A Real-Time Depth Estimation System Based on Intel RealSense and Hailo-8 Edge AI Accelerator Using YOLOv8m

Manot Mapator[1], Chutimon Khaikaewr[2], Phonlawat Jintawiset[3], Chanatip Orathaiwan[4]*

[1]*School of Electronic Engineering, Suranaree University of Technology, Thailand*
[2]*School of Intelligent Embedded Systems and High Frequency Electronic Engineering, Suranaree University of Technology, Thailand*
[3]*School of Electronic Engineering, Suranaree University of Technology, Thailand*
[4]*School of Electrical Engineering, Suranaree University of Technology, Thailand*
**m6702991@g.sut.ac.th*

Keywords: depth estimation, RGB-D cameras, object detection, edge AI, Hailo-8.

Abstract

This research presents the development of a real-time object detection and depth-distance estimation system integrating an Intel RealSense depth camera with the Hailo-8 AI accelerator to measure object distances under real operating conditions. The system operates at high frame rates with low latency and low power consumption, while providing three-dimensional spatial coordinates (X, Y, Z) suitable for robotics and automation. The processing pipeline includes RGB-D acquisition, YOLOv8 inference compiled to HEF for Hailo-8 acceleration, and depth estimation from detected bounding boxes. Experimental results show approximately 30 FPS with a depth error of ≈5–15 mm, demonstrating the feasibility of edge AI–based depth perception for real-world deployment.

1 Introduction

Real-time depth perception is a critical capability for robotics, automation, and industrial inspection systems, where accurate three-dimensional object localization is required for safe navigation, reliable manipulation, and robust scene understanding [3], [4]. RGB-D sensors such as Intel RealSense cameras provide depth measurements with sufficient accuracy for these applications. Nevertheless, the joint processing of RGB-D data and real-time object detection remains computationally intensive, particularly when employing high-performance detectors from the YOLO family [1], [2].

On resource- and power-constrained edge platforms, CPU-only implementations impose inherent limitations in latency, throughput, and real-time stability, hindering the deployment of end-to-end RGB-D perception pipelines. Recent developments in edge AI accelerators provide a viable alternative by enabling higher inference throughput and improved energy efficiency compared with conventional CPU-based processing [6], [7]. Among these accelerators, the Hailo-8 device is designed for edge deep learning and has demonstrated competitive inference performance for industrial vision applications [8].

Despite these advances, applied studies that integrate Hailo-8 accelerators with RGB-D cameras and YOLO-based object detection into a complete real-time depth perception system remain limited. In particular, existing work rarely provides comprehensive end-to-end evaluations that include system-level performance metrics such as frame rate, end-to-end latency, real-time stability, and depth-estimation accuracy under practical operating conditions on low-power edge platforms.

In this paper, we present the design and experimental evaluation of a real-time object detection and depth estimation system that integrates an Intel RealSense camera with the Hailo-8 AI accelerator, using YOLOv8 as the detection backbone. The proposed system implements a synchronized end-to-end pipeline comprising RGB-D acquisition, hardware-accelerated object detection, bounding-box–depth association and real-time visualization using GStreamer. Experimental results evaluate frame rate, end-to-end latency, real-time stability, and Z-axis depth accuracy, demonstrating the feasibility and practicality of the proposed system for deployment in robotics and automation scenarios [1], [2], [8].

The rest of this paper is organized as follows. Section II reviews the related work. Section III describes the architecture of the proposed system. Section IV presents the design and implementation details. Section V discusses the experimental results, and Section VI concludes the paper.

2. Related Work

Related research on real-time object detection and depth estimation can be broadly categorized into three main directions: deep-learning–based object detection, the use of RGB-D data from Intel RealSense cameras, and hardware-accelerated approaches on edge platforms.

In deep-learning–based object detection, the YOLO family of networks has gained widespread popularity due to its ability

to achieve a good trade-off between speed and accuracy. The original YOLO models and their subsequent variants have demonstrated strong potential for real-time processing across a wide range of applications. However, most studies still primarily focus on accuracy-oriented metrics (e.g., mAP), rather than evaluating system performance under constraints such as power consumption, latency, and limited computational resources. In general, YOLO models are deployed on desktop systems or platforms equipped with high-performance GPUs for object-detection tasks in domains such as security surveillance, transportation, and intelligent robotics [1], [2].

For depth acquisition, RGB-D cameras such as Intel RealSense which rely on stereo-based depth sensing are widely adopted in robotics and automation because they provide depth data with acceptable accuracy across practical operating ranges. Numerous studies have investigated sensor calibration, depth-error behavior, and the influence of environmental conditions on depth quality [3], [4]. Nevertheless, these works typically analyze the sensor in isolation, without integrating real-time object detection together with depth information within a unified perception pipeline.

Alternative depth-estimation techniques have also been explored, such as chromatic aberration–based depth estimation in fluid environments, which utilizes wavelength-dependent refraction characteristics for distance inference (Mukai et al., 2018). Although fundamentally different from stereo-based RGB-D sensing, such studies highlight the diversity of depth-estimation methodologies and provide broader context for evaluating RGB-D systems [11].

In recent years, hardware acceleration on edge platforms has received considerable attention. Specialized devices such as NPUs and deep-learning accelerators have been introduced to reduce latency and energy consumption compared with CPU-only processing [6], [7]. The Hailo-8 device is a representative example, designed for computer-vision and deep-learning workloads, and has demonstrated competitive performance in real-time object-detection scenarios [8]. However, most related studies continue to focus primarily on RGB-only detection and do not incorporate depth information into perception or decision-making.

Several studies have attempted to integrate YOLO with RGB-D cameras for distance estimation. However, these systems are typically executed on high-resource computers or GPUs rather than low-power platforms, and often do not report system-level metrics such as frame rate, latency, stability, and depth-estimation error under realistic operating conditions [4]. Furthermore, although many works provide detailed evaluations of Intel RealSense accuracy, they generally do not combine object-detection models with hardware accelerators in a single, unified system [4].

On the other hand, a number of studies have begun applying YOLO-based object detection on low-power edge platforms such as Raspberry Pi. For example, a system employing a Raspberry Pi 4 with an RGB camera and a YOLO-model was developed for helmet-wearing detection in the petroleum industry, emphasizing low cost and ease of field deployment [9]. However, such systems typically process only RGB images without incorporating depth information to estimate distance or improve decision reliability. Moreover, their evaluations are often limited to accuracy and detection rate, without considering power consumption, latency, or long-term operational stability on edge platforms.

Therefore, this study focuses on the design and evaluation of a real-time object-detection and depth-estimation system that integrates an RGB-D camera with a YOLO model and hardware acceleration on an edge platform. The proposed system is evaluated in terms of accuracy, computational performance, and energy-related metrics to provide insights into its practical feasibility and limitations, together with guidelines for tuning the system under constrained computational resources.

In addition, image quality under challenging illumination conditions can significantly affect object-detection performance. Recently, Çiftçi (2025) proposed a real-time low-light image enhancement method based on autoencoder-driven histogram matching, demonstrating effective contrast improvement while maintaining computational efficiency. Such approaches could be integrated as a preprocessing stage in RGB-D perception systems to improve robustness under low-light indoor or industrial environments [10].

3 System Overview

This section describes the architecture of the proposed real-time object-detection and depth-estimation system. The system integrates an Intel RealSense RGB-D camera, a Raspberry Pi 5 edge platform, and the Hailo-8 AI accelerator into a synchronized processing pipeline, as illustrated in Fig. 1.

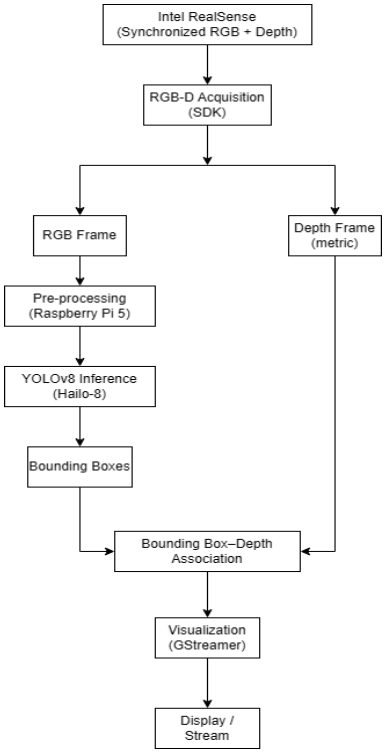

Fig. 1 Overall system architecture of the proposed RGB-D perception pipeline.

3.1 Overall System Architecture

As shown in Fig. 1, the system begins with an Intel RealSense camera used to capture synchronized color and depth data. The RealSense SDK is used to acquire both streams and to align the color pixels with the corresponding depth values at the same image locations.

The RGB frames are sent to the Raspberry Pi 5 for processing, including resizing and formatting to match the input requirements of the YOLOv8 model. The preprocessed frames are then forwarded to the Hailo-8 accelerator, where object detection is performed to generate bounding boxes and object classes. Meanwhile, the depth frames, expressed in metric units, are obtained directly from the RealSense SDK for each detected object. The system selects the region in the depth image corresponding to each bounding box and computes the depth value at the centroid of the box, which is then used as the Z-axis distance between the camera and the object.

In the final stage, the system overlays the detected bounding boxes and the computed distance values onto the color image, after which the results are displayed and streamed in real time via GStreamer, allowing immediate visualization and network distribution of the processed output.

3.2 RGB-D Data Acquisition

The RGB-D camera provides aligned color and depth images at a fixed resolution and frame rate. The depth data are represented in metric units, allowing direct estimation of object distances along the Z-axis. Synchronization between the color and depth streams is handled by the camera software development kit, ensuring accurate correspondence between detected objects in the color image and their associated depth values.

Prior to inference, the color images are resized and normalized according to the requirements of the object-detection model, while the depth images are preserved at their original resolution to reduce interpolation-induced errors during distance computation.

3.3 Hardware-Accelerated Object Detection

Object detection is performed using the YOLOv8 model, which is selected for its favorable balance between detection accuracy and efficiency. The trained model is converted into a hardware-compatible format using the Hailo toolchain and deployed on the Hailo-8 accelerator.

During operation, RGB images are streamed to the Hailo-8 for pipelined inference. The detection results, including object classes and bounding box coordinates, are then transmitted back to the host processor for subsequent depth computation.

3.4 Depth Estimation from Bounding Boxes

Depth estimation is performed by associating each detected object's bounding box with the corresponding region in the depth image. For each detected object, a Region of Interest (ROI) is defined based on the bounding box coordinates. Depth values within the ROI are then extracted, and statistical measures such as the median are applied to mitigate the effects of noise and outliers.

The resulting depth value represents the object's distance along the camera's Z-axis and can be further combined with the camera intrinsic parameters to compute the full three-dimensional (X, Y, Z) coordinates.

3.5 Real-Time Visualization Pipeline

To support real-time monitoring and visualization, the system incorporates a visualization module developed using GStreamer. Detected object bounding boxes, object classes, and estimated depth values are overlaid on the RGB video stream. The streaming-based architecture ensures that visualization introduces minimal additional system latency, while enabling continuous display and recording.

4 Experimental Setup

This section presents the experimental setup for evaluating the real-time object detection and depth estimation system, detailing the hardware and software configurations, test environments, and evaluation metrics to ensure experimental reproducibility.

4.1 Hardware Platform

The hardware configurations are specified in Table 1.

Table 1 The hardware configurations

Component	Specification
Camera	Intel RealSense D435
Edge Device	Raspberry Pi 5 (8 GB RAM)
Accelerator	Hailo-8
Display	HDMI 1080p
Lighting	Indoor, constant illumination
Power Supply	5 VDC

The experiments were conducted on a Raspberry Pi 5 equipped with 8 GB of RAM and connected to a Hailo-8 accelerator. An Intel RealSense D435 camera, connected via a USB 3.0 interface, was used to simultaneously capture color and depth data.

The camera was mounted on the robot, as shown in Fig. 2, with the distance between the camera lens and the ground fixed at 1100 mm to ensure consistent image geometry and viewpoint throughout the experiments.

All experiments were conducted indoors under controlled and consistent lighting conditions in order to minimize the effects of illumination changes.

Fig. 2 The camera was mounted on the robot

4.2 Software Configuration

The software configurations are specified in Table 2.

Table 2 The software configurations

Component	Specification
Raspberry Pi OS	Raspberry Pi OS (64bit) (2025-12-04)
Intel RealSense SDK	pyrealsense
HailoSDK	HailoRT + GStreamer integration
YOLOv8	YOLOv8m
GStreamer	GStreamer 1.0 (PyObject)

The system runs on Raspberry Pi OS (64-bit), and the color and depth frames are acquired through the Intel RealSense SDK. For object detection, the pretrained YOLOv8m model was used. After installing the Hailo libraries on the Raspberry Pi 5, the model could be executed on the hardware accelerator. This model is capable of detecting people, vehicles, animals, and common household objects.

However, in this study, the Python implementation was configured to detect only water bottles. For real-time visualization and video streaming, GStreamer is used to overlay the detection bounding boxes and distance values onto the RGB images.

4.3 Test Environment and Objects

The system was evaluated by placing a water bottle within the camera's field of view. The bottle was positioned at various locations along the camera's X, Y, and Z axes in order to investigate the behavior of the 3D coordinate estimation and the resulting FPS performance. The camera position was kept fixed throughout the experiments, as shown in Fig. 2, to ensure reproducibility.

4.4 Experimental Procedure

The experiments were designed to evaluate both the real-time performance and the depth-estimation accuracy of the proposed system. A water bottle was placed at different distances ranging from 500 mm to 1100 mm along the camera's Z-axis.

For the real-time performance evaluation, the system was run with the Intel RealSense camera configured to 640 × 480 resolution at 30 FPS, while the input images were resized to 640 × 640 for YOLO inference. Three water bottles were continuously detected for 5 minutes. FPS values were logged every second into a CSV file to compute the average FPS, and the video output was visually inspected to identify any stuttering or freezing during operation.

For the depth-estimation accuracy experiment, the same camera configuration was used, but only a single bottle was placed in the scene to clearly observe the centroid position. The bottle was positioned at distances of 580, 680, 800 and 1000 mm from the camera. Ground-truth distances were measured using a tape measure, while depth values reported by the camera were recorded to a CSV file once per second for 1 minute. The recorded values were averaged, and the absolute error was calculated as |ground-truth − estimated.

4.5 Performance Metrics

System performance is evaluated using the following metrics:

Frame Rate (FPS) : The average number of frames the system can process per second is used to evaluate its real-time capability. In many real-time vision applications, processing rates close to the camera frame rate (≈30 FPS) are generally considered sufficient.

2. Real-Time Stability: The consistency of frame rate during prolonged operation.

3. Depth Estimation Accuracy: In prior studies evaluating depth accuracy of RGB-D sensors, typical depth error values have been reported on the order of several millimeters when comparing to reference measurements. For example, a benchmarking method for estimating depth data errors in RGB-D systems demonstrated RMS depth errors on the order of 10 mm for commonly used sensors [5] , suggesting that a ±10–20 mm error bound along the Z-axis is reasonable for practical evaluations.

5 Results and Discussion

5.1 Real-Time Performance

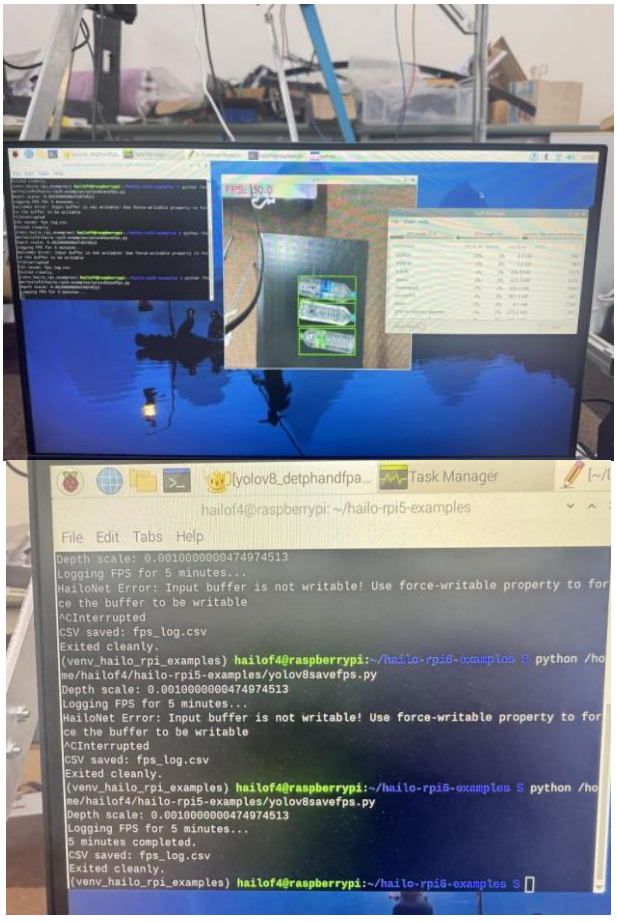

Fig. 3 Experimental setup showing real-time object detection and system resource utilization during operation.

Fig. 3 shows the real-time execution of the proposed system, including the detection output and the Task Manager display. The results confirm that the system can continuously process frames in real time while keeping CPU and memory usage within acceptable limits.

The frame rate was logged once per second for 5 minutes, resulting in 300 samples. The average FPS was computed using the arithmetic mean:

$$\mathrm{FPS}_{\mathrm{avg}} = \frac{1}{N}\sum_{i=1}^{N} FPS_i, N = 300 \quad (1)$$

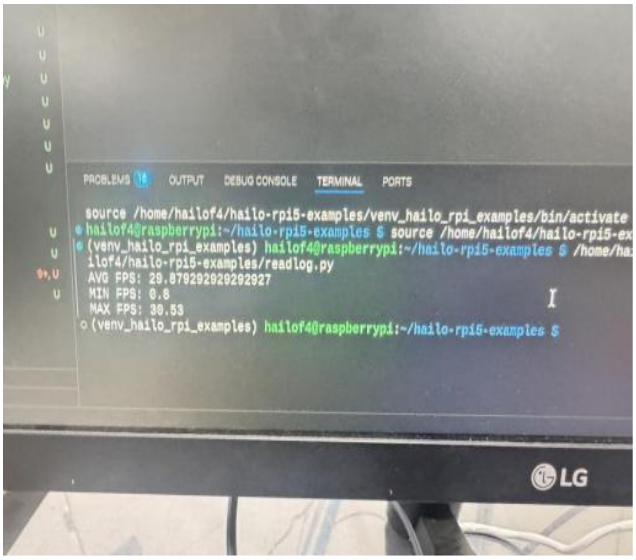

Fig. 4 Results frame-rate statistics (maximum, minimum, and average FPS) during real-time operation

Fig. 4 shows the recorded frame-rate statistics during continuous operation. The system achieved an average processing rate of 29.87 FPS, with a maximum of 30.53 FPS, which is close to the camera frame rate. A temporary drop to 0.80 FPS was observed at the beginning of the recording, which was caused by pipeline initialization and logging overhead rather than sustained processing slowdown. Importantly, the frame-rate drop occurred only once during initialization and did not affect steady-state performance.

Overall, the measured results indicate that the proposed system is capable of maintaining real-time operation with stable frame-rate performance throughout the experiment.

Table 3 Real-time performance

Metric	Value
Max FPS	30.52
Min FPS	0.80
Average FPS	29.87

5.2 Depth-Estimation Accuracy

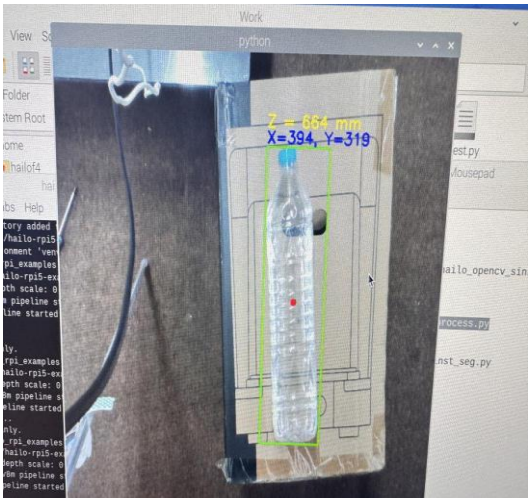

Fig. 5 Example of detected object with the estimated depth (Z-axis) obtained from the Intel RealSense RGB-D camera.

Fig. 5 shows an example of the proposed system during operation. The YOLO model detects the water bottle, and the centroid of the bounding box is used to read the depth value from the RealSense depth image. The displayed value represents the distance of the object along the Z-axis relative to the camera.

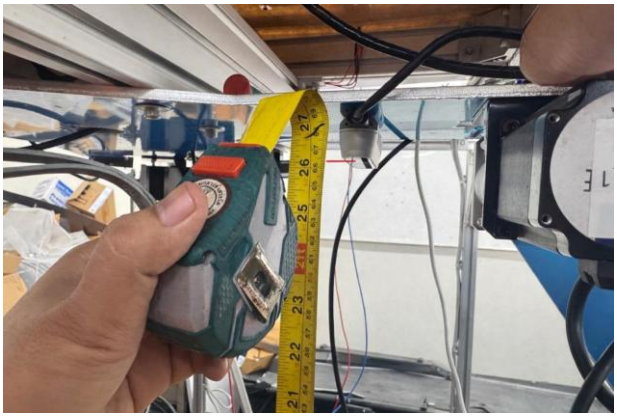

Fig. 6 Ground-truth distance measurement using a tape measure during the depth-estimation experiments.

Fig. 6 shows how to obtain the ground-truth distance, a tape measure was used to measure the distance from the camera lens to the front surface of the bottle, as illustrated in Fig. 6. These measurements were used to compute the absolute depth error.

For each distance, both the RealSense-estimated depth and the tape-measured ground-truth value were recorded. The absolute error was computed as

$$\mathrm{Error} = |\,\mathrm{ground\ truth} - \mathrm{measured}\,| \quad (2)$$

The measured depth value was obtained by averaging the depth readings reported by the camera over a 10-second interval. Depth readings were recorded once per second into a

CSV file, resulting in 10 samples for each distance, and the averaged results are summarized in Table 4.

Table 4 Averaged RealSense Depth Measurements and Corresponding Ground-Truth Distances

Ground truth (mm)	Measured (mm)	Error (mm)
580 mm	585.47	5.47
680 mm	687.67	7.67
800 mm	812.33	12.33
1000 mm	1010.82	10.82

Table 4 summarizes the depth-estimation performance of the proposed system. Across all tested distances, the measured values closely matched the ground-truth distances, with absolute errors ranging from 5 to 15 mm. These results indicate that the depth estimation is highly stable and sufficiently accurate for practical indoor applications within the evaluated range.

Note that the evaluation was performed within a short indoor range, and larger distances may introduce higher errors.

6 Conclusion and Future Work

This paper presents a real-time object detection and depth estimation system that integrates an Intel RealSense RGB-D camera with a Hailo-8 hardware accelerator on a Raspberry Pi 5 platform. The proposed system consists of synchronized acquisition of color and depth images, hardware-accelerated object detection using a YOLO model, and depth estimation based on the centroid of the bounding box of the detected water bottle.

The experimental results show that the system can operate at approximately 30 frames per second with stable real-time performance. In addition, the depth-estimation results indicate that the measured distances are close to the ground-truth values, with errors of approximately 5–15 mm within the tested indoor range. These results suggest that the proposed system is well suited for applications in robotics and automation that require real-time perception on low-power or resource-constrained platforms.

Future work may incorporate real-time low-light enhancement techniques and alternative depth estimation strategies to further improve robustness under challenging environmental conditions.

7 References

[1] Redmon, J., Divvala, S., Girshick, R., Farhadi, A.:
'You Only Look Once: Unified, Real-Time Object Detection',
Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR),
Las Vegas, NV, USA, 2016, pp. 779–788,
doi: 10.1109/CVPR.2016.91

[2] Jocher, G., et al.:
'YOLOv8: Ultralytics YOLO Object Detection',
Ultralytics Docs, 2023. Available at:
https://docs.ultralytics.com

[3] Saxena, A., Schulte, J., Ng, A.Y.:
'Depth Estimation Using Monocular and Stereo Cues',
Proc. Int. Joint Conf. Artificial Intelligence (IJCAI),
2007, pp. 2197–2203

[4] Sturm, J., Engelhard, N., Endres, F., Burgard, W., Cremers, D.:
'A Benchmark for the Evaluation of RGB-D SLAM Systems',
Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS),
Vilamoura, Portugal, 2012, pp. 573–580,
doi: 10.1109/IROS.2012.6385773

[5] Cabrera, E.V., Ortiz, L.E., da Silva, B.M.F., Clua, E.W.G., Gonçalves, L.M.G.:
'A versatile method for depth data error estimation in RGB-D sensors',
Sensors, 2018, 18, (9), pp. 1–22,
doi: 10.3390/s18093122

[6] Mittal, S., Umesh, S.:
'A survey on hardware accelerators and optimization techniques for RNNs',
Journal of Systems Architecture, 2021, 112, pp. 1–18,
doi: 10.1016/j.sysarc.2020.101839

[7] Chen, Y., Xie, Y., Song, L., Chen, F., Tang, T.:
'A survey of accelerator architectures for deep neural networks',
Engineering, 2020, 6, (3), pp. 264–274,
doi: 10.1016/j.eng.2020.01.007

[8] Hailo Technologies Ltd.:
'Hailo-8 Deep Learning Processor – Product Brief and Performance Benchmarks',
2022. Available at: https://hailo.ai

[9] Alhaila, S., Almelhem, I., Alsnafi, A., et al.:
'YOLO-based helmet detection system for safety compliance in oil and gas industry',
Proc. 5th Int. Conf. Industrial Engineering and Artificial Intelligence (IEAI),
Bangkok, Thailand, 2024, pp. 16–23,
doi: 10.1109/IEAI62569.2024.00012

[10] Çiftçi, S.:
'Low-light image enhancement using autoencoder-based histogram matching',
Engineering Science and Technology, an International Journal,
2025, 72, Art. 102236

[11] Mukai, N., Matsuura, Y., Oishi, M., Oshima, M.:
'Chromatic Aberration Based Depth Estimation in a Fluid Field',
Journal of Image and Graphics, 2018, 6, (1), pp. 59–63,
doi: 10.18178/joig.6.1.59-63