SmartShip - AI-based assistance systems for the maritime sector

Pavle Ivanovic[1], Jonas J. Schöttler[2], Alexander Windmann[2], Philipp Neumann[1]
[1]*Chair for High Performance Computing*
[2]*Chair for Computer Science in Mechanical Engineering*
Helmut-Schmidt-Universität / University of the Federal Armed Forces
Hamburg, Germany
SmartShip@hsu-hh.de

***Abstract*—Modern ships, equipped with various sensors, devices of communications, etc., could produce vast amounts of data that should be efficiently collected, processed, and analyzed in order to fully unlock their operational potentials for prospective maritime activities. The project SmartShip explores how artificial intelligence (AI) can assist in search and rescue missions by expanding digital capabilities of ships and improving the decision making efficiency of operators. In this paper, we investigate the state-of -the-art of smart ships on technology development, explore the current state of the project, explain the types and formats of collected data and provide the first sensor analysis. Furthermore, a draft for the software and hardware architecture is detailed, emphasizing system scalability and platform independence. Finally, the paper explains how to use and process the ship's data to build an AI-enhanced system for condition monitoring and person detection that should assist in locating people in offshore distress.**

***Index Terms*—Artificial Intelligence, Condition Monitoring, Cyber-physical Systems, Object Detection**

I. Introduction

Maritime shipping is becoming increasingly complex. Not only has the sheer volume of shipping increased by around 300 percent over the last thirty years[1]and is forecast to grow by around 250 percent over the next few decades[2], but the complexity of individual ships has also increased. Whereas ships were operated manually in the past, today a multitude of sensors control their success on board. However, managing vast numbers of multi-vendor sensor devices, interfaces, protocols, and connectivity options requires the reorganization of the traditional data acquisition approach by putting more emphasis on data standardization. Also, new hardware and software solutions need to conform to several criteria, such as limited onboard space and power, reliability, efficient onboard/fleet scaling, and effective data processing. On the other hand, increased system complexity leads to situations where the evaluation and overview of these sensors can no longer be managed without digital aids. This is where the SmartShip project comes in and supports the crew and the inland coordination centers with systems such as condition monitoring, object detection, and network connections.

[1]wiwo.de: Drastische Zunahme seit 1992 setzt Ozeane unter Druck
[2]eskp.de: Rasante Entwicklung im Flug- und Schiffsverkehr

Condition monitoring can be operated in different ways. The simplest approach would be the empirical vibration evaluation of moving parts, e.g. the motor. However, the knowledge gained from this analysis is limited and no longer does justice to a system with complexity of a modern ship. For this reason, we present in this paper a machine learning (ML) based system that evaluates operational data from hundreds of sensors and derives condition monitoring. These systems can be used, for example, to optimize maintenance intervals and reduce costs by only replacing parts when the predicted service due time has already been approached. In this way, maintenance is more efficient and can also save valuable resources. At the same time, components that are already worn out and thus pose a risk to the system can be replaced before the planned maintenance. Correspondingly, the operational readiness of a ship is always guaranteed.

Systems for tracking objects in the water can help to better and more quickly locate people in distress during a man overboard maneuver or search and rescue (SAR) operations. Particularly useful is the combination with infrared (IR) data, which allows the crew to locate people even in bad weather conditions, such as fog, heavy rain, high waves, or strong winds. Previous solutions for people detection usually comprised personal engagement where crew members stood on the cams or bridges and used binoculars to scan the area. If camera systems were engaged, they would have to be operated manually and there was no digital evaluation of the video feed. We therefore present a solution that can automatically locate and track people and rescue assets in the water by leveraging the sensor fusion of electro-optical (EO) and IR data. As a result, the search supported by ML introduces an additional safety factor and a fallback for the mandatory manual search by the crew.

The rest of the paper is structured as follows. In Section II, the first results of the data analysis are presented and the future direction of condition monitoring in SmartShip is described. Section III explains how to use object detection algorithms to assist in search and rescue missions. In Section IV, a draft for the software and hardware architecture that generates the ship's data is presented. Finally, Section V

https://doi.org/10.24405/14567

concludes the paper and gives an outlook on the future direction of the project.

II. Condition Monitoring

In order to monitor the system's behavior, the machinery of a ship is equipped with sensors that track features like oil pressure or cooling water temperature. The current data are shown to the ship's crew that knows the expected values and can act if there is a deviation. However, the sensor values are changing dynamically and are highly correlated. As the number of sensors of the individual installed subsystems increases, it becomes harder to keep up. ML offers an opportunity to work through that data and assist the ship's crew with monitoring the condition of the ship [5]. The proposed approach can be separated into three phases: First, the state of the engine has to be identified in order to group the highly dynamic data. Second, for every state, anomalies that might signal a malfunctioning component have to be detected. Third, by working through the historic anomalies, patterns in the data can be detected and projected into the future, providing a guideline for maintenance routines.

A. Engine State Identification

In order to qualify particular system behavior as an anomaly, one needs to determine the so-called "normal" or expected performance metrics. However, dynamic change of system parameters typical for maritime rescue missions requires precise classification of the current SAR deployment phase. For instance, the high values of pressure and temperature gauges do not immediately indicate abnormal ship behavior, if the cruiser is in a top-speed mission mode. On the other hand, the same parameters would probably imply serious malfunction if the ship is stationed in a harbor.

There are several methods to identify the state of the engine. First of all, the states can be labeled due to some underlying feature. One of the parameters that could give a precise picture of the current engine state is the number of revolutions per minute (RPM). As depicted in Figure 1, four major phases of a typical cruising mission can be distinguished: Shut off (RPM $= 0$), neutral (RPM ≈ 600), acceleration/deceleration ($600 <$ RPM < 2300), and cruising speed (RPM ≈ 2300). However, if we observe the neutral engine state, there is a deviation from the expected (catalog) value of 600 RPM. This deviation is less than a few percent, if external factors, such as waves, are not exhibited. Nevertheless, the maximum neutral value is not reaching over 700 RPM, resulting in a fairly good upper threshold for setting the boundary between engine states.

Another way to derive the states is to cluster the data, for example by using the k-means algorithm, which can be seen in Figure 2. By using a statistical method or an ML algorithm, it is possible to identify the system state without requiring a domain expert to pick a characteristic feature and derive thresholds by hand. Furthermore, a more nuanced separation of the data is possible, as clusters can be formed

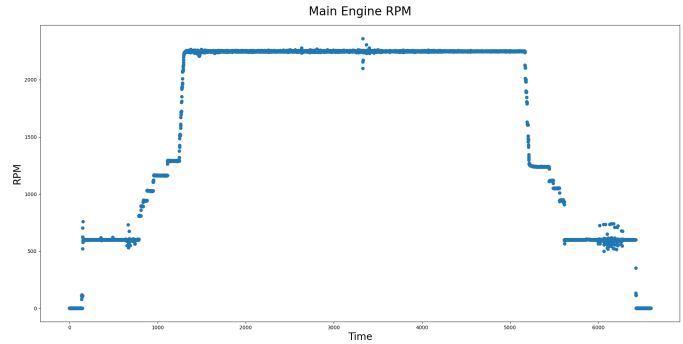

Figure 1. Main Engine RPM Measurements during Cruise Mission

over all features. However, most clustering algorithms struggle to separate time series data properly, which can also be seen in Figure 2. There are two distinct paths in the data: The lower path describes the data points during acceleration, and the upper path the data during deceleration, when the engine is heated. However, during high RPMs, these data are grouped together, as the algorithm has no information about the temporal context. Recently, there have been ML algorithms that learn the states together with their transitions [3], [7]. By clustering the time series data, the next steps can be performed much more efficiently.

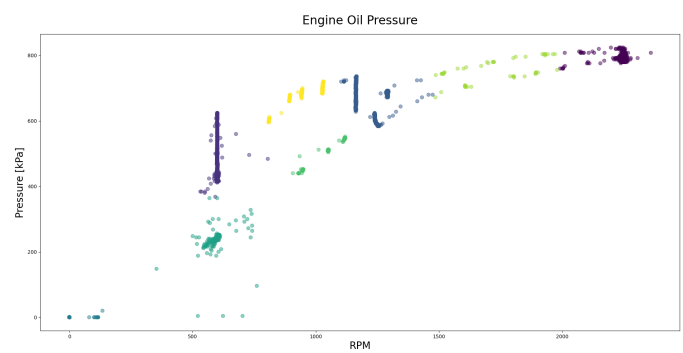

Figure 2. Main Engine Oil Pressure against RPM Measurements. In color: Clusters derived from the k-means algorithm.

B. Anomaly Detection

During the operation of the ship, certain ranges for the sensor values are expected, as they happen regularly and signal a well-functioning machine. Furthermore, the manufacturers of the machinery, in this case, Böning and MTU, provide tolerances of the sensor values that alert the crew when a threshold is crossed. There is, however, a problem when several subsystems interact: A slightly increased cooling water temperature might not be an anomaly on its own, but when the cooling water temperatures of all other subsystems are cooler than usual, it might indicate a deteriorating cooling system of the first machine and should be reported. With a high number of interacting subsystems, overseeing all sensors at once becomes increasingly difficult. To help with that, ML

algorithms can be used to detect anomalies automatically [1]. The detection can be performed on live data or retrospectively on historic data, where they can help to gain new insights into the system. For example, a minor anomaly might be ignored during a SAR mission, as the engine works regardless. However, if this type of anomaly accumulates across several different ships of the same type, it might be worth investigating, as it might hint at a bigger problem. Thus, learning the behavior is not constrained to a single ship but can be expanded to a whole fleet in the future.

C. Predictive Maintenance

ML algorithms can use sensory machine data and information about maintenance routines in order to effectively predict the remaining useful life of a system [6]. However, in general information about how the maintenance has been performed and what parts had to be replaced is not available digitally. Therefore, detected anomalies can act as a guide on how maintenance could be improved. For example, if a certain type of anomaly only occurs in ships of the same type that operate in the north sea, maintenance routines for these ships should be adjusted, as future anomalies of the same kind are to be expected. Furthermore, by comparing data points to data of the same cluster, long-term trends can be detected and extrapolated, which allows to perform maintenance before failures occur.

III. Object Detection

In close, bilateral cooperation with the Wehrtechnische Dienststelle 71 (WTD71) for ships and naval weapons of the German Federal Armed Forces, Maritime Technology and Research, we have started field tests for the search and retrieval of persons and rescue equipment in the water. With several divers and various rescue equipment, accident victims were simulated and recorded with a state-of-the-art camera system. These images now serve as the basis for training the ML model to detect and track persons in distress.

In parallel, we use data from Naval Support Command (MUKdo) to classify naval vessels. Here the nation, type and capabilities of the ship are determined (according to NATO STANAG 1166). Another use case with the MUKdo is the classification of radar emitter.

A. Data

For the detection of people and rescue equipment, many hours of video footage are available, consisting of EO and IR recordings. These show divers and rescue equipment, as well as typical objects from the shipping industry. These include buoys, seabirds and drifting gear. The data were taken on different days in different weather conditions to create as much variety in the data as possible. Thus, the weather conditions change among the days from sunny and windless to wavy and overcast.

The data for ship classification shows ships and boats from various angles, from the horizontal perspective. The database consists of about 67,000 entries from most seagoing nations and shows units from small coastal patrol boats to aircraft carriers and submarines.

B. Conclusion

A pipeline is to be created for the detection and tracking of people and rescue equipment in the water, which receives both EO and IR data as a combined input (see Figure 3).

A pipeline for classifying nation, prefix and suffix has been created. This is the classification by multi-class problems for the nation and prefix and a multi-label multi-class problem for the suffix. This pipeline uses prior knowledge from the first network to flesh out subsequent outcomes.

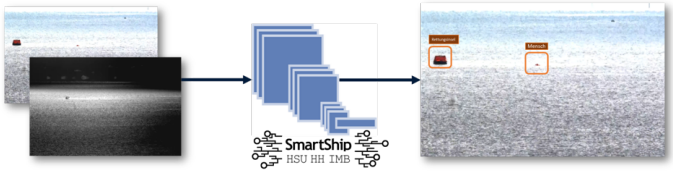

Figure 3. The schematic sequence for the detection of persons and rescue means is shown. On the left, the two video channels (EO and IR) are transferred to a neural network, which annotates the video feed and visually marks and classifies the victims. Here in the right picture on the left a life raft and on the right a WTD71 diver in a survival suit.

C. Experiments

Various networks of different sizes were tried to find the best converging network for object detection. The final model is based on the ResNet [4] architecture. Since the data set is very heterogeneous, different balancing solutions are tested. These include Stratifyer, Sampler and Loss-weights. The design of the classification task also turns out to be open to experimentation with respect to the multi-label-multi-class problem. Thus, combinations of classes as well as multi-hot vectors were used for this task.

IV. System Design

Condition monitoring and object detection tasks on ships require an optimal hardware/software solution to host the respective ML solvers, processing units and communication links (e.g. to sensors, camera, data). Therefore, we developed such an architecture and lay out its details in the following sections.

A. Hardware Optimization

The hardware architecture (Figure 4) is reflected in two distinctive implementations: edge (onboard) devices enhanced with LTE antennas and backend server, augmented with GPU accelerators. Onboard sensory data consists of three subsystems: Böning, MTU1, and MTU2. Böning metrics provide for the general ship data, such as indoor temperature, pressure, fuel levels, etc. while MTU represents engine parameters. Collected data proceeds via CAN [2] bus

interface to the onboard unit (OBU), which acts as a data logger and simple forwarding device. However, our future upgrades will extend its capabilities with data processing which are important for several reasons. First, the majority of engine (MTU) sensors are in fact not implemented, although broadcasting the default values. Therefore, by filtering only relevant sensors, we can significantly reduce the LTE data loads. Second, OBU receives only raw data in CAN-compatible J1939 [8] format and the decoding is performed in the backend. If the decoding is moved to the OBU, the ML models could be run on the ship itself. Finally, by having direct access to the onboard raw data, we can calibrate and optimize data formats and decoding pipelines, improving thus the overall system efficiency.

In addition to built-in sensors, the camera module is being installed to allow critical objects/person detection in water. Besides optical and gyroscopic data, the visual parameters could be complemented with weather and AIS data, creating a comprehensive digital representation of vessel deployment conditions.

In addition to the OBU, we provide the NVIDIA Jetson units, complemented with solid-state drives (SSD). Jetsons are light, compact, and low-power GPU-accelerated devices, adapted to real-time condition monitoring and anomaly detection. Paired with selected 256GB SSDs, they allow for more than a month of uninterrupted raw data logging, which exceeds greatly typical rescue mission intervals (1-2 days).

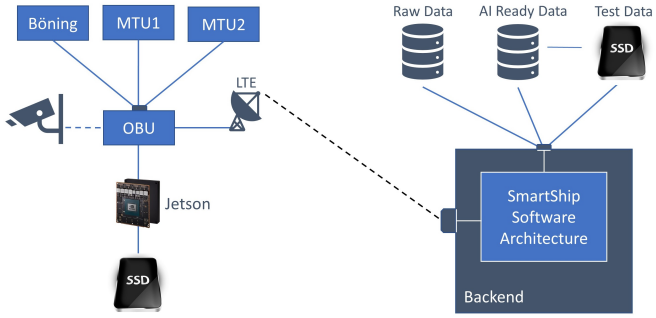

FIGURE 4. SMARTSHIP HARDWARE ARCHITECTURE

The server side is envisioned as the backbone for computationally heavy tasks, such as predictive maintenance (PM) model training, concurrent data processing, and fleet optimization. However, the exact implementation is not determined, as the hosting capabilities of the end user (DGzRS [3]) have to be defined first. Nevertheless, our system aims to be platform-independent by building on top of container software infrastructure. This approach provides great flexibility and the option to migrate the entire software stack from servers to the cloud and vice versa.

[3]German maritime search and rescue service

Also, regardless of the backend implementation, our solution allows for two high-capacity storage drives (raw/AI-ready), capable of one year of data logging each. In addition, the pre-production test environment envisions a third, faster SSD unit needed for experiments and model adjustments.

B. Software Architecture

The SmartShip software architecture (FIGURE 5) aims to provide a highly portable and platform-independent design that is easily adjustable to different edge/backend deployments. We achieve this by centralizing configuration options, allowing users to adapt the software stack to their own preferences. For instance, by selecting the appropriate data source, storage, and ML model, one can deploy the same system on a ship as well as on the backend server/cloud. In the following, we provide a short overview of the key software components.

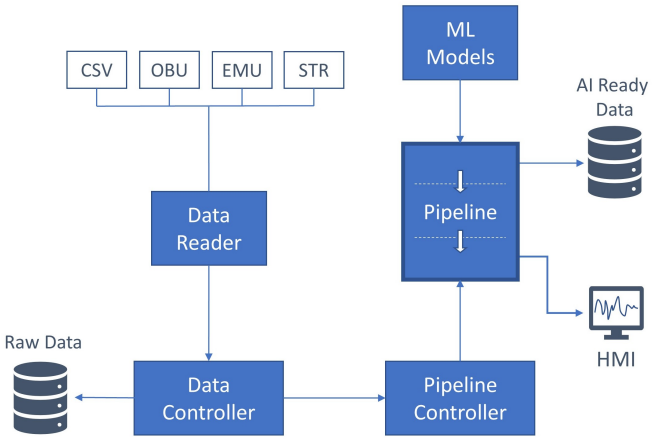

FIGURE 5. SMARTSHIP SOFTWARE ARCHITECTURE

Data Reader allows the selection of different sources (formats) of sensor data that should flow into the system: CSV logs, physical OBU interface, data emulator (EMU), or LTE stream (STR) of OBU raw data (FIGURE 4). However, in the production environment, CSV and OBU sources will converge into a single onboard interface while LTE streaming API remains as the backend connectivity option. After the source selection, the Data Reader notifies Data Controller if the new data is available and forwards it accordingly. Data Controller initiates storage of the Raw Data and signals the Pipeline Controller about the incoming traffic. Depending on the use case, deployment side, or user requirements, different ML models are engaged by the pipeline. Finally, Pipeline has two major outputs. In the first step, AI-ready data is created after initial raw data (pre)processing. Then, different ML models are called, allowing anomaly detection and predictive maintenance calculations. In the last step, based on user requests, data is visualized and displayed on maritime-certified screens. This modular approach allows a lot of flexibility in adding and removing different AI services

without influencing the core software structure. In other words, by wrapping each new model into a new container, extra features could be added over time or even migrated to a different platform, if scaling up the system surpasses the customer hosting limitations.

V. Conclusion and Outlook

In this paper, we explained the goals of the SmartShip project and gave an overview of the essential research components, summarized by ML models for condition monitoring, offshore object detection, and a ready-to-deploy hardware/software solution. An initial analysis showed that the onboard sensory data can be efficiently clustered and separated according to the engine states, which is a precondition for effective anomaly detection. Also, we indicated that object tracking algorithms can combine both electro-optical and infrared signals to assist in search and rescue missions. Finally, our system design developments are demonstrated through the emulation of the actual sensory data, allowing real-time condition monitoring and engine state evaluation. However, further tests should be undertaken in order to improve the overall system efficiency, reevaluate the findings in the dedicated hardware and migrate the created AI tools into a near-production prototype. There may also be an extension towards other maritime companies and sea-based organizations, as well as further cooperation with other universities and research institutes.

Acknowledgment

This research project (SmartShip) is funded by dtec.bw – Digitalization and Technology Research Center of the Bundeswehr. The authors gratefully acknowledge discussion and exchange with project partners: German Maritime Search and Rescue Service (DGzRS), OptoPrecision GmbH and JAKOTA Design Group GmbH.

References

[1] Raghavendra Chalapathy and Sanjay Chawla. Deep learning for anomaly detection: A survey.

[2] Ricardo De Andrade, Kleber N. Hodel, João Francisco Justo, Armando M. Laganá, Max Mauro Santos, and Zonghua Gu. Analytical and experimental performance evaluations of can-fd bus. *IEEE Access*, 6:21287–21295, 2018.

[3] Vincent Fortuin, Matthias Hüser, Francesco Locatello, Heiko Strathmann, and Gunnar Rätsch. Deep self-organization: Interpretable discrete representation learning on time series. In *ICLR*, 2019.

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[5] Iraklis Lazakis, Konstantinos Dikis, Anna Lito Michala, and Gerasimos Theotokatos. Advanced ship systems condition monitoring for enhanced inspection, maintenance and decision making in ship operations. *Transportation Research Procedia*, 14:1679–1688, 2016.

[6] Jay Lee, Fangji Wu, Wenyu Zhao, Masoud Ghaffari, Linxia Liao, and David Siegel. Prognostics and health management design for rotary machinery systems—reviews, methodology and applications. *Mechanical Systems and Signal Processing*, 42(1-2):314–334, 2014.

[7] Henrik Steude, Alexander Windmann, and Oliver Niggemann. Learning physical concepts in cps: A case study with a three-tank system. *IFAC-PapersOnLine*, 55(6):15–22, 2022.

[8] W. Voss. *A Comprehensible Guide to J1939*. Copperhill Technologies Corporation, 2008.