European Processor Initiative Demonstration of Integrated Semi-Autonomous Driving System

D. Hofman, M. Brcic, M. Kovac
University of Zagreb
Faculty of Electrical Engineering and Computing
Zagreb, Croatia

T. Hotfilter, J. Becker
Karlsruhe Institute of Technology (KIT)
Karlsruhe, Germany

D. Reinhardt
Bavarian Motor Works
Department of New Technologies
Munich, Germany

S. M. Grigorescu
Transilvania University of Brasov
Brasov, Romania

R. Stevens, T. T. Vo
Kalray
Montbonnot, France

***Abstract*—The European Processor Initiative (EPI) is developing a processor for various sectors, including the automotive industry. To benchmark the new processor, EPI uses a test vehicle to demonstrate different use cases, like semi-autonomous driving. In this paper, we focus on object detection and describe the use cases in the perception stage of autonomous driving. Therefore, we introduce four applications that include face recognition, blind spot detection, near-range object detection and far-range spatial perception to cover a wide range of different domains. Each use case runs on a different architecture representing future domains of the EPI processor, allowing for optimal utilization and performance. Our evaluation shows that all use cases can be mapped on the future EPI platform. The performance of the image processing tasks such as blind spot detection meets real-time requirements and runs at over 30 FPS. The face recognition task can process an image with low-power consumption, allowing for battery-powered inference. Finally, we validate our results and findings using a BMW X5 car as a real-world demonstrator.**

***Index Terms*—Autonomous Vehicles, Object Detection, Automotive Safety and Security, European Processor Initiative**

I. Introduction

It is expected that 58 million autonomous vehicles with at least autonomy level 3 will be sold in 2030, which is about 15 times more than in 2022. The European Union has identified the automotive industry as one of the main customers for the chips being developed within the European Processor Initiative (EPI). EPI develops new hardware IPs along with applications to be demonstrated with the new chips. The automotive section of the project is developing a test vehicle to demonstrate different use cases of European technology in a semi-autonomous vehicle. This paper presents four use cases covering facial recognition and short/mid/far-range detection of obstacles and surrounding areas.

Advanced Driver Assistance Systems (ADAS) can be divided into three conceptual processing stages, depending on their ability to autonomously drive the vehicle. The three stages are perception, planning, and control. Vehicle perception deals with numerous sensors used for detection of the surrounding area, mainly using cameras and LiDAR sensors. In the planning stage, a trajectory is planned to suggest the best route for driving the vehicle. This trajectory is used in the control stage to set the actual movement of the car without the need for human interaction. In this paper, we focus on the first two stages, perception and planning, to demonstrate a future drive in the test car without a human controlling the car.

II. Related work

Test cars have been used in numerous projects to demonstrate newly developed systems. Obstacle avoidance, path planning and control for autonomous vehicles were presented in [1] on an ARTEMIPS test car, showing its adaptability for effective obstacle avoidance.

Our first use case, face recognition, has been widely studied already. Currently, traditional algorithms like holistic or local methods are falling behind in favor of deep neural networks [2]. In the recent decade, neural network approaches improved continuously, achieving now better performance than traditional algorithms [3]. To tackle the computational complexity of neural networks, their inference is now moving further into specialized accelerators running, e.g., in FPGAs [4]. For example, Zhuge et al. evaluated different configurations of HLS generated face detection accelerators on FPGAs, showing a 3.75x latency improvement in comparison to GPUs [5]. More recently, Liu et al. demonstrated a latency of 1.3 ms per face recognition on their heterogeneous computing system, consisting of CPUs, GPUs and FPGAs [6]. However, their approach consumes over 25 W. Since the feasibility of FPGAs in acceleration of face recognition has been proven, we will focus on a low-energy implementation, which can be hosted on the EPI embedded FPGA.

Short-range blind spot detection systems are used in autonomous driving for obstacle avoidance. Multimodal sensor fusion blind spot detection combines several modes from possible sensor choices: camera, LiDAR, RADAR, etc. [7]. Prior to circa mid-2010s, classical methods based on manually crafted features dominated the field [8]. By now, more modern approaches like the YOLO object detector family [9] emerged and are showing tremendous performance. The current challenge in the area is to find a powerful but energy-

efficient platform and neural architectures with a good accuracy/computational cost trade-off to enable real-time performance in embedded systems.

From a sensor perspective, fisheye cameras (field of view $\geq 100°$) are today used for a better coverage and lower costs. However, the severe distortion of fisheye cameras still poses a major challenge. Currently, there are two approaches, either rectification [10] or working directly with distorted input [11]. The first usually requires extra transformation to emphasize the corners [12], while for the latter dedicated networks like spherical CNNs [13] have to be used. With LiDAR or RADAR information, distances can be estimated, enabling a full blind spot detection [14].

Extending this perception system with standard video cameras, enables also far-range perception. Here, CNNs play a central role. Architectures, like YOLO [9] or SqueezeNet [15], are now standards in 2D-object detection. For semantic segmentation that aims to identify certain structures in the driving scene, such as the lanes or the drivable area, standard deep learning architectures like SegNet [16] or encoder-decoder systems became standards/ Object detection and semantic segmentation tasks have been successfully combined into single deep learning architectures which simultaneously predict both the 2D bounding boxes of objects, and semantic regions of the driving scene. These types of algorithms are called panoptic segmentation networks. Notable systems are Uber's UPSNet [17] and Detectron2 [18] from Meta research. More recent approaches for self-driving cars even integrate LIDAR information [19], which outputs 3D poses of objects of interest in real-world Cartesian coordinates.

III. System design

Our ADAS system has been demonstrated on a BMW test car BMW X5, which was equipped with additional hardware for this purpose. We wanted to derive an automotive version of a chip from a high-performance architecture. Our intention and focus were to test the future chip architecture without actually steering the vehicle. Therefore, we will not control the car autonomously. Our output is the proposed trajectory for the car shown on the central information display of our test car.

For our demonstration, we use an X5 test car of BMW. We use series components and enrich the system with an additional sensor setup. For our setup, we use:

- 4x Sony 8MPx cameras, collecting raw image data from the front and rearview of the car
- 4x Nvidia Jetson AGX platforms to convert raw image data to JPEGs and do a first object detection
- 1x LiDAR Hesai Technology Pandar40 sensor, which is used to measure the range of detected obstacles to the front
- 4x iCAMs (series component), which is typically used for a bird's-eye perspective
- 4x radar sensors (series component), which are used to detect other cars on the road
- 1x high-precision c103 GPS sensor system from uBlox

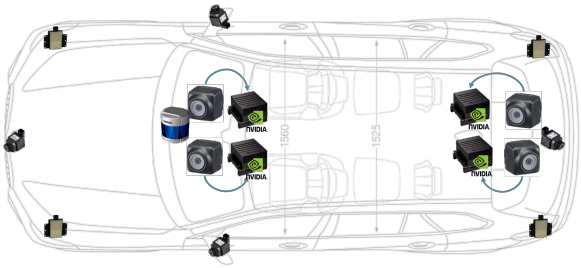

Fig. 1. Location of sensor components within the BMW test car.

- 1x MPPA Reference Board K200 from Kalray for mid-range objects detection
- interconnection network and MCU used for transmission of data between the components and sensor-fusion

The location of the cameras and radars is shown in Fig. 1.

The following subsections explain the design of the subsystems for face recognition, blind-spot detection, mid-range detection and far-range detection.

A. Near-range object detection

The near-range use case for our automotive object detection pipeline is face recognition. If a known person is detected, our system can adjust the user settings in the vehicle. Later, it might be used for access control. The main objectives for the near-range detection system are low-energy consumption along with high precision to allow operation when the engine is not running and energy is only provided by the battery.

Our face recognition system builds on neural networks. To meet the low-power requirement, we choose SqueezeNet [15], which is a highly optimized neural network topology, as the starting point. For our face recognition, we train the network from scratch using the Casia web faces dataset and pruned around 70% of the weights, achieving an accuracy of 91.1%.

From the hardware perspective, our face recognition system runs on the embedded FPGA (eFPGA) tile of the EPI chip. The aforementioned neural network is executed on a custom-designed accelerator for the eFPGA [20]. Since the accelerator is a custom build, we can also adjust the precision of the operands to enable quantization. Having the accelerator on the eFPGA allows us to reconfigure the accelerator on the fly, e.g., the face recognition can be replaced by an application that is required in a driving situation once the ride starts. The eFPGA is coupled with an Infineon AURIX safety microcontroller, that handles safety critical tasks like evaluating the recognized person and interfacing with the vehicle.

B. Blind spot detection

The near-range blind spot detection use case has two central components (Fig. 2): The Jaguar 12-bit JPEG decoding library and a neural network object detector. It uses information from two different external sensors - cameras and radars. Based on the camera data, we detect moving vehicles in blind spots. In parallel, we collect information about detected objects

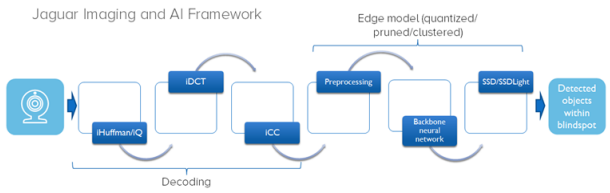

Fig. 2. Jaguar Imaging and AI Framework base blocks

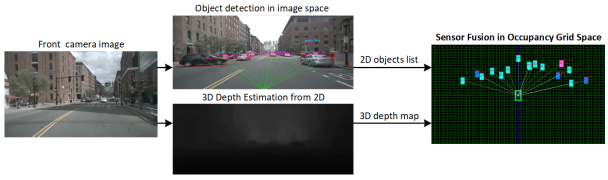

Fig. 3. Environment perception and sensor fusion in occupancy grid space

using radars. All detected objects are processed and fused in the sensor-fusion component at a high level. Camera data is decoded by our Jaguar image processing framework and then processed by the Jaguar AI framework, which provides bounding box information of detected vehicles. Both frameworks were tested on x86 Intel architectures and on NVIDIA GPUs, as well as on ARM architectures.

Fig. 2 shows the components of the Jaguar Imaging and AI Framework. Image data from two fisheye cameras located left and right of the car is sent to the Jaguar library via the car's Ethernet. The library supports 8 and 12-bit JPEG image decoding and preparation for transmission to the edge AI object detection model. Decoding is supported for Huffman tables used in the JPEG standard and optimized Huffman tables generated by the encoder. The library has been successfully validated using a real image data from the BMW X5 test car.

Decoded images are now passed to the Jaguar AI framework for object detection. The AI network is trained on several labeled open-source datasets, some with rectilinear and some with fisheye images, like Waymo, Audi, Kitti, Kitti-360 and Woodscape. As network topology candidates, we chose several state-of-the-art models: MobileNet-V2, EfficientDet-D0, YOLOv4-tiny, PP-YOLOv2 and NanoDet. We have created this vast toolkit to drive experimentation to find the best solution. In addition, we use image data collected while driving our test car. All the networks were pretrained on a rectilinear COCO dataset and had to pass through extensive fine-tuning for fisheye images. Out of the tested models, YOLOv4-tiny had the best accuracy on GPUs. However, in our constrained environment with ARM cores, a quantized and pruned MobileNetv2 – the weakest and oldest of all models - shows the best computing performance using TFLite as framework. Hence, we select the computationally most efficient MobileNet-V2 for camera mode and rely on LiDAR object recognition as a supplement. The outputs from this phase are bounding boxes around detected objects and their classification – this is identical for all the models in our portfolio.

C. Mid-range object detection

Mid-range object detection implies being in real-time traffic to detect, identify and classify obstacles and surrounding area to trigger car reactions or safety controls within a certain range. The characteristic of this environment is that the car is constantly moving, potential obstacles are also in motion, and targets can be numerous in heavy traffic. This requires an efficient identification and classification algorithm with low latency. Our mid-range detection use case, hence, uses 4K cameras in real traffic situations. Their data is fed into the MPPA (Massively Parallel Processor Array) processor of Kalray installed in the BMW demonstrator car, which is also part of the later EPI chip. Using the KaNN (Kalray Neural Network) we can compile a trained neural network model from, e.g., Tensorflow into a version that can be accelerated on the MPPA.

D. Far-range objects detection

Far-range environment perception in 3D space is achieved by constructing an occupancy grid representation of the space surrounding the car, through the usage of two stereo cameras and a LiDAR sensor. One for detecting objects in the image space and another for calculating the depth between the camera and the imaged scene. For object detection, we choose a YOLOv5 single-shot detector, trained on the Berkeley Deep Drive dataset for autonomous driving. It provides an objects list, encoding 2D object locations in the image space. The camera and LiDAR Fusion can then take place directly on our computation platform MCPv2. A grid map is generated as a representation of the environment. Our setup for obstacle detection consists of a LiDAR placed on top of the car, and two Sony cameras placed underneath the LiDAR, capturing data from the environment ahead of the car. One of the cameras has a field of view (FOV) of 120 degrees, designed for obtaining a wider shot including close-range objects, while the other has a FOV of 30 degrees, capturing detailed information at a longer distance from the car, but with a narrow shot.

To plan the motion of the car, the 3D reconstructed model should embed the traffic participants' locations in real-world metrics. Hence, the second deep neural network is a CNN trained to estimate the sensor-surface distance from monocular cameras. It provides a one-to-one mapping from RGB input images to a grey-level image, in which each pixel encodes the sensor-surface distance. As shown in the lower part of Fig. 3, the brighter the pixel value, the further away the imaged surface is.

A reconstruction of the whole 360° environment around the car is achieved by processing the images acquired from different sensors. The pipeline in Fig. 4 showcases the full process, from reading live images from the Sony cameras, running object detection on the Jetson AGX modules, sending the data packets via Ethernet to the MCPv2 platform where the sensor fusion is processed and then passing the results to custom clients, in this case, being a visualizer. Each video

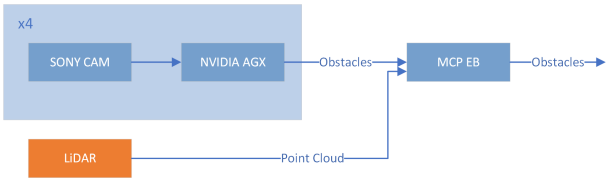

Fig. 4. Camera and LiDAR-based processing pipeline for far-range object detection

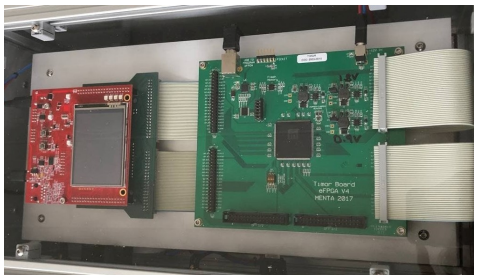

Fig. 5. Modular box to demonstrate the near-range detection system for face recognition, featuring the eFPGA and an Infineon AURIX

stream is processed using a dedicated NVIDIA AGX Xavier embedded computer board, using the aforementioned two deep neural networks. The networks' output, i.e., the 2D objects list and depth maps, are synchronized on the target MCPv2 platform, which acts as a delivery server for the five camera streams. The environment is finally reconstructed as a binary occupancy grid, having a resolution of $0.2m$. If an obstacle is located at a specific real-world location, then its location in the occupancy grid will be marked with an occupancy flag.

IV. Experimental Setup

Our testing environment is a BMW X5 test car with cameras, radars, network and processing elements (see Fig. 1).

The near-range detection system is built in a modular box that can be easily attached to the car, but can also work independently to increase flexibility. Fig. 5 shows the box, with the two used tiles: the eFPGA and the AURIX. Input data for the near-range recognition system is provided by multiple cameras placed inside the vehicle pointing to each seat. Our modular box can communicate with the car and the cameras using standard Ethernet.

The blind spot detection use case is based on the detection of objects in the area outside the car where the driver cannot see them directly or in the mirrors. Detection is performed with two independent systems and the detected objects are fused to get more certainty in the detection. The first detection system uses cameras that are located on the left and right sides of a car. Both cameras have a resolution of 1280x960 px and record 30 frames per second. Decoding and detection using the Jaguar framework is performed on an Nvidia AGX device. Detected objects are then forwarded through a network to the R-Car-H3 platform, where it is fused with the radar data. The second detection system is based on radars, which are located on the rear side of the car. Detected objects and

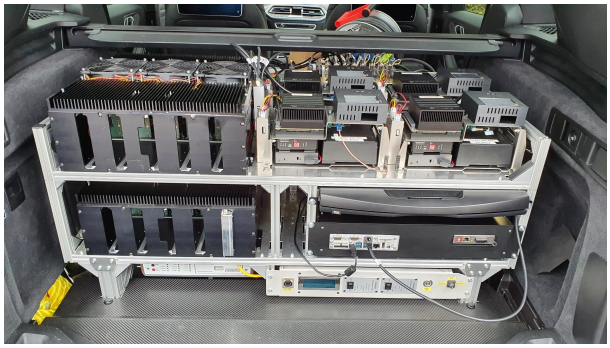

Fig. 6. Real-world setup of the rack including two MCPv2s, the four zonal controllers and the Neoverse platform (bottom right, below the monitor panel)

free-space information are transferred over the car's network to the R-Car-H3. The entire mid-range detection system is then tested for performance. Tests of the Jaguar Imaging and AI Framework are executed in a laboratory environment on similar configurations. Performance was tested using separable parts of the framework to get the approximate throughput of the whole blind spot detection system.

The mid-range detection system uses the data from the stereo 4K camera, which is sent directly via Ethernet to the Kalray MPPA board, to be able to analyze the video in real-time and detect obstacles.

The test environment for the far-range perception system makes use of the four Sony cameras connected to four NVIDIA Jetson AGX boards, as well as the Hesai Pandar64 LiDAR and the MCP processor (Fig. 1). NVIDIA GPUs are used for DNN inference of the camera images for 2D object detection. Detected objects are sent to the MCPv2 for reconstructing the driving environment using fusion with LiDAR data, as explained in section III-D.

All components are validated in an isolated laboratory setup and after the complete integration, additional verification is performed with the test car and on actual streets.

V. Results

Within the EPI project, we implement and test all previously described components and integrate them into our BMW X5 test car. Since we are building a prototype system, components are larger than those of a final product and are fitted in a rack in the trunk of the X5 (Fig. 6). Our ADAS system consists of three conceptual processing stages - perception, planning and control. First, various sensors detect the environment of the car, which is then computed. In the next state, a trajectory for driving is computed and finally the car's actuators are controlled accordingly.

A. Perception

Within this stage, obstacles around the car are detected and the sensors' data are fused to a common environmental model. It is important to detect all obstacles, independent of how close or far located to the car, or if they move or stand still. The focus is on synchronous and real-time efficient data

processing. Hence, the output of this step is an environmental real-time model with all obstacles and their exact position around the car.

1) Face detection: To map the face recognition neural network efficiently onto the eFPGA, we apply various optimizations and perform a design space exploration. In our application, the base network is tuned to just recognize faces, therefore many redundant parameters occur. Pruning them yields a 75% parameter reduction. In addition, we explored the impact of quantization. With ten-bit parameters and four-bit activations, we were able to reduce the memory movements by 70%, while maintaining an accuracy of 84% [21]. To bring the neural network to the eFPGA in a resource-optimized way, we explored various design options. First, we fixed the bit widths according to the found quantization scheme. With the 52 available DSPs, we were then able to fit 16 accelerator cores on the eFPGA. In-depth details of our design space exploration and the architecture can be found in a previous work [20]. With the hardware and optimizations in place, we can process one input image within 150ms. Thereby, the eFPGA consumes about 4.7 mW per inference, which results in a performance of 0.48 GOP/s or 99.37 GOPs/W. The overall latency as well as the energy-to-performance ratio meets the requirements and demonstrates that a near-range face recognition is feasible on the EPI platform.

2) Blind spot detection: Our evaluation in the laboratory environment proves the ability to detect other vehicles in blind spots. The quality of the AI processing framework is highly influenced by the available computational power. Hence, our AI model has to be scaled to a suitable size. The performance of the Jaguar Imaging framework for decoding is tested on an Ubuntu x64 machine with four cores. Parallelization shows great scaling with an increasing number of cores because decoding of individual images is independent. On four threads, we can process 24.8 images per second. For comparison, a single thread only achieves 8.48 images per second.

Our AI framework uses MobileNet-v2 as architecture with an SSD head and frame sizes of 320x320 pixels. Our network is trained (i.e. fine-tuned from COCO pretrained network) directly on fisheye camera frames from the Woodscape dataset. The network performance was measured on an NVIDIA K80. We have measured mAP@[.5, .95]=0.114 and mAR@[.5, .95]=0.246 at maximal detections of 100. K80 is a GPU that was released in 2014 and its performance is a lower bound on AGX. Using TensorFlow 2.7 we got a throughput of 29 FPS with a batch size of 1. After that, camera results are fused with radar data. The computational complexity of our fusion is much lower than what is required for decoding. Hence, it can run in real-time even with other workloads on the same R-Car-H3 device. The overall performance of our blind spot detection using Jaguar Imaging and AI Framework indicates that it can run sufficiently at 30 FPS.

3) Mid-range objects detection: Our mid-range detection set-up is tested in various driving configurations, for pedestrian detection, identifying incoming and outgoing vehicles, on medium roads with traffic lights and finally on high-speed motorways (Fig. 7). As network, we use a YOLOv3, with inputs of 416x416. The inference has been performed using FP16.32 arithmetic, with KaNN targeting 5 clusters of the MPPA processor. When operating at 1.2 GHz, the performance is about 20 FPS, about 3 TFLOPS peak.

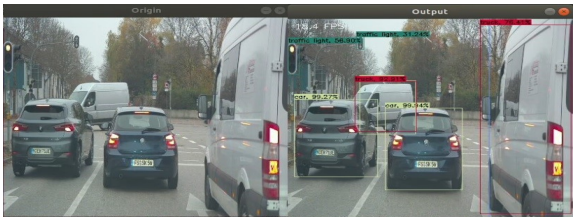

Fig. 7. Mid-range detection in real traffic

4) Far-range objects detection: Our far-range detection generates the exact position of our test car, with bounding boxes and classifications of obstacles around it. The horizontal position of the boxes is obtained by mapping the bounding boxes generated through object detection; while the vertical position is given by scaling the depth obtained from LiDAR to the grid map (Fig. 3). The far-range perception system is tested offline with the Berkeley autonomous driving dataset, as well as online, using real-time data from the car's cameras and LiDARs. Since the goal of EPI is to go beyond the state-of-the-art in embedded processor development, we focus on the computation time achieved by our deep learning pipeline. For 2D object detection and depth perception, we achieve abut 18 FPS and 0.9 FPS, respectively, on each of the four NVIDIA boards. However, sensor fusion on the MCP had a higher sampling rate of 50 FPS. Thus, the overall system was running at a speed equal to the depth perception rate. For real-time performance, we choose to deactivate the depth perception neural network and rely only on the LiDAR for estimating 3D distances.

B. Planning & Control

In the planning step, the environmental model, with all obstacles around the car, is used to plan the trajectory around blocking obstacles. Furthermore, the car needs to be located precisely on the road to plan overtaking maneuvers or simply stay within its lane. In the planning phase, the Arm Neoverse platform represents the processor of the EPI chip. As for path planning, a lot of CPU computation power is necessary, this is the ideal platform. The output of this step is finally a trajectory for our test car.

For the control phase, this trajectory is shown on the Central Information Display (CID) within our BMW X5 test car, since actual driving is very challenging and not the focus of the EPI project. It represents a suggestion from the EPI Neoverse chip to the driver. All data is processed using Apollo's Dreamview.

VI. Discussion

Our results on face detection underline the performance of the EPI processor and its components in energy-constrained environments. With only 4.7 mW of energy, we can power

the face recognition from the car battery while the engine is not running. The processing time of 150 ms is still reasonable, since it does not require real-time performance. Performance measurements of our blind spot detection show an overall performance of around 30 FPS. This fulfills the requirements of the automotive use case. Results observed in mid-range detection correspond to what is expected for accuracy and latency. Additional enhancements are expected with optimizations of the MPPA toolchain, together with the possibility to process various sensors on several compute clusters in parallel. Our far-range perception can reconstruct the driving environment around the car with a sufficient sampling rate. However, this is only achieved after deactivating the depth estimation neural network. This points us to follow further improvements in embedded DNNs, which would enable larger neural networks, with multiple task outputs, to perform at a sampling rate acceptable for self-driving cars.

VII. Conclusion

Within this paper, we showed our work and final demonstrator setup for semi-autonomous driving of the EPI test car. The work was integrated into a demonstrator car and to develop the architecture of the central MCP. This platform should house all computing units of several EPI processors. Computer vision and object detection happen through additional 4K cameras from Sony and a LiDAR scanner from Hesai. All other sensors are already released to BMW's customers and reused for EPI from the existing BMW X5 test car. The car was equipped with an additional high-precise GPS, which was used on additional HD maps. Sensors for Short-Range-Radars, additional cameras for lane-keeping and all sensors to measure the behavior of the car must not be replaced. Thus, BMW supplied the partners with an all-in-one test car solution and helped to integrate those solutions. For software integration, all systems are based on Yocto Linux. They have all relevant drivers integrated and can be used for further refinements.

The near-range face recognition runs on the eFPGA and AURIX tile, which allows for both low-energy consumption and safe execution. Face recognition can be performed in 150 ms while consuming only 4.7 mW, which makes its operation feasible even when the car engine is not running. Short-range objection detection was realized with a combined and sensor-fused operation of cameras and radars, which produce the high safety requirements for ADAS. The result is a very close environmental model around the car, where the long-range sensor cannot capture any obstacles. Our mid-range detection, based on stereo 4K cameras, was realized on a Kalray MPPA. The solution acts as a fallback solution in case of system and sensor failure. Far-range perception was enabled through the combination of cameras and LiDARs in four zones. All sensor information is fused to a common 3D environmental model around the car. Finally, we realized a planning stage on the ARM Neoverse system.

The performance achieved by our test shows, the potential performance of the future EPI chip, as all tested components can be integrated into the EPI chip.

Acknowledgement

This project has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No. 826647 and from the European High Performance Computing Joint Undertaking (JU) under Framework Partnership Agreement No 800928 and Specific Grant Agreement No101036168 EPI SGA2.

References

[1] H. Laghmara *et al.*, "Obstacle avoidance, path planning and control for autonomous vehicles," *IEEE Intelligent Vehicles Symposium, Proceedings*, vol. 2019-June, no. Iv, pp. 529–534, 2019.

[2] H. Zhou *et al.*, "Recent advances on singlemodal and multimodal face recognition: A survey," *IEEE Transactions on Human-Machine Systems*, vol. 44, no. 6, pp. 701–716, 2014.

[3] M. Wang and W. Deng, "Deep face recognition: A survey," *Neurocomputing*, vol. 429, pp. 215–244, 2021.

[4] A. Shawahna, S. M. Sait, and A. El-Maleh, "FPGA-Based accelerators of deep learning networks for learning and classification: A review," *IEEE Access*, vol. 7, pp. 7823–7859, 2019.

[5] C. Zhuge *et al.*, "Face recognition with hybrid efficient convolution algorithms on FPGAs," *Proceedings of the ACM Great Lakes Symposium on VLSI, GLSVLSI*, no. March, pp. 123–128, 2018.

[6] X. Liu *et al.*, "Collaborative Edge Computing With FPGA-Based CNN Accelerators for Energy-Efficient and Time-Aware Face Tracking System," *IEEE Transactions on Computational Social Systems*, vol. 9, no. 1, pp. 252–266, 2022.

[7] V. K. Kukkala, J. Tunnell, S. Pasricha, and T. Bradley, "Advanced Driver-Assistance Systems: A Path Toward Autonomous Vehicles," *IEEE Consumer Electronics Magazine*, vol. 7, no. 5, pp. 18–25, Sep. 2018, conference Name: IEEE Consumer Electronics Magazine.

[8] B.-F. Wu, C.-C. Kao, Y.-F. Li, and M.-Y. Tsai, "A Real-Time Embedded Blind Spot Safety Assistance System," *International Journal of Vehicular Technology*, vol. 2012, 2012. [Online]. Available: https://trid.trb.org/view/1138934

[9] P. Jiang, D. Ergu, F. Liu, Y. Cai, and B. Ma, "A Review of Yolo Algorithm Developments," *Procedia Computer Science*, vol. 199, pp. 1066–1073, Jan. 2022.

[10] K. Zhao, K. Liao, C. Lin, M. Liu, and Y. Zhao, "Joint distortion rectification and super-resolution for self-driving scene perception," *Neurocomputing*, vol. 435, pp. 176–185, May 2021.

[11] H. Rashed *et al.*, *FisheyeYOLO: Object Detection on Fisheye Cameras for Autonomous Driving*, Dec. 2020.

[12] E. Plaut, E. B. Yaacov, and B. E. Shlomo, "3D Object Detection from a Single Fisheye Image Without a Single Fisheye Training Image," May 2021, arXiv:2003.03759 [cs].

[13] T. S. Cohen, M. Geiger, J. Koehler, and M. Welling, "Spherical CNNs," Feb. 2018, arXiv:1801.10130 [cs, stat].

[14] L. Peng, F. Liu, Z. Yu, S. Yan, D. Deng, Z. Yang, H. Liu, and D. Cai, "Lidar Point Cloud Guided Monocular 3D Object Detection," Jul. 2022, arXiv:2104.09035 [cs].

[15] F. N. Iandola *et al.*, "Squeezenet: Alexnet-level accuracy with 50x fewer parameters and <0.5mb model size," 2016.

[16] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 12, pp. 2481–2495, 2017.

[17] Y. Xiong *et al.*, "Upsnet: A unified panoptic segmentation network," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2019-June, 2019, pp. 8810–8818.

[18] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo, and R. Girshick, "Detectron2," https://github.com/facebookresearch/detectron2, 2019.

[19] M. Liang, B. Yang, Y. Chen, R. Hu, and R. Urtasun, "Multi-task multi-sensor fusion for 3D object detection," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2019-June, 2019, pp. 7337–7345.

[20] T. Hotfilter *et al.*, "Towards reconfigurable accelerators in hpc: Designing a multipurpose efpga tile for heterogeneous socs," in *2022 Design, Automation and Test in Europe Conference and Exhibition (DATE)*, Mar 2022, p. 628–631.

[21] I. Walter *et al.*, "Embedded Face Recognition for Personalized Services in the Assistive Robotics," in *Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, 2021, pp. 339–350.