transformer-based encoder for images of each trip. These RGB 512-dimensional embeddings were projected into 2D with TSNE (or UMAP for the sequences under 100 samples) and clustered using DBScan with parameters of $\epsilon = 2.1$ and a minimal number of samples per cluster selected as one since single outliers are an excellent addition to the final dataset. Each cluster was then subsampled using AffinityPropagation with a damping parameter of 0.65. An example of the annotated pairs from the resulting dataset can be seen in Figure 3.35. The final dataset numbers around 16,000 samples.

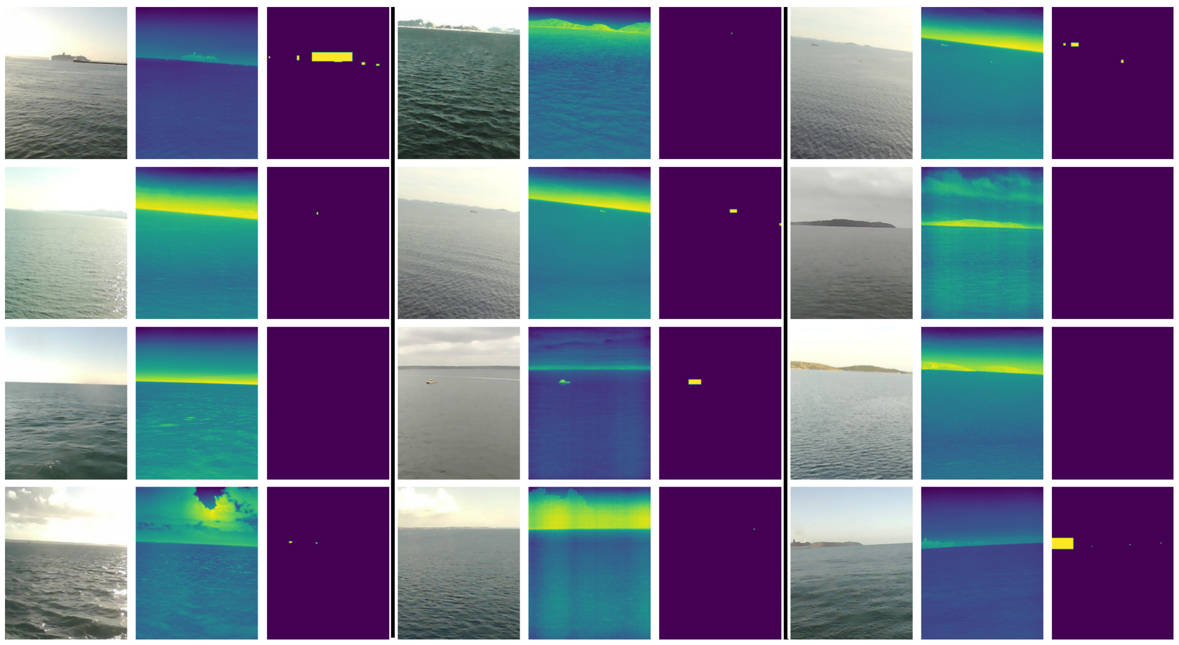
Figure 3.35: 12 RGB and thermal pairs with merged ground truth masks from the distilled and aligned dataset.

Chapter 4

Approach overview

This chapter details the architectures that are benchmarked in the next chapter of this thesis. Some of the architectures were already outlined in Chapter 2, and these are described here in more detail. Nevertheless, this chapter also introduces our proposed and baseline architectures formerly used in the SEA.AI devices. Firstly, architectures for multimodal semantic segmentation are discussed in Section 4.1, and then their trianing details are discussed in Section 4.2.

4.1 Multimodal semantic segmentation

4.1.1 U-Net based architectures

U-Net is a cornerstone architecture for semantic segmentation and also serves as a baseline for our benchmark. The initial applications of U-Net were in biomedical image segmentation, such as brain image segmentation [42]. Nevertheless, U-Net implementations have also found their use in other fields, such as physical sciences [3].

It is based on encoder-decoder architecture, where the encoder extracts features from the input image, and the decoder upsamples the features to the original image size. The encoder-decoder architecture is connected by skip connections, which allow the decoder to use the features from the encoder at the same level. These connections allow the network to learn both high-level and low-level features. The depth (i.e., the number of downsampling blocks) and channels obtained in each depth vary among implementations and available computational resources. A diagram of our implementation of the original U-Net architecture is shown in Figure 4.1.

The Double 2D Convolution operation consists of two 2D convolutional layers with a kernel size of 3×3, a stride of 1, and a padding set to preserve the resolution, followed by a rectified linear unit (ReLU) non-linear activation function. It is an industry-standard to use a batch normalization layer after the convolutional layers, which is also used in our implementation, but this differs from the original implementation. Batch normalization learns parameters of batch mean and standard deviation to normalize them to 0 and 1, respectively. This stabilizes the training procedure and acts as a regularizer, which helps to prevent overfitting [40]. The Maxpooling operation is a 2×2 layer with a stride of 2, which selects the maximal value from the 2×2 window. This operation can be replaced by a stride of 2 in the first convolution operation in the subsequent Double 2D Convolution following the original Maxpooling operation. An upsampling with linear interpolation and factor 2 is used for the Upsampling operation, followed by a single 2D convolution layer. This upsampling layer can be replaced by transposed convolution. Lastly, in place of the 2D convolutional layer mentioned so far, one can use a normal 2D convolution layer or something called separable convolution, which

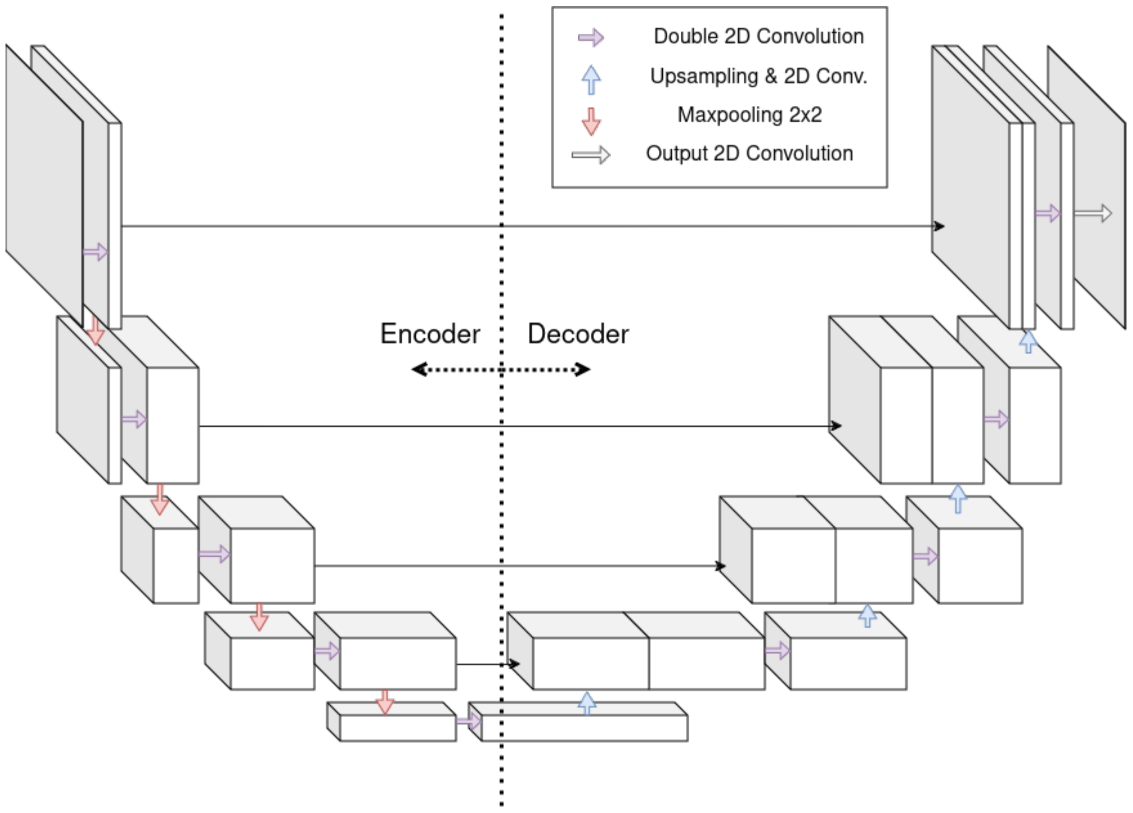

Figure 4.1: Our variation on the original U-Net architecture.

is a depthwise convolution followed by a pointwise convolution. The differences between these types of convolution are shown in Figure 4.2. The main advantage of separable convolution is the significant reduction of the number of parameters and, therefore, computation costs. This reduction was leveraged in MobileNet [32]. Both of the 2D convolution versions are more explored in the benchmark. The number of channels and specific resolution values are intentionally omitted in the schematic, as it differs across the UNet version used in this benchmark.

Now, to obtain an architecture capable of multimodal data processing, the modalities need to be fused at some point. The fusion can then be approached in several ways. The first aspect that is often different among the architectures for the fusion is the point in the processing pipeline where the features are fused. In this light, fusion can be divided based on the part of the stage where they are fused into pre-encoder or input fusion, multilevel fusion, and output fusion. These strategies are visually overviewed in Figure 4.3.

A typical input fusion approach included in most publications, at least as an ablation study, is adding the other modality as an additional channel to the input RGB image, thus creating a 4-channel image input image $I_{in} \in [0,1]^{H\times W\times 4}$ instead of typical $I_{RGB} \in [0,1]^{H\times W\times 3}$. The 4-channel approach was explored in ablation studies, e.g., [22] or [24], but was shown to be inferior to the other fusion strategies. The rest of the fusion approaches introduce separate encoders for each modality and then fuse the features at some point. The output fusion was used in the early works, such as [29], or ablation study of [24]. Even though it led to some improvement over the RGB-only model or 4-channel approach, it was shown to be inferior to the multilevel fusion [24]. Multilevel fusion is then the one generally adopted in the multimodal

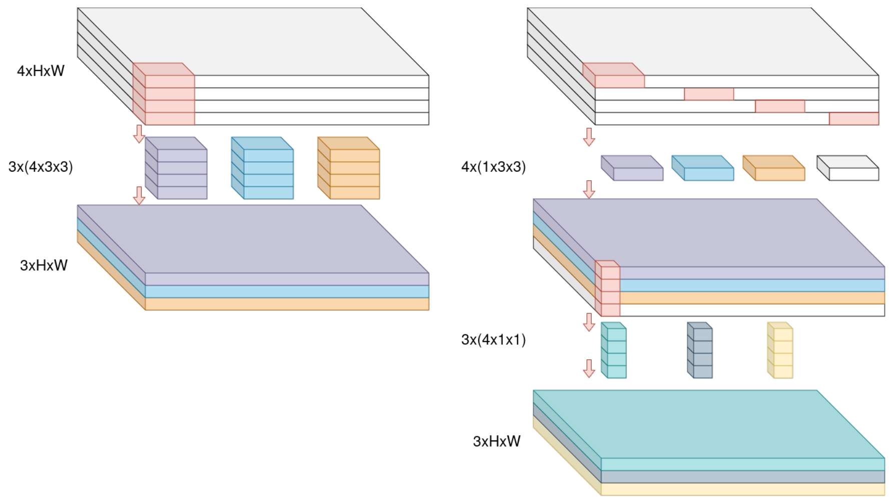

Figure 4.2: 2D separable convoltion vs. normal 2D convolution with 3x3 kernel, 4-channel input and 3-channel output. The number of parameters in this setup is 108 vs. 48.

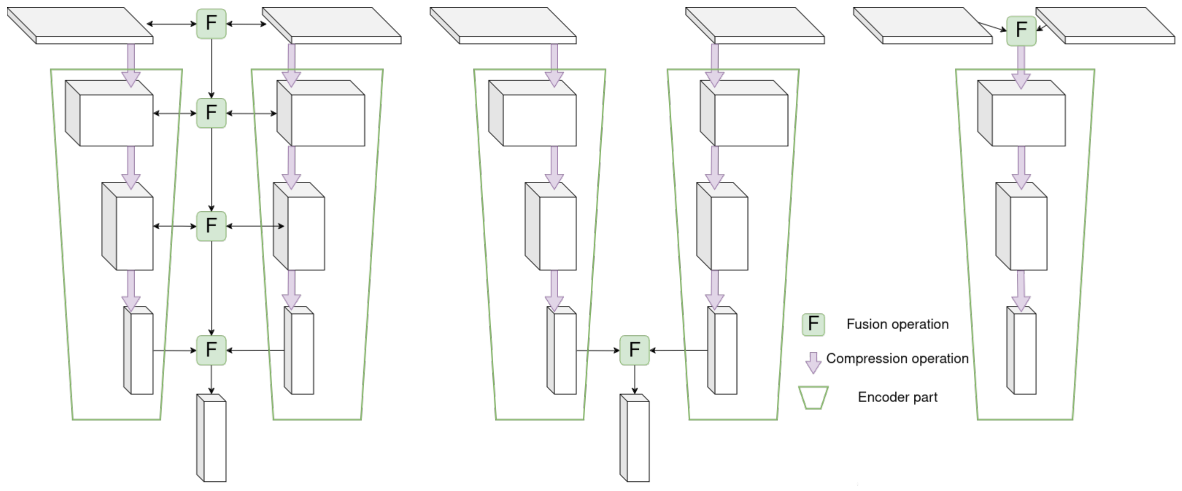

Figure 4.3: Typical architecture concepts for processing multiple modalities: multilevel, output, and input fusion.

field [7], [22], [24], as overviewed in Chapter 2.

We mentioned previously that the double-encoder approach requires a fusion strategy to combine the features from both encoders. Therefore, another differentiation aspect among the architectures stems from the fusion strategy itself, i.e., it is based on how the features are fused independently of where they are fused in the architecture. Some architectures even use multiple fusion strategies at different depth levels [14].

A general double-encoder U-Net (W-Net) architecture with a multilevel black-box fusion strategy is shown in Figure 4.4, with the fusion strategy denoted by F block. This architecture was inspired by U-Net and the prominent architecture features leveraged in multimodal computer vision. Note that U-Net architecture contains skip-connections (connections propagating the encoded features to the decoded ones on the same depth level) at each level so that the encoded fused information from both modalities is propagated at each level to the corresponding decoder level, and this is the also the case for the dual-encoder U-Net. These connections are also displayed in the figure.

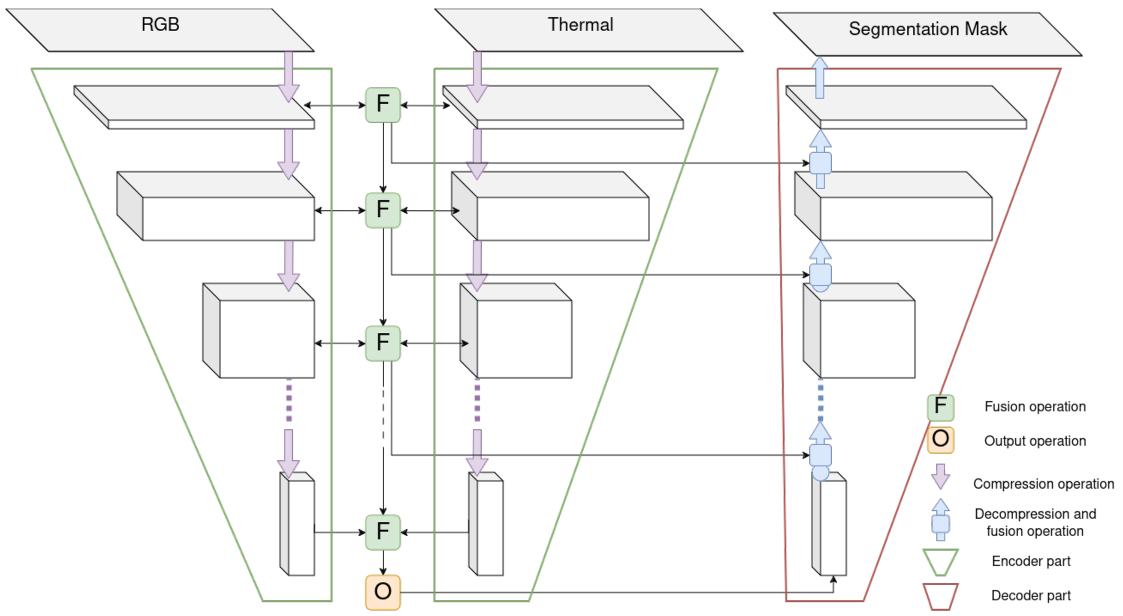

Figure 4.4: U-Net double encoder architecture.

The multiple dots in the figure represent the option of a deeper encoder. Note that the encoder and decoder are also, in this general figure, more of a schematic black box, as there is no particular restriction on the compression or decompression strategy or number of channels produced in each layer. Now, as we have introduced the WNet architecture, we can compare it to the state-of-the-art architectures and see that most of them can be reorganized to a variation of this schematic architecture; in particular, all of the architectures presented in this chapter fit the outlined scheme. One of the minor differences in the strategies is that only some outlined connections are used or that sometimes there are more outputs from the decoder block for multiple losses taken from different decoder levels. The latter applies to, e.g., GMNet [14] or CAINet [1]. The more significant difference is the encoder backbone, where the most common are ResNet, DenseNet, or some visual transformer, but this is not specified in the schematic either, so all of these options can be used for WNet, too. The encoder is then the primary determinator of computation capacity and the number of parameters of the model. These aspects stem from the encoder depth, the number of channels in the layers, and the convolution type. Nevertheless, the most significant contribution of the individual architectures is the fusion strategy, i.e., the F block in our figure.

This fusion can be simply an element-wise addition of the features, their concatenation, or any more complex approach. A more complex fusion block can have even trainable parameters and any number of concatenations, summations, multiplications, flattenings, and similar operations compressed into one block. This block can even produce more than one output, which is used for different purposes, such as fusing into modality encoders, deeper layers, or decoders. The fusion strategies are presented in more detail in the following sections corresponding to the models that introduced them.

Nevertheless, UNet and its double-encoder WNet version are used in several flavors in the benchmark; the tweaks to the architectures defining these flavors are marked with model name suffixes. Firstly, UNet is used as a simple baseline with simple 1-channel input: a thermal image and channel numbers of $\{32, 64, 64, 128, 256\}$ and separable convolutions everywhere besides the first and last one. This network is named UNet-S in the benchmark, where the S stands for separable convolution. The thermal image is chosen over RGB, as it is (i) more robust to adverse environmental conditions, (ii) has a higher resolution, and (iii) is used in the single-modality SEA.AI architecture, which is also included in the comparison. Secondly, the already discussed 4-channel version of this UNet is also included in the benchmark named UNET-S4. Additionally, we include a version of the UNet that uses only RGB images, named UNet-S3, where the number 3 in the suffix stands for three channels of RGB. The other architectures tested in the benchmark are various WNet flavors. In all of them, both encoders are taken from the UNet implementation and, in some, the decoder, too. The only difference is that the last upsampling block in the decoder, whose function is replaced by a simple upsampling of the result, is omitted in the same way as is described in LNSeg Section 4.1.2. This replacement significantly reduces the number of parameters and computation costs, while our experiments showed that the performance is not significantly affected. In the first flavor, the fusion strategy is simply an addition of the features from both encoders on all levels, which does not change the dimensionality with regard to the UNet implementation, so the decoder with concatenation can be used directly. This architecture is almost identical to the one used in [22]. This model is called WNet, and benchmarked is also WNet-S, a version with separable convolution employed, which is denoted by the suffix S. There is also a version named WNet-NS that does not use separable (NS) convolution but normal 2D convolution. The last of these simple models is WNet-S-DP, which is a deeper version with $\{32, 64, 128, 256, 512\}$ channels, deeper stands for DP. Moreover, the modularity of our implementation allows for a combination of WNet with encoders or fusion strategies from other models. However, these will be introduced in more detail in their respective sections. All the models mentioned in this subsection were trained using binary cross-entropy loss [36] with logits, where logits are the raw output of the network before the activation function is applied. Logits are used to increase the numerical stability [36].

4.1.2 LNSeg

LNSeg [19] is an architecture formerly used in the SEA.AI Offshore devices, and it now serves as a baseline for our experiments. Firstly, it is a suitable baseline as its complexity and inference speed are good indicators of the performance needs of the SEA.AI edge devices. Moreover, we can use the model with both the original weights, trained on the full thermal segmentation dataset comprising 400,000 images (LNSeg-ORI), and the weights trained on our distilled dataset version to compare the distillation effect on training. Note that this model was trained and evaluated on 16-bit thermal data, while the benchmark models are trained and evaluated on 8-bit data, so the comparison is not entirely fair, but it is a rough representative of the effect of the dataset distillation. As mentioned, this architecture is deployed only on thermal images because thermal images are a more robust detection source. Because of the higher robustness and absence of a calibration good enough for sensor fusion, the thermal image was used as the only source of detections in the areas where thermal information is available and RGB otherwise. The differences in the fields of view are shown in Section 3.3. Therefore, it makes sense to compare the performance of the thermal-only model with the multimodal models, as they would replace LNSeg at the same place in the pipeline, where both modalities are available.

The LNSeg architecture is a variation of a simple UNet architecture with a single encoder and decoder. It can almost fit onto the schematic in Figure 4.1 with minor tweaks shown in Figure 4.5. Firstly, downsampling is performed with a stride of 2 in the first convolutional layer instead of Maxpooling. The Double 2D Convolution operation is replaced with Triple 2D Convolution. The upsampling is done with the nearest upsampling layer instead of bilinear upsampling, followed by a single 2D convolution layer. Moreover, the last upsampling block is omitted, and the second to last upsampling block is connected to the output layer. The output is then interpolated to the original resolution. The lack of the last upsampling block, along with the separable convolutions, significantly reduces the parameter number and computation costs. The number of channels is $[32, 64, 64, 128, 256]$. Lastly, the concatenation from UNet is replaced by an element-wise addition. This architecture is named LNSeg-S in the benchmark. There is also a version with 4-channel input, LNSeg-S4, which is used for comparison with the other architectures. The last version is LNSeg-S3, which takes as an input only RGB images. All the models mentioned in this subsection were trained using binary cross-entropy loss with logits.

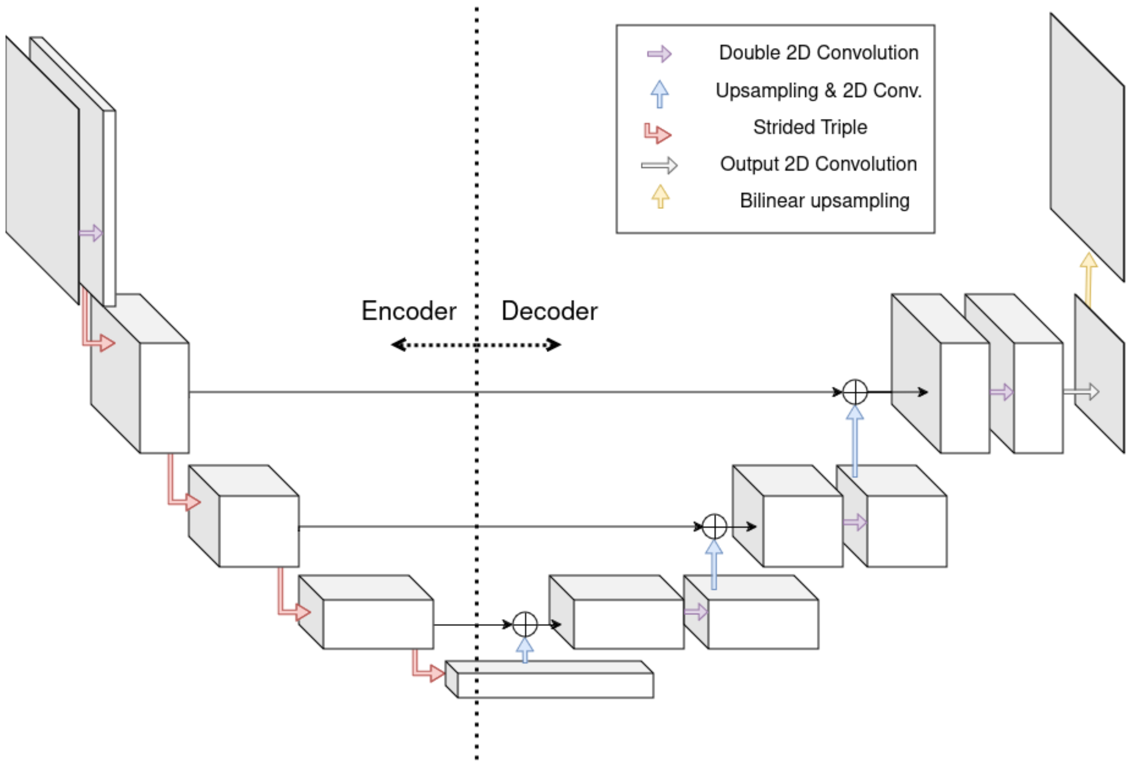

Figure 4.5: Variation on the original U-Net architecture used in LNSeg.

4.1.3 RTFNet

RTFNet [24] is one of the early architectures that introduced the multilevel fusion strategy for RGB and thermal images. The main distinction from the general WNet scheme is that the thermal image is fused in a one-way manner into RGB at each decoder level using a simple element-wise addition. The output of the RGB encoder is then processed by several upsampling blocks, but there are no skip connections from the encoder levels. The encoder backbone is ResNet, and to keep the architecture to the complexity limitations posed by the SEA.AI devices, we trained the network with a ResNet18 backbone. While benchmarking ResNet, we introduced a few modifications to the original architecture, such as the number of input layers, where the thermal encoder was initially designed with three channels but has only one channel in our case, and the output layer, where the number of classes was set to $C = 1$. The number of channels was changed for all the architectures in this section. Moreover, the original architecture was terminated with a ReLU layer, which prevented us from training with binary cross-entropy loss with logits. Therefore, another output layer was added to the architecture This architecture is named RTFNet18 in the benchmark, where the 18 in the name stands for the size of ResNet used. For more architecture details, please refer to the original paper [24]. Both models mentioned in this subsection were trained using binary cross-entropy loss with logits.

4.1.4 GMNet

GMNet [14], in comparison to the previous architecture, introduced more complex fusion strategies, namely Shallow and Deep Feature Fusion Modules (SFFM and DFFM). The SFFM blocks are used in the first two levels of the decoder, and the DFFM blocks in the last three. The schematics and further architecture details of these blocks can be found in the original paper [14]. As the first architecture overviewed in this work, GMNet introduces multiple training losses, particularly boundary loss, binary mask loss, and semantic mask loss. The boundary loss is computed using binary cross entropy loss between the boundaries of the predicted masks and the ground truth masks regardless of class. This loss is omitted in our training, as the boundaries are vague rectangles around the objects, and it is not an essential part of the detection process. The binary mask loss is then computed using binary cross-entropy loss between the predicted binarized masks regardless of the class and the ground truth binary masks. This loss is kept with our training, but we use binary cross-entropy loss with logits. The semantic mask loss is computed using Lovasz softmax loss [25] between the predicted semantic masks and the ground truth masks. The semantic, boundary, and binary loss weights used in the original implementation are $[0.4, 0.3, 0.3]$. Even though we use only a single class, we keep the semantic loss because it is taken from a different part of the architecture than the binary loss and uses a different function. However, we use Lovasz hinge loss instead of Lovasz softmax loss, as the outputs are only binary, and Lovasz hinge loss is designed for binary outputs [43]. We keep the ratio of the two used losses the same as in the original implementation. The architecture schematic is shown with our tweaks in Figure 4.6. Again, the encoder backbone for GMNet architecture is ResNet. In the benchmark, there are several flavors of GMNet, namely GMNet18, a dual-loss-trained GMNet with a ResNet18 backbone. Then GMNet18-1L, where 1L stands for a single loss, is the same architecture with binary loss omitted, and finally, GMNet34, which has a ResNet34 backbone.

4.1.5 CMX

CMX [7] is the only transformer-based architecture in the benchmark. Specifically, we trained CMX versions with two backbones, SegFormer [13] MiT-b0 and MiT-b2, named CMX-b0 and CMX-b2 in the benchmark where the suffixes b0 and b2 stand for the backbone size. CMX also introduces two fusion strategies: the Cross-Modal Feature Rectification Module (CMFRM) and the Feature Fusion Module (FFM). The CMFRM is used at all levels of the encoder for cross-modal information flow before propagation to the next level. In contrast, the FFM is used for the fusion of the modality data before feeding it into the decoder. The CMX paper [7] also delivers implementations of several decoders, namely Fully Connected Head Network (FCN), Multi-Layer Perceptron (MLP), and Atrous Spatial Pyramid Pooling (ASPP), which was introduced in [28]. The decoder used in the final implementation of CMX

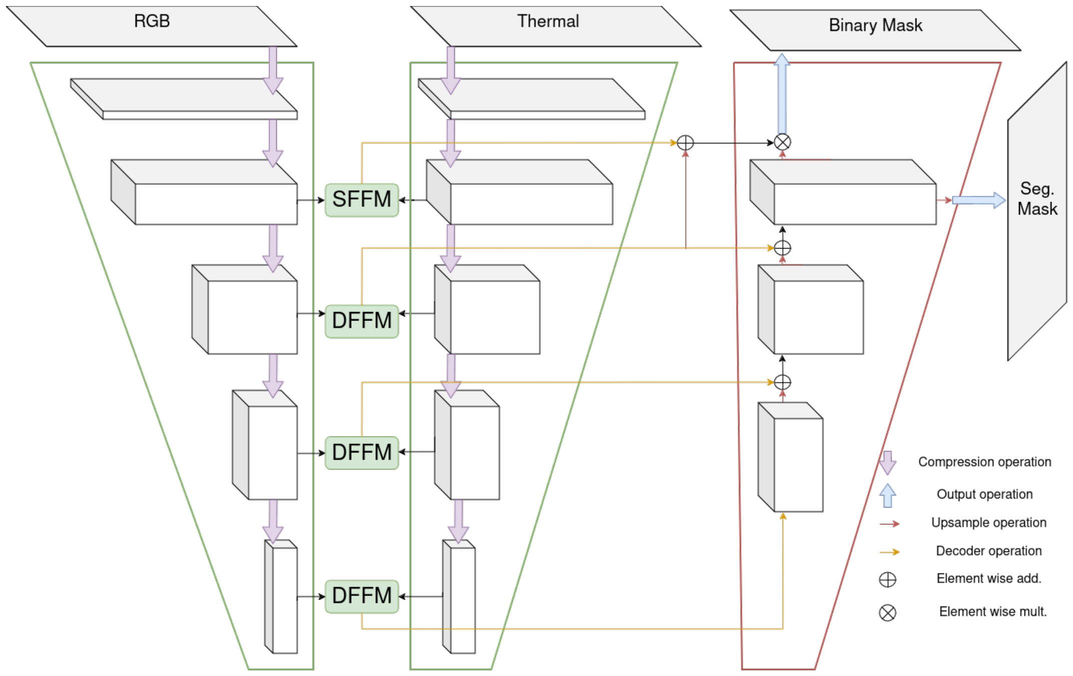

Figure 4.6: GMNet variation with custom MLP encoder on our proposed WNet.

is the MLP, which we also used in the CMX networks trained for our benchmark, as the other two encoder heads were inferior to MLP in our experiments, too. A Schematic of the CMX architecture with MLP decoder is shown in Figure 4.7, as the MLP schematic is missing in the original paper. In the schematic, the 'Map to embedding space' connection represents a linear layer that maps the input so that the number of channels is a pre-specified embedding dimension. This block is then upsampled to the spatial resolution of the first encoder layer so that the outputs from all layers can be concatenated in the channel dimension, resulting in four times embedding dimension number of channels. Nevertheless, please refer to the original paper for a more detailed description of the architecture or other fusion strategies. Moreover, the results CMX presented in the paper were reinforced by promising results delivered in our initial experiments, so we decided to replace the backbone with our UNet separable convolution version to reduce the number of parameters and computation costs and train a WNet version with custom fusion or head. Therefore, there is WNet-CX architecture in the benchmark that uses the CMX fusion strategies with the WNet's $\{32, 64, 64, 128, 256\}$-channeled encoder and decoder. Other CMX-WNet crossover architectures are WNet-MLP, which replaces in WNet the previously employed WNet encoder with the MLP encoder from CMX, and WNet-CMX, which uses both the MLP encoder and the CMX fusion strategies. All the models mentioned in this subsection were trained using Binary cross-entropy loss with logits.

4.1.6 SA-Gate

SA-Gate is an architecture initially developed for RGBD semantic segmentation. However, as it showed promising results in the RGBD domain, we also tested it in the RGBT domain. It is yet another architecture with the ResNet backbone, but it introduces an attention-based fusion strategy. This architecture focuses on the noisy features of cross-modalities by introducing attention mechanisms. The attention mechanisms have already been used in computer vision tasks to select the most representative and informative regions of input signals.

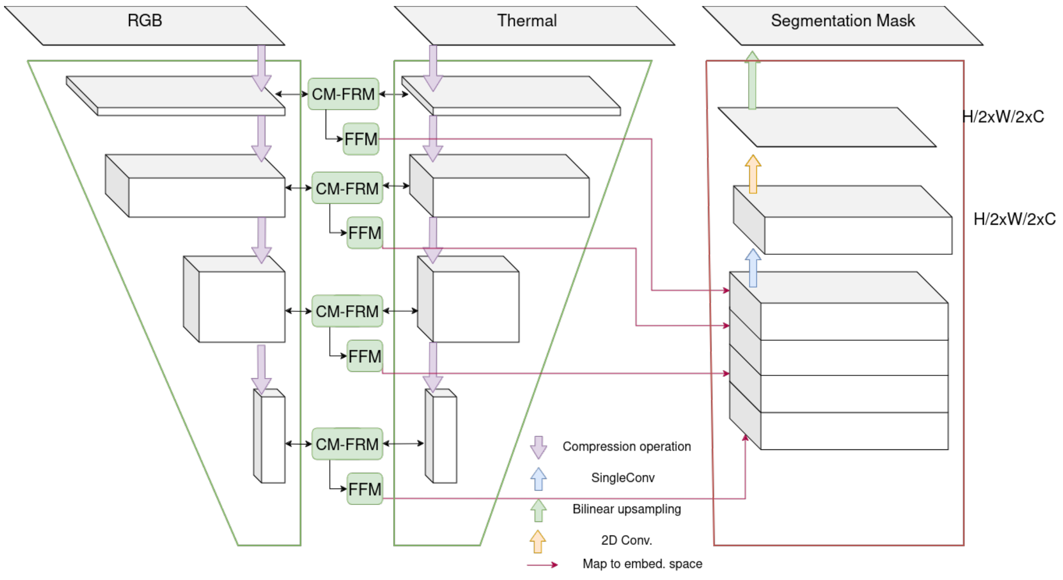

Figure 4.7: CMX variation with custom MLP encoder on our proposed WNet.

This attention usage is similar to the mechanism used in natural language processing (NLP), where the model learns to focus on the essential parts of the input [35]. The attention-based module is called Separation-and-Aggregation (SA) Gate, from which we induced the architecture name, as there is none explicitly given in the paper [16]. For further details on the architecture, please refer to the original paper. This module is very similar to CMFRM in the CMX architecture. The inspiration from SA-Gate is also mentioned in CMX. However, in contrast to CMX, SA-Gate modules are also used as input to the decoder, but there is also one connection, which is at the decoder's first level. In the original implementation, two losses are used. One is computed between the network output and the ground truth, and the other, auxiliary loss, is calculated between the upsampled low-resolution encoder output process through a single convolutional auxiliary layer and the ground truth to enforce correct information propagation through the decoder. A cross-entropy loss was used for both losses, which we replaced with a binary cross-entropy loss because of only binary output. We kept the original weights for the two losses, $[1, 0.2]$. There are then three versions of the SA-Gate architecture in the benchmark, SA-Gate18, SA-Gate18-1L, and SA-Gate34, where the number in the name represents the ResNet backbone used, and 1L stands for a single loss version. Lastly, the same double loss architecture with an auxiliary output as in SA-Gate was adopted in the WNet-S architecture named WNet-S2L, where the 2L in the suffix stands for two losses.

4.2 Training details

All the models were implemented in Pytorch and trained on NVIDIA GeForce RTX 4070 Ti. Optimizer Adam was used with a learning rate of $5 \cdot 10^{-4}$, which was changed to $1 \cdot 10^{-4}$ or $5 \cdot 10^{-5}$ depending on architecture used. All the models were trained with augmentation techniques, namely random horizontal flip and affine random transformation from the Albumentations library [15]. The affine transformation includes random scale, rotation, shear[1], and translation. The training was performed for 60 epochs with a multistep learning rate scheduler. The learning rate was reduced by a factor of 0.1 twice at epochs 40 and 50 of the training. The mean Intersection over Union (mIoU) metric was used to save the best model, computed on the evaluation dataset for every epoch. This metric is introduced in Section 5.1. The training and validation split was 75% and 25%, which was performed on the full training set, a description of which is provided in Chapter 3. However, the split was not performed on images randomly to prevent information leaks from training to validation set across the same scene. The split was done with whole trips assigned to training or validation in approximately the desired ratio. A separate test set was generated for the final evaluation of the models in Chapter 5. The used batch size was four due to the limited GPU memory of 16GB.

[1]Shear is indeed not an affine transformation, but it is still part of the affine transformation augmentation function in Albumentations.

Chapter 5

Results

This chapter aims to compare architectures presented in Chapter 4 and evaluate their performance on a test dataset composed again in the way introduced in Chapter 3. Another goal is to identify the architectures' essential features by introducing them with different tweaks and comparing their performance on the same dataset. The features can be specific fusion architectures, encoders, decoders, or convolution types. The tweak-based approach is similar to the approach adopted typically in the ablation studies of current papers dealing with CNN architectures. All the architectures and their modifications are trained on the same dataset (the dataset was introduced along with the creation procedure in Chapter 3), and the tweaks (named as flavors or versions) were presented in Chapter 4 along with the architectures. In this chapter, the used metrics are first introduced for segmentation along with results in Section 5.1, then the object detection metrics and results are introduced in Section 5.2.

5.1 Segmentation metrics and results

For each input image pair of RGB image $I_i \in [0,1]^{3\times H\times W}$ and thermal image $T_i \in [0,1]^{H\times W}$, each of the tested architectures M_j produces is a tensor $O_{ij} = M_j(I_i)$, where $O_{ij} \in [0,1]^{C\times H\times W}$, where C is the number of classes. The output O_{ij} is typically interpreted as a probability distribution over the classes, where the class with the highest probability is chosen as the predicted class. As this work aims to explore whether the multimodal fusion can improve the general object detection capabilities in the image, and since the multi-class classification could act as a confounder, we decided to have a single class of a 'general floating object'. In other words. In our case, $C = 1$. In this single-class case, the output is interpreted as a probability of the pixel belonging to the class rather than to the background.

In order to evaluate the segmentation or obtain production results, a confidence threshold needs to be set, and the pixels with a probability higher than the threshold are considered the predicted object. This way we obtain a mask, which is matrix $O^T_{ij} \in \{0,1\}^{H\times W}$, what is the same format as the ground truth masks $G_{ij} \in \{0,1\}^{H\times W}$ have. With these pairs in hand, we can compute the metrics for each image.

The first of the used metrics is mean Intersection over Union (often also referred to as Jaccard Index), which stands for an equation

$$\text{IoU} = \frac{|O^T_{ij} \wedge G_{ij}|}{|O^T_{ij} \vee G_{ij}|} = \frac{TP}{TP + FP + FN}, \tag{5.1}$$

where $\vee$ is element-wise logical or, and $\wedge$ is a logical and. The operation $|\cdot|$ then represents the sum of all the elements in the matrix. In the context of the evaluation of semantic segmentation, TP stands for a number of the true positive matrix elements, FN for false negative,

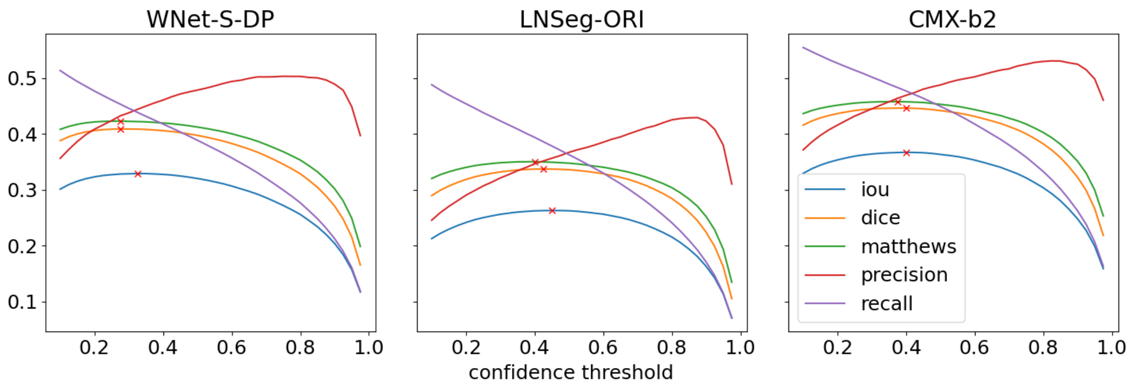

Figure 5.1: Several segmentation metrics computed for more confidence thresholds with maximums marked. All the metrics computer for three benchmarked models: WNet-S-DP, LNSeg-ORI and CMX-b2.

FP for false positive, and TN for true negative. Another used metric is the Dice coefficient (also called F1 score), defined as

$$DC = \frac{2|O_{ij}^T \wedge G_{ij}|}{|O_{ij}^T + G_{ij}|} = \frac{2TP}{2TP + FP + FN}, \tag{5.2}$$

where + is an element-wise addition in the middle part of the equality. Since the F1 score is the harmonic mean of precision and recall, we should also specify precision and recall

$$\text{Recall} = \frac{TP}{TP + FN}, \tag{5.3}$$

and

$$\text{Precision} = \frac{TP}{TP + FP}. \tag{5.4}$$

The last of the used metrics is the Matthews coefficient

$$MC = \frac{TP \cdot TN - FN \cdot FP}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \tag{5.5}$$

With all the metrics set, we can reopen the topic of confidence threshold. The threshold is a hyperparameter, and the choice of the threshold can significantly affect the results. As one would strive to choose the threshold optimally in the production environment based explicitly on the model's performance on the validation, we evaluate the model with multiple thresholds first and choose the best one for each of the architectures to preserve fairness. The metric curves for the thresholds with maximums of the metrics marked are presented in Figure 5.1 for three sample models. The results for other models are very similar.

To choose the threshold for each model, we selected the threshold corresponding to the maximum of the mean IoU metric, as it is the most conservative metric. With the threshold selected, each metric was computed separately for each image and then averaged over the whole test dataset. The results are presented in Table 5.1. The table also contains the average time (in milliseconds) needed for a single-sample model inference on the test dataset. All the models were running on the same unoccupied GPU in evaluation mode. The table also shows the model's size in millions of parameters and the selected threshold. The best results are highlighted in bold. The models are separated by a horizontal line into groups of the

Table 5.1: Segmentation benchmark results. The time is in milliseconds, the size is in millions of parameters, and the threshold is the confidence threshold.

model	time	size	thr	iou	dice	matthews	precision	recall
RTFNet18	4.13	31.00	0.28	31.74	39.76	41.18	40.43	45.73
GMNet18	6.70	31.98	0.25	33.17	41.11	42.39	42.38	45.94
GMNet18-1L	6.66	31.87	0.28	32.76	40.57	41.81	41.73	45.42
GMNet34	8.14	52.20	0.22	32.04	39.88	41.16	40.40	45.55
SAGate18	4.28	9.82	0.35	32.93	40.80	42.11	42.28	45.60
SAGate18-1L	4.38	9.67	0.30	32.99	40.95	42.33	42.35	46.27
SAGate34	5.71	14.99	0.32	34.46	42.46	43.74	43.72	47.39
WNet-S	3.52	**0.62**	0.28	33.13	41.12	42.46	43.38	45.32
WNet-NS	2.94	5.17	0.32	32.69	40.57	41.97	43.66	44.23
WNet-S2L	3.62	0.64	0.35	31.88	39.92	41.34	42.69	44.30
WNet-S-DP	3.67	2.25	0.28	32.82	40.89	42.29	43.22	45.32
WNet-MLP	**2.24**	0.88	0.32	32.96	41.07	42.41	44.52	44.15
WNet-CMX	5.06	4.95	0.25	32.41	40.51	41.97	42.61	45.53
WNet-CX	5.18	4.51	0.25	32.67	40.46	41.76	42.15	45.41
CMX-b0	6.98	12.11	0.30	33.99	41.84	43.05	42.03	47.48
CMX-b2	11.68	66.56	0.38	**36.67**	**44.61**	**45.76**	**46.37**	**48.29**
LNSeg-S4	**1.99**	0.35	0.28	31.99	39.89	41.29	42.20	44.46
UNet-S4	2.40	**0.29**	0.25	31.68	39.68	41.19	41.98	44.82
LNSeg-ORI	2.25	0.36	0.38	31.94	40.41	41.70	40.27	46.74
LNSeg-S	**2.02**	0.35	0.22	30.67	38.36	39.87	41.17	43.26
UNet-S	2.31	**0.29**	0.22	30.40	38.20	39.58	40.70	42.50
LNSeg-S3	**2.04**	0.35	0.15	16.52	23.00	24.57	22.45	31.48
UNet-S3	2.28	**0.29**	0.10	16.17	22.57	24.28	21.91	31.87

same architecture type. The architecture types are double-encoder, single-encoder with four-channel input, single-encoder thermal input and single-encoder RGB input. The model names correspond to the names introduced in Chapter 4, and the tweaks are marked with model name suffixes. If the ResNet backbone is used, there is a number 18 or 34 in the name, which corresponds to the version of the ResNet backbone. The 1L suffix stands for the single-loss training for architectures originally trained with multiple losses. For CMX, the suffixes b0 and b2 denote the backbone used, Segformer MiT-b0 or MiT-b2. If the suffix contains the letter S, it means that a separable convolution was used, and NS means the opposite. The suffix 2L denotes that the model was trained with two losses. The suffix DP is used in the trained models with more channels in each layer, giving the model more parameters and, thus, more computation capabilities. Then there are WNet and CMX crossovers WNet-MLP, WNet-CMX, WNet-CX, more details on which can be found in Section 4.1.5. The number 4 in LNSeg-S4 and UNet-S4 stands for the number of channels in the input. The number 3 in LNSeg-S3 and UNet-S3 stands for the number of channels in the input, meaning that only RGB input is used. Finally, LNSeg-ORI is the original LNSeg model without any tweaks and original weights trained on the original full datasets.

We can see that all the other architectures strongly dominate the solely RGB input models. The resolution of the RGB image is lower than the thermal image, and the RGB image is more affected by the environment, so it is a good additional modality, but it does not work well as a standalone modality. Therefore, it is excluded from the further discussion. We can see that inference time is not strictly linearly dependent on the number of parameters, as the number of parameters is not the only factor affecting the inference time but also the architecture structure. As this observation is challenging to make from the table, these results are visualized in Figure 5.2. Note that this might change when the models are deployed on the edge devices, as graph optimization is also performed when the models are compiled before the deployment. However, the inference time in the development environment can still be a reasonable indicator of the model's performance on the edge device, as most architectures are based on the same components and architecture concepts. As the main criterion in the production deployment on the near-real-time device is the inference time, another plot was constructed to show the dependence between inference time vs. the mean IoU metric in Figure 5.3.

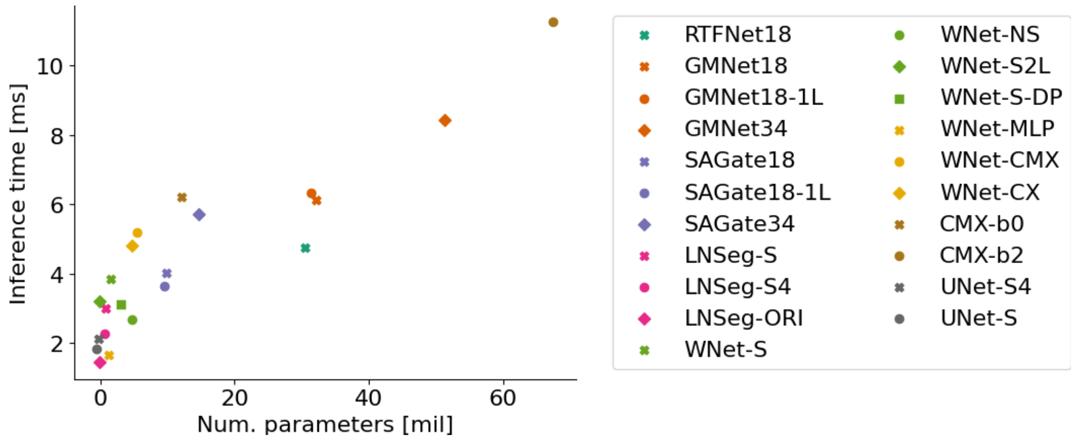

Figure 5.2: The number of parameters vs. the inference time of benchmarked models.

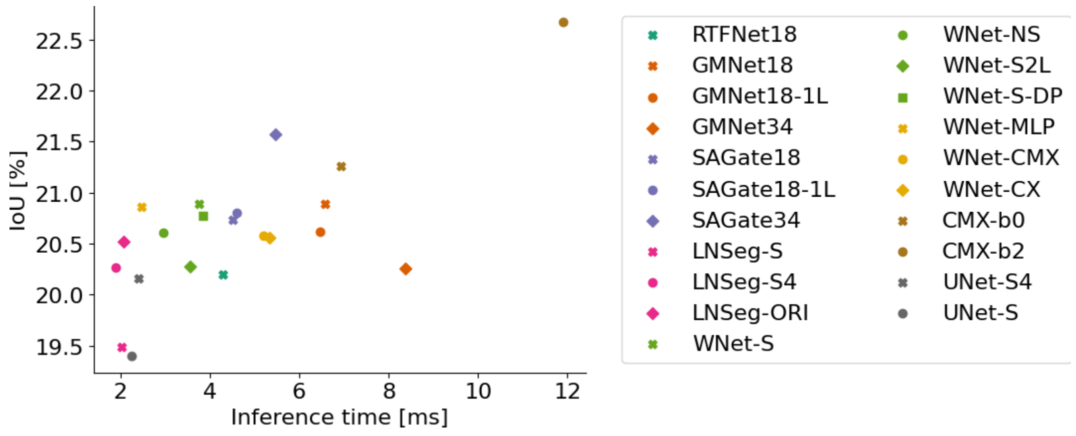

Figure 5.3: The inference time of benchmarked models vs. their seg. IoU score.

Finally, four example images from the test dataset with the generated segmentation masks and ground truth are presented in Figure 5.4 for the models CMX-b0, RTFNet18, and LNSeg-S.

From the results presented in this section, we can make several observations. The first and most important conclusion we can make is that introducing the second modality into

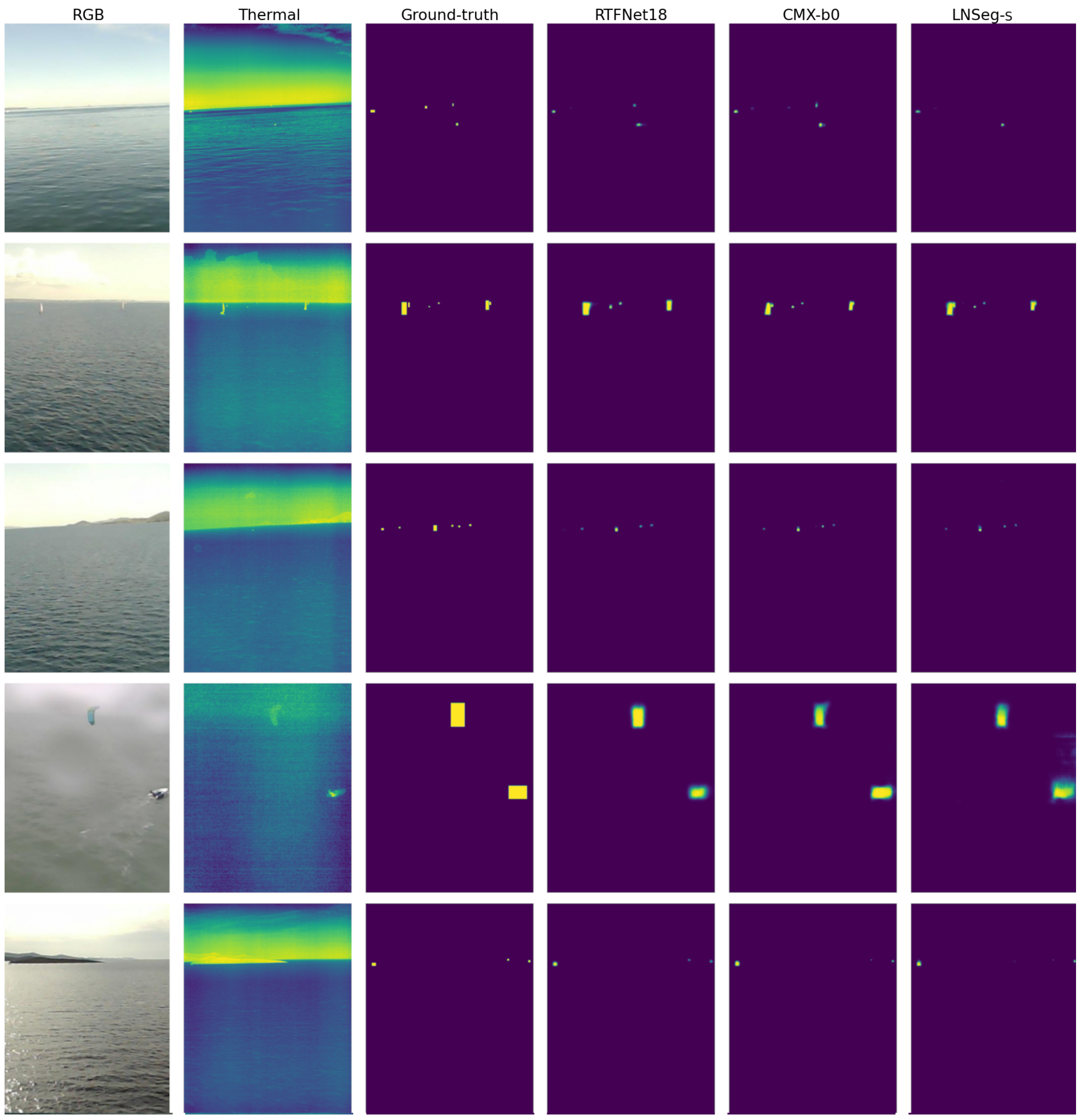

Figure 5.4: Samples from the aligned test set with segmentation masks visualized as collage. RGB image, thermal image, ground-truth mask, and outputs of RTFNet18, CMX-b0, and LNSeg-S from left to right for each row.

the detection process is beneficial, as the models with the second modality (RGB image in our case) in the input have superior performance in the segmentation task compared to single-modal architectures. Even the 4-channel-input variants of the models have a better performance than the 1-channel-input variants. Nevertheless, the segmentation scores are dominated by the double-encoder models. Secondly, we can see that the assumption that the bigger the encoder, the better the segmentation results hold for all models that were trained with multiple encoder sizes besides GMNet. Moreover, there is no clear stance that one can take on the general beneficiality of multiple loss training, which, on the other hand, leads only to an improvement in the case of GMNet. However, we can see that the benchmark is dominated by the biggest model, CMX-b2, a transformer architecture that also has the highest inference time. Its smaller version, CMX-b0, a significantly smaller model in terms of parameters and inference time, is also among the best models. If we look at the inference time vs. the IoU score ratio, among the best models can be counted CMX-b0, SAGate34, WNet-MLP, and WNet-S.

5.2 Object detection metrics

Our pipeline uses the segmentation results for bounding box extraction and object detection. Therefore, the object detection metrics are also computed for those. The process of bounding box extraction is done by finding the connected components of the mask, and then the bounding box is fitted around every component. The score or confidence of each bounding box is simply the mean of the mask's pixel-wise confidence. After these two steps, the output is the same as any other object detection model, i.e., for each input image pair, it is a list of bounding box coordinates with their confidence and class. Again, in our case, there is only one class. The metrics are then computed as typical for object detection algorithms described below.

The main metric used for object detection evaluation is the Average Precision (AP). It builds on three core concepts: precision, defined in Eq. 5.4, recall from Eq. 5.3, and the intersection over union (IoU). IoU was already introduced in the context of segmentation evaluation in Eq. 5.1; the gist of it is the same, but we were comparing two matrices, so it was evaluated pixel-wise. Here, we are comparing two rectangles, so the approach is area-wise, and it cannot be computed using TP, FP, and FN as there are no such values for areas. Therefore, let us start with IoU in the context of object detection. IoU is a measure of the overlap between two bounding boxes A, B, and it is defined as

$$\text{IoU} = \frac{|A \cap B|}{|A \cup B|}, \tag{5.6}$$

where $|A \cap B|$ is the area of the intersection of the bounding boxes, and $|A \cup B|$ is the area of their union. Then, naturally, the more these two bounding boxes overlap, the higher the IoU. In order to find correct detections, we typically define an IoU threshold th. Based on this degree of overlap, the detected bounding box is a true positive if the IoU is higher than the set threshold with some ground-truth bounding box and a false positive otherwise. With this classification, we can easily define TP as the number of true positive bounding box detections, FP as the number of false positive bounding box detections, etc. AP is computed as the area under the precision-recall curve, where the precision and recall are defined in Eq. 5.3 and Eq. 5.4. The precision-recall curve is built using detections ordered by their confidence. A variation of the original 11-point Pascal VOC interpolation algorithm [47] is used to estimate the average precision from the area under the precision-recall curve. Namely, its 101-point version is implemented in the pycocotools library [44]. This process results in the AP^{th} for an initially specified IoU threshold th. However, having a single threshold to assess our detection model's performance might not be robust enough, as a single threshold may induce a bias in the results. Also, it can be more lenient for some models than others. Therefore, the final AP value is computed as a mean over multiple IoU thresholds. In the case of pycocotools, the thresholds are at an interval of $[0.5, 0.95]$ with a step of 0.05. Moreover, the AP can be computed for objects of different sizes. We defined four size categories: small, medium, large, and valid. Valid encompasses all three categories and shares a lower bound of 1.75×1.75px with the small category. This lower bound is used to filter out the small objects that are not relevant

Table 5.2: Detection benchmark AP results for various detection size classes. All in percents.

model	valid		small		medium		large	
	$AP^{0.5}$	AP	$AP^{0.5}$	AP	$AP^{0.5}$	AP	$AP^{0.5}$	AP
RTFNet18	33.96	13.35	13.42	4.58	42.80	15.04	71.39	31.48
GMNet18	36.27	14.63	13.76	4.91	50.19	18.49	70.91	32.25
GMNet18-1L	37.41	15.33	13.82	4.68	53.05	19.20	71.48	33.32
GMNet34	34.90	13.98	12.04	4.12	46.65	16.72	70.54	31.78
SAGate18	37.66	14.48	18.05	6.42	50.21	17.75	68.63	29.31
SAGate18-1L	37.50	14.81	18.14	6.52	46.53	17.03	71.19	31.63
SAGate34	40.96	16.01	19.73	6.94	52.97	19.44	76.33	32.91
WNet-S	42.26	16.35	23.49	**8.55**	57.32	21.13	65.52	27.58
WNet-NS	39.22	15.23	23.54	8.49	51.29	18.59	61.29	26.14
WNet-S2L	37.48	14.18	21.58	7.79	50.65	18.60	58.73	24.16
WNet-S-DP	41.18	15.70	**23.87**	8.42	53.20	19.37	65.87	27.73
WNet-MLP	39.15	14.86	23.57	8.31	51.34	18.90	61.98	25.32
WNet-CMX	40.15	15.11	23.26	8.23	50.92	18.59	65.75	26.75
WNet-CX	41.56	15.83	22.41	7.97	56.02	20.45	66.43	27.53
CMX-b0	41.39	16.56	20.24	7.10	54.38	20.17	74.88	34.02
CMX-b2	**45.17**	**18.67**	21.87	7.72	**60.75**	**23.20**	**76.60**	**36.77**
LNSeg-S4	40.95	15.46	23.70	8.37	54.54	20.22	64.17	26.40
UNet-S4	39.23	14.57	23.63	8.38	54.81	19.54	59.38	23.69
LNSeg-ORI	34.61	14.00	8.24	2.87	51.14	17.78	75.75	35.52
LNSeg-S	37.72	14.30	21.92	7.91	51.96	18.41	55.40	23.38
UNet-S	36.53	13.41	22.56	7.82	49.17	17.60	53.73	20.94

for the detection and would be filtered out as noise even in production deployment. The other two thresholds between size categories are 8×8px and 20×20px. For more information on AP computation details, refer to [12].

Even though AP is a good metric for a robust comparison of overall model performance, there are also other indicators that we want to extract from the model outputs. Sometimes, more specific analysis is needed, such as the model's precision or false positive rate. False positive and false negative rates and the precision and recall stemming from them are typically essential metrics in the context of autonomous vehicles, where an overlooked object or false alarm can cause severe issues. In the specific context of SEA.AI systems, an alarm system is connected to the detections, so the precision, recall, and their harmonic mean, the F1 score, are good indicators of potential reliability for the user. Therefore, we also compute the TP, FP, FN, precision, recall, and F1 scores for each size range. The IoU threshold used in the tables related to these indicators is 0.5. The results are presented in Table 5.3 for the F1 score, precision, and recall, and the rest are presented in Table 5.4.

Besides the table representation of AP and detection scores, we also visualized the results vs the inference time as in the segmentation case. These results are presented in Figure 5.5 for AP values and in Figure 5.6 for the detection F1 score. The FP number vs. inference time is then shown in Figure 5.7. All of the plots were computed from the 'valid' size class and with an IoU threshold of 0.5.

Finally, a visualization of the extracted bounding boxes from the segmentation masks

Table 5.3: Detection benchmark precision, recall and F1 score results for various detection size classes. All in percents.

model	valid			small			large		
	prec	rec	f1	prec	rec	f1	prec	rec	f1
RTFNet18	45.41	42.88	44.11	28.62	23.54	25.83	78.36	76.06	77.19
GMNet18	46.47	44.77	45.60	28.77	24.56	26.50	76.07	75.57	75.82
GMNet18-1L	49.80	45.19	47.39	31.64	24.02	27.31	76.82	75.57	76.19
GMNet34	46.87	43.63	45.19	28.86	23.18	25.71	78.31	74.10	76.15
SAGate18	50.56	47.44	48.95	35.06	29.32	31.93	69.67	76.71	73.02
SAGate18-1L	49.18	47.70	48.43	34.51	30.52	32.40	74.38	78.01	76.15
SAGate34	50.82	50.70	50.76	34.89	31.91	33.33	72.96	83.06	77.68
WNet-S	50.66	51.61	51.13	37.27	36.48	36.87	70.02	71.50	70.75
WNet-NS	49.82	49.95	49.89	38.39	36.85	37.60	62.95	68.89	65.79
WNet-S2L	43.90	48.52	46.09	30.71	34.80	32.63	64.21	66.61	65.39
WNet-S-DP	48.28	51.58	49.87	35.32	**38.11**	36.66	69.42	72.48	70.92
WNet-MLP	44.39	50.64	47.31	32.96	37.09	34.90	61.24	72.31	66.32
WNet-CMX	48.86	50.11	49.48	35.98	36.30	36.14	67.37	73.29	70.20
WNet-CX	51.65	51.12	51.38	37.63	35.82	36.71	69.27	73.78	71.45
CMX-b0	53.47	50.67	52.03	37.70	32.39	34.84	76.12	80.46	78.23
CMX-b2	**56.06**	**54.22**	**55.13**	**40.01**	35.34	37.53	**81.48**	**82.41**	**81.94**
LNSeg-S4	48.38	50.64	49.48	35.11	36.54	35.81	68.95	70.52	69.73
UNet-S4	48.19	49.79	48.97	36.42	35.76	36.09	61.73	68.57	64.97
LNSeg-ORI	33.76	41.90	37.39	13.18	16.92	14.82	78.22	80.13	79.16
LNSeg-S	47.84	48.26	48.05	35.99	35.34	35.66	66.49	61.40	63.84
UNet-S	45.91	48.06	46.96	34.35	36.54	35.41	61.97	62.38	62.18

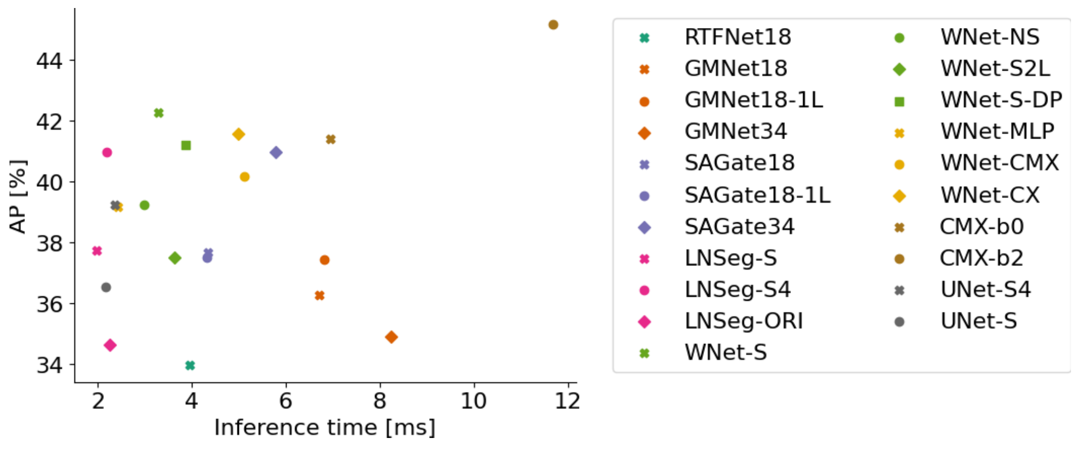

Figure 5.5: The inference time of benchmarked models vs. their Average Precision value.

Table 5.4: Detection benchmark number of true positive, false positive and negative detections for various detection size classes.

model			valid			small			medium			large
	tp	fp	fn	tp	fp	fn	tp	fp	fn	tp	fp	fn
RTFNet18	1316	1582	1753	391	975	1270	460	501	337	467	129	147
GMNet18	1374	1583	1695	408	1010	1253	504	452	293	464	146	150
GMNet18-1L	1387	1398	1682	399	**862**	1262	525	420	272	464	140	150
GMNet34	1339	1518	1730	385	949	1276	501	469	296	455	126	159
SAGate18	1456	1424	1613	487	902	1174	500	327	297	471	205	143
SAGate18-1L	1464	1513	1605	507	962	1154	479	399	318	479	165	135
SAGate34	1556	1506	1513	530	989	1131	517	338	280	510	189	104
WNet-S	1584	1543	1485	606	1020	1055	540	346	257	439	188	175
WNet-NS	1533	1544	1536	612	982	1049	500	322	297	423	249	191
WNet-S2L	1489	1903	1580	578	1304	1083	503	379	294	409	228	205
WNet-S-DP	1583	1696	1486	**633**	1159	**1028**	507	347	290	445	196	169
WNet-MLP	1554	1947	1515	616	1253	1045	496	430	301	444	281	170
WNet-CMX	1538	1610	1531	603	1073	1058	487	326	310	450	218	164
WNet-CX	1569	1469	1500	595	986	1066	523	**287**	274	453	201	161
CMX-b0	1555	1353	1514	538	889	1123	525	321	272	494	155	120
CMX-b2	**1664**	**1304**	**1405**	587	880	1074	**574**	324	**223**	**506**	**115**	**108**
LNSeg-S4	1554	1658	1515	607	1122	1054	516	349	281	433	195	181
UNet-S4	1528	1643	1541	594	1037	1067	515	355	282	421	261	193
LNSeg-ORI	1286	2523	1783	281	1851	1380	515	548	282	492	137	122
LNSeg-S	1481	1615	1588	587	1044	1074	520	387	277	377	190	237
UNet-S	1475	1738	1594	607	1160	1054	488	350	309	383	235	231

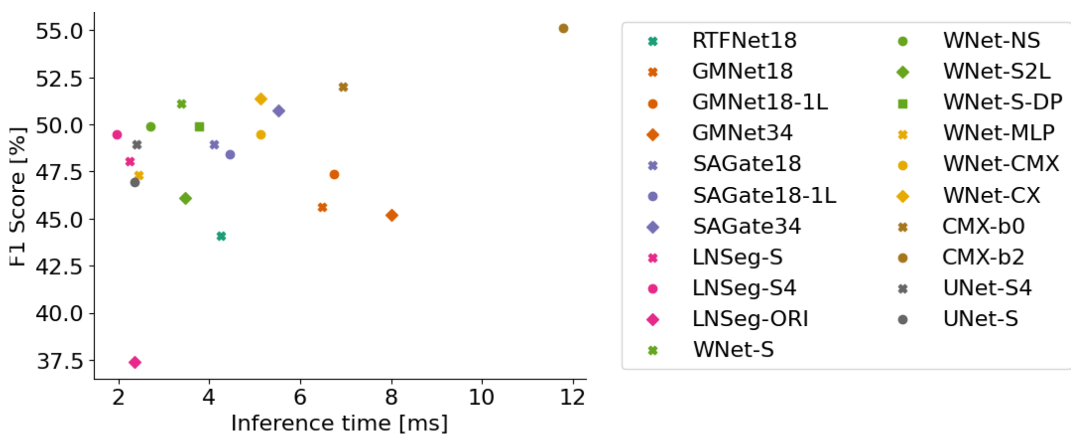

Figure 5.6: The inference time of benchmarked models vs. their detection F1 Score.

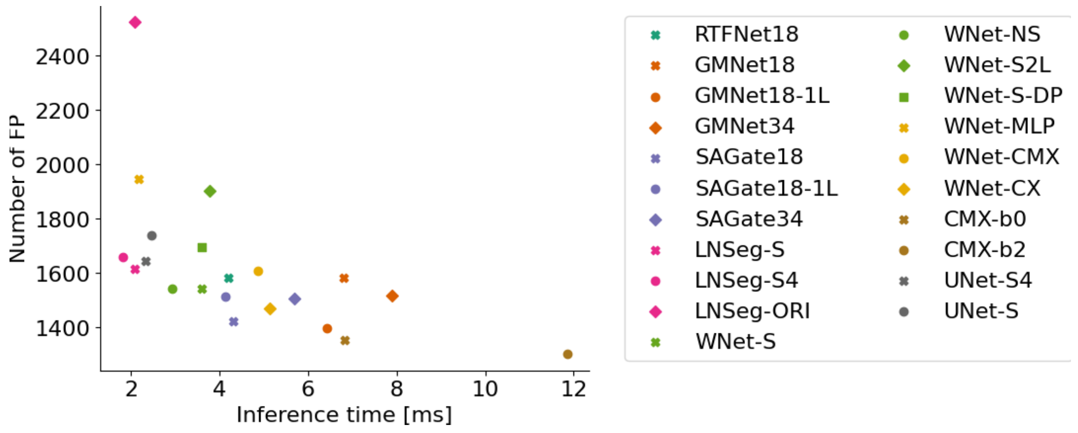

Figure 5.7: The inference time of benchmarked models vs. their number of false positives.

is presented on four example images in Figure 5.8 for the models CMX-b0, RTFNet18, and LNSeg-S.

From the results in this section, we can see the main result from the previous section reinforced, i.e., the use of the other modality is beneficial in all metrics used for object detection. Moreover, the metrics are again mostly dominated by the bigger transformer architecture, with its smaller version closely following. However, the previous conclusion about the size of the encoder now holds only for CMX in all the metrics from this section, thus making it not a strong conclusion. However, the CMX fusion strategy has proven effective since its implementation into WNet with a normal WNet head often outperforms the original WNet, and most of the architectures are in the same inference-time range. We can also see from the tables that most of the false positives and false negatives come from the small object category. Moreover, we have demonstrated that the distillation procedure retains the model's performance while reducing the training time. This improvement can be seen between the models LNSeg-ORI and LNSeg-S, where the first was trained on the entire dataset, and the second was trained on the distilled dataset, while the performance is better for the newly trained one. This may be accounted for by the usage of augmentation. Notable is also the decreasing trend of the number of false positives with the inference time. Another conclusion we can reach is that the number of parameters of the architecture and its complexity are not always the most reliable performance indicators. Bigger and more complex architectures such as RTFNet or GMNet are excellent examples of this, as WNet-S matched their performance in most metrics, while remaining a way smaller and simpler model. Lastly, we can outline models that deliver a good performance vs. inference time ratio: WNet-S, WNet-CX, CMX-b0, and SAGate34.

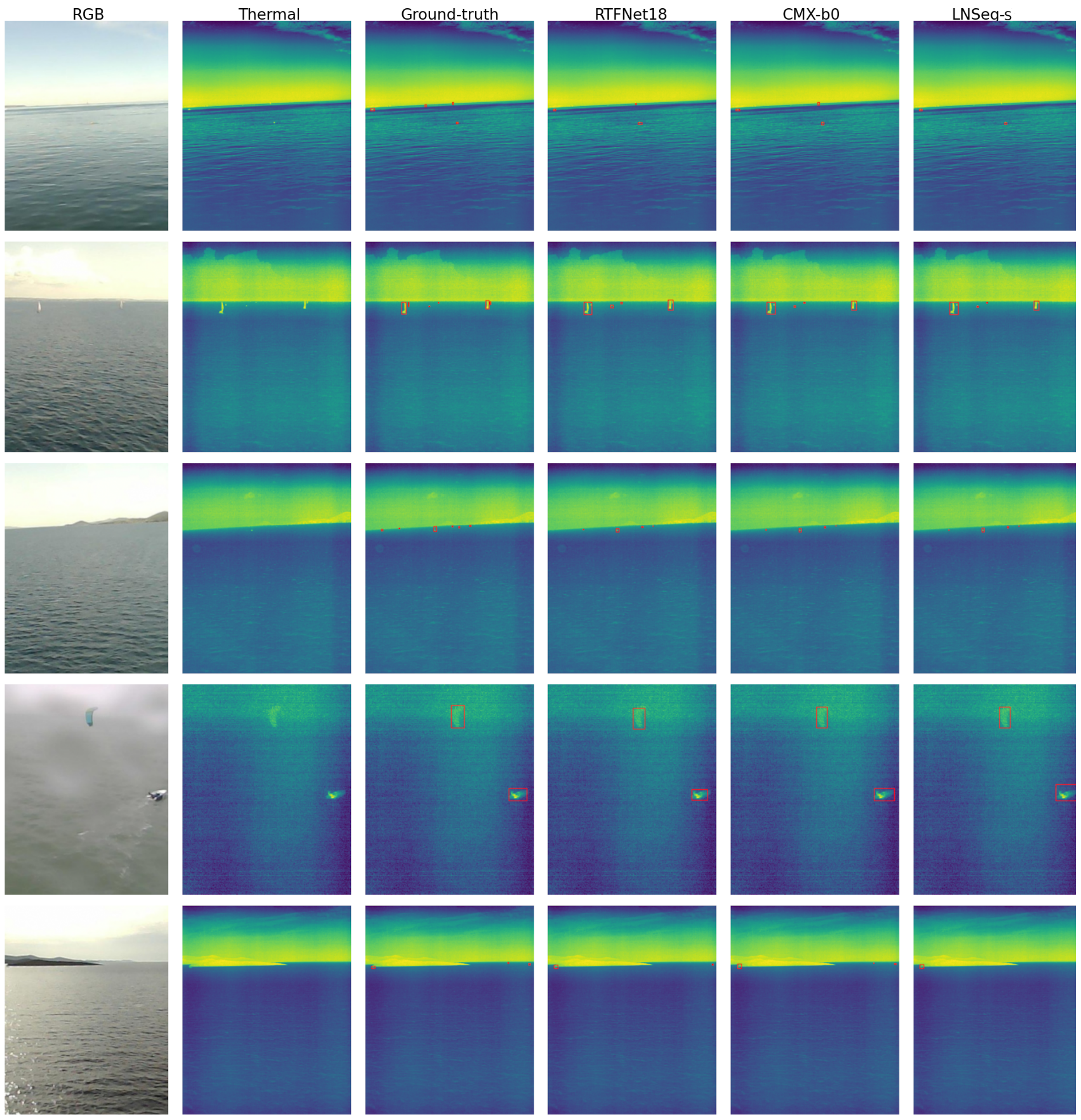

Figure 5.8: Samples from aligned test set with extracted bounding boxes visualized as collage. RGB image, thermal image, ground-truth mask, and outputs of RTFNet18, CMX-b0, and LNSeg-S from left to right for each row.

Chapter 6

Conclusion and oulook

This work explored the origin and possible solutions to thermal and RGB multimodal object detection challenges on datasets collected in marine environments. Firstly, the dataset, sensors, and collection methodology were introduced. However, the explored machine learning task requires robustness and precision, which demands high-quality training data. Thus, we developed processing methods related to alignment and data distillation to obtain such a dataset. The first challenge with the dataset was the misalignment between the modalities. The first explored approach to misalignment is an FFT-based translation and rotation estimation. These methods were not robust enough for the marine environment and often malfunctioned due to visual differences between thermal and RGB images. Therefore, another approach was introduced based on correspondence search using deep learning followed by homography estimation. This approach proved more robust, based even on a single image pair. Moreover, the correspondences can be collected through more images in the same sequence. Thus, images with no distinct features can also be aligned if at least some correspondences are found in the sequence. Therefore, the latter approach was used for image alignment and propagating ground-truth annotations from both modalities to create a single training target with sufficient precision. The size of the aligned dataset was reduced while striving to maintain most of its information using embeddings and clustering methods to less than one-twentieth of the original size.

Several state-of-the-art models were then trained on the gathered dataset, our proposed models, and a model formerly used in SEA.AI devices. All the models were then evaluated on a separate test dataset to identify architectures and features suitable for multimodal segmentation in marine environments. Several metrics were used for the benchmarking. These included segmentation and object detection metrics, which were computed directly on thresholded segmentation outputs in the first case and after bounding box extraction from the outputs in the latter. From the results, we made several observations. Firstly, leveraging a second modality has shown to be an improvement in all of the metrics in the benchmark. Most of the benchmarks were dominated by transformer-based architecture CMX, which is, on the other hand, the one with by far the highest inference times and one of the most complex architectures, However, besides this architecture, we cannot conclude that the bigger the inference time or the more complex the architecture, the better the performance. There are opposite examples in the benchmarks, where simple architectures such as WNet-S reached very similar results to the more complicated and slower GMNet or RTFNet. However, isolating a single architecture feature that would be the best in all the metrics while maintaining a reasonable complexity to run in a near-real-time regime is impossible besides the double encoder element. However, we can point out architectures with a consistently high ratio of performance to inference time, such as WNet-S, SAGate34, WNet-CX and CMX-b0.

Future work building on these results may include exploring the inference times after compilation, as the graph optimization can alter the inference time. Also, as SA-Gate and

WNet-S consistently performed well, extracting the SA-Gate fusion strategy and training WNet architecture makes sense as one of the future experiments. With the segmentation and bounding box extraction results showing significant improvement after introducing the second modality, using the multimodal fusion strategies in traditional object detection architectures such as YOLO or MaskRCNN is also a promising extension to the benchmark. Another starting point for future research is that the dataset distillation method has proven efficient enough for our purpose. It has even shown some improvement in the training results, so the logical development is to explore the effects of the model training in a more controlled environment to draw more definitive conclusions. Future research may also explore the environmental effects on the results of all the models, such as day vs. night time or snowy vs. rainy environment. Lastly, as multimodal fusion, in this case, has shown improvements in detection performance, fusion with more modalities, such as LiDAR data, may bring even more improvement.

References

[1] Y. Lv, Z. Liu, and G. Li, "Context-aware interaction network for rgb-t semantic segmentation," *IEEE Transactions on Multimedia*, 2024.

[2] M. Brenner, N. H. Reyes, T. Susnjak, and A. L. Barczak, "Rgb-d and thermal sensor fusion: A systematic literature review," *IEEE Access*, 2023.

[3] F.-X. R. Chen, C.-Y. Lin, H.-Y. Siao, C.-Y. Jian, Y.-C. Yang, and C.-L. Lin, "Deep learning based atomic defect detection framework for two-dimensional materials," *Scientific data*, vol. 10, no. 1, p. 91, 2023.

[4] H. Li, S. Han, J. Liu, H. Wang, Y. Wang, and S. Xie, "Sgnet: A fast and accurate semantic segmentation network based on semantic guidance," *Advanced Robotics*, vol. 37, no. 20, pp. 1301–1317, 2023.

[5] D. A. Maharani, C. Machbub, L. Yulianti, and P. H. Rusmin, "Deep features fusion for kcf-based moving object tracking," *Journal of Big Data*, vol. 10, no. 1, p. 136, 2023.

[6] B. Yin, X. Zhang, Z. Li, L. Liu, M.-M. Cheng, and Q. Hou, "Dformer: Rethinking rgbd representation learning for semantic segmentation," *arXiv preprint arXiv:2309.09668*, 2023.

[7] J. Zhang, H. Liu, K. Yang, X. Hu, R. Liu, and R. Stiefelhagen, "Cmx: Cross-modal fusion for rgb-x semantic segmentation with transformers," *IEEE Transactions on Intelligent Transportation Systems*, 2023.

[8] W. Zhou, H. Zhang, W. Yan, and W. Lin, "Mmsmcnet: Modal memory sharing and morphological complementary networks for rgb-t urban scene semantic segmentation," *IEEE Transactions on Circuits and Systems for Video Technology*, 2023.

[9] Z. Chen, Y. Duan, W. Wang, *et al.*, "Vision transformer adapter for dense predictions," *arXiv preprint arXiv:2205.08534*, 2022.

[10] H.-X. Cheng, X.-F. Han, and G.-Q. Xiao, "Cenet: Toward concise and efficient lidar semantic segmentation for autonomous driving," in *2022 IEEE International Conference on Multimedia and Expo (ICME)*, IEEE, 2022, pp. 01–06.

[11] D. Seichter, S. Fischedick, M. Köhler, and H.-M. Gross, "Efficient multi-task rgb-d scene analysis for indoor environments," in *IEEE International Joint Conference on Neural Networks (IJCNN)*, 2022, pp. 1–10. DOI: 10.1109/IJCNN55064.2022.9892852.

[12] R. Padilla, W. L. Passos, T. L. B. Dias, S. L. Netto, and E. A. B. da Silva, "A comparative analysis of object detection metrics with a companion open-source toolkit," *Electronics*, vol. 10, no. 3, 2021, ISSN: 2079-9292. DOI: 10.3390/electronics10030279. [Online]. Available: https://www.mdpi.com/2079-9292/10/3/279.

[13] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J. M. Alvarez, and P. Luo, "Segformer: Simple and efficient design for semantic segmentation with transformers," *Advances in neural information processing systems*, vol. 34, pp. 12 077–12 090, 2021.

[14] W. Zhou, J. Liu, J. Lei, L. Yu, and J.-N. Hwang, "Gmnet: Graded-feature multilabel-learning network for rgb-thermal urban scene semantic segmentation," *IEEE Transactions on Image Processing*, vol. 30, pp. 7790–7802, 2021.

[15] A. Buslaev, V. I. Iglovikov, E. Khvedchenya, A. Parinov, M. Druzhinin, and A. A. Kalinin, "Albumentations: Fast and flexible image augmentations," *Information*, vol. 11, no. 2, p. 125, 2020.

[16] X. Chen, K.-Y. Lin, J. Wang, *et al.*, "Bi-directional cross-modality feature propagation with separation-and-aggregation gate for rgb-d semantic segmentation," in *European conference on computer vision*, Springer, 2020, pp. 561–577.

[17] A. Dosovitskiy, L. Beyer, A. Kolesnikov, *et al.*, “An image is worth 16x16 words: Transformers for image recognition at scale,” *arXiv preprint arXiv:2010.11929*, 2020.

[18] D.-P. Fan, Y. Zhai, A. Borji, J. Yang, and L. Shao, “Bbs-net: Rgb-d salient object detection with a bifurcated backbone strategy network,” in *European conference on computer vision*, Springer, 2020, pp. 275–292.

[19] D. Moser, “The processing pipeline of a thermal imaging based maritime surveillance system,” Ph.D. dissertation, Technische Universität Wien, 2020.

[20] P.-E. Sarlin, D. DeTone, T. Malisiewicz, and A. Rabinovich, “Superglue: Learning feature matching with graph neural networks,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 4938–4947.

[21] S. S. Shivakumar, N. Rodrigues, A. Zhou, I. D. Miller, V. Kumar, and C. J. Taylor, “Pst900: Rgb-thermal calibration, dataset and segmentation network,” in *2020 IEEE international conference on robotics and automation (ICRA)*, IEEE, 2020, pp. 9441–9447.

[22] Y. Sun, W. Zuo, P. Yun, H. Wang, and M. Liu, “Fuseseg: Semantic segmentation of urban scenes based on rgb and thermal data fusion,” *IEEE Transactions on Automation Science and Engineering*, vol. 18, no. 3, pp. 1000–1011, 2020.

[23] X. Hu, K. Yang, L. Fei, and K. Wang, “Acnet: Attention based network to exploit complementary features for rgbd semantic segmentation,” in *2019 IEEE International conference on image processing (ICIP)*, IEEE, 2019, pp. 1440–1444.

[24] Y. Sun, W. Zuo, and M. Liu, “Rtfnet: Rgb-thermal fusion network for semantic segmentation of urban scenes,” *IEEE Robotics and Automation Letters*, vol. 4, no. 3, pp. 2576–2583, 2019.

[25] M. Berman, A. R. Triki, and M. B. Blaschko, “The lovász-softmax loss: A tractable surrogate for the optimization of the intersection-over-union measure in neural networks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 4413–4421.

[26] D. DeTone, T. Malisiewicz, and A. Rabinovich, “Superpoint: Self-supervised interest point detection and description,” in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2018, pp. 224–236.

[27] J. Redmon and A. Farhadi, “Yolov3: An incremental improvement,” *arXiv preprint arXiv:1804.02767*, 2018.

[28] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, “Rethinking atrous convolution for semantic image segmentation,” *arXiv preprint arXiv:1706.05587*, 2017.

[29] Q. Ha, K. Watanabe, T. Karasawa, Y. Ushiku, and T. Harada, “Mfnet: Towards real-time semantic segmentation for autonomous vehicles with multi-spectral scenes,” in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2017, pp. 5108–5115.

[30] C. Hazirbas, L. Ma, C. Domokos, and D. Cremers, “Fusenet: Incorporating depth into semantic segmentation via fusion-based cnn architecture,” in *Computer Vision–ACCV 2016: 13th Asian Conference on Computer Vision, Taipei, Taiwan, November 20-24, 2016, Revised Selected Papers, Part I 13*, Springer, 2017, pp. 213–228.

[31] K. He, G. Gkioxari, P. Dollár, and R. Girshick, “Mask r-cnn,” in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2961–2969.

[32] A. G. Howard, M. Zhu, B. Chen, *et al.*, “Mobilenets: Efficient convolutional neural networks for mobile vision applications,” *arXiv preprint arXiv:1704.04861*, 2017.

[33] G. Lin, A. Milan, C. Shen, and I. Reid, “Refinenet: Multi-path refinement networks for high-resolution semantic segmentation,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1925–1934.

[34] S.-J. Park, K.-S. Hong, and S. Lee, “Rdfnet: Rgb-d multi-level residual feature fusion for indoor semantic segmentation,” in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 4980–4989.

[35] A. Vaswani, N. Shazeer, N. Parmar, *et al.*, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[36] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning.* MIT press, 2016.

[37] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[38] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[39] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1440–1448.

[40] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International conference on machine learning*, pmlr, 2015, pp. 448–456.

[41] Y. Li, S. Wang, Q. Tian, and X. Ding, "A survey of recent advances in visual feature detection," *Neurocomputing*, vol. 149, pp. 736–751, 2015.

[42] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*, Springer, 2015, pp. 234–241.

[43] J. Yu and M. Blaschko, "Learning submodular losses with the lovász hinge," in *International Conference on Machine Learning*, PMLR, 2015, pp. 1623–1631.

[44] T.-Y. Lin, M. Maire, S. Belongie, *et al.*, "Microsoft coco: Common objects in context," in *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, Springer, 2014, pp. 740–755.

[45] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[46] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: An efficient alternative to sift or surf," in *2011 International conference on computer vision*, Ieee, 2011, pp. 2564–2571.

[47] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *International Journal of Computer Vision*, vol. 88, no. 2, pp. 303–338, Jun. 2010.

[48] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*, Ieee, 2009, pp. 248–255.

[49] M. Muja and D. Lowe, "Flann-fast library for approximate nearest neighbors user manual," *Computer Science Department, University of British Columbia, Vancouver, BC, Canada*, vol. 5, p. 6, 2009.

[50] R. C. Gonzalez and R. E. Woods, *Digital Image Processing.* Pearson, 2007.

[51] C. M. Bishop, *Pattern Recognition and Machine Learning.* Springer, 2006, ISBN: 978-0387310732.

[52] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision.* Cambridge University Press, 2004, ISBN: 978-0521540513.

[53] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, pp. 91–110, 2004.

[54] B. S. Reddy and B. N. Chatterji, "An fft-based technique for translation, rotation, and scale-invariant image registration," *IEEE transactions on image processing*, vol. 5, no. 8, pp. 1266–1271, 1996.

[55] K. Zuiderveld, "Contrast limited adaptive histogram equalization," in *Graphics gems IV*, Academic Press Professional, Inc., 1994, pp. 474–485.

[56] L. G. Brown, "A survey of image registration techniques," vol. 24, no. 4, 325–376, 1992, ISSN: 0360-0300. DOI: 10.1145/146370.146374. [Online]. Available: https://doi.org/10.1145/146370.146374.

[57] H. E. Wallis, "Adaptive smoothing for image analysis," *Pattern Recognition Letters*, vol. 2, no. 5, pp. 263–272, 1984.