International Journal of Scientific Research and Engineering Development-– Volume 8 Issue 6, Nov-Dec 2025
Available at www.ijsred.com

RESEARCH ARTICLE OPEN ACCESS

EDGE DETECTION AND IMAGE SMOOTHING USING SOBEL FILTERING ON A ZYNQ SoC PLATFORM

A Arathi
Department of Electronics and Communication Engineering
The Oxford College of engineering
Bangalore, India
arathiii24@gmail.com

Chandana ML
Department of Electronics and Communication Engineering
The Oxford College of engineering
Bangalore, India
ml.chandana02@gmail.com

Bhumika YH
Department of Electronics and Communication Engineering
The Oxford College of engineering
Bangalore, India
bhumikayh05@gmail.com

Ganavi K
Department of Electronics and Communication Engineering
The Oxford College of engineering
Bangalore, India
ganavi0202@gmail.com

SWATI KUMARI ROY
Assistant Professor,
Department of Electronics and Communication Engineering
The Oxford College of engineering
Bangalore, India
royn8417@gmail.com

----------**********----------

Abstract:

Real-time edge detection is a fundamental requirement in embedded vision applications such as robotics, industrial inspection, surveillance, and autonomous systems Traditional software-based Sobel edge detection methods are limited by sequential execution, leading to increased processing latency. To address this issue, this work presents a hardware-accelerated Sobel edge detection system implemented on a Xilinx Zynq System-on-Chip (SoC). The design leverages FPGA parallelism for high-speed, low-latency image processing, while the ARM processor handles control and memory operations. Bitmap images stored in DDR memory are transferred to the programmable logic using AXI DMA. A fully pipelined Verilog-based accelerator performs Sobel convolution, gradient magnitude calculation, and thresholding, enabling continuous pixel processing at one pixel per clock cycle. Experimental results obtained through simulation and hardware validation using Vivado 2018.2 demonstrate significant performance improvement over CPU-based execution, producing accurate real-time edge outputs. Although extended operation results in increased device temperature, requiring thermal optimization, the proposed system remains efficient, scalable, and suitable for real-time embedded image-processing applications.

Keywords— **Sobel operator, FPGA acceleration, Zynq SoC, real-time image processing, AXI DMA, Verilog HDL.**

----------**********----------

I. INTRODUCTION

Digital image processing has become a core component of modern embedded systems, enabling a wide range of applications such as autonomous navigation, medical imaging, industrial inspection, surveillance, and robotics [1], [6]. Among the various operations involved in computer vision, edge

detection is one of the most fundamental tasks, as it identifies object boundaries, structural features, and regions of interest by detecting sharp intensity variations within an image [2]. Accurate and real-time edge extraction is particularly important for systems that require fast and reliable decision-making, including autonomous robots and safety-critical monitoring platforms [3].

The Sobel operator is one of the most commonly used edge detection techniques due to its simplicity, low computational overhead, and effectiveness in highlighting intensity gradients [8]. However, implementing Sobel filtering on general-purpose processors often fails to satisfy real-time constraints because of sequential execution and the large number of convolution operations involved [4]. Although GPUs offer improved parallel processing capability, they are generally unsuitable for low-power embedded platforms due to higher energy consumption, increased memory bandwidth requirements, and additional processing latency [5]. Previous research has demonstrated the benefits of FPGA-based image processing through the use of hardware convolution engines, optimized filtering architectures, and hybrid processing pipelines [6], [7]. These approaches have shown significant improvements in execution speed and determinism when compared to software-based implementations. However, many existing works primarily focus on individual processing modules and do not fully address system-level challenges such as efficient DDR memory access, DMA scheduling, processor-to-logic communication, and continuous high-resolution image streaming [8], [9].

These limitations highlight the need for a fully integrated and optimized FPGA-based architecture capable of performing real-time Sobel edge detection with minimal processor intervention and efficient memory management. The objective of this work is to design and implement a fully pipelined Sobel edge detection accelerator on a Xilinx Zynq System-on-Chip (SoC). The proposed system processes bitmap images directly from DDR memory using AXI DMA streaming, combining the flexibility of an ARM processor with the high parallelism offered by FPGA logic [6], [10].

The key contributions of this work are summarized as follows:

1. A hardware–software co-design that integrates ARM-based system control with FPGA-based pixel-level processing.
2. A fully pipelined Sobel operator implemented in Verilog HDL, achieving one pixel per clock cycle throughput.
3. Direct processing of BMP images from DDR memory without intermediate format conversion.
4. Complete system validation through simulation and real-time hardware execution.
5. Performance evaluation demonstrating significant improvements in speed, latency, and computational efficiency compared to software-based implementations.

Overall, this work demonstrates that a carefully optimized FPGA pipeline can provide deterministic, scalable, and high-performance edge detection, making it well suited for real-time embedded vision applications [7], [10].

II. LITERATURE SURVEY

An Digital image processing has received considerable attention in fields such as medical diagnostics, object recognition, and autonomous systems, where retaining key structural information like edges and textures is crucial. Numerous approaches have been developed to improve image quality without compromising real-time performance. Among these, classical edge detection techniques such as the Laplacian of Gaussian (LoG) and Sobel operators remain widely used. The LoG method is particularly effective in emphasizing fine structural details, making it suitable for feature extraction tasks [1]. In contrast, while the Canny edge detector offers high detection accuracy, its computational complexity limits its suitability for real-time embedded applications [2].

To overcome the constraints of software-based implementations, researchers have increasingly turned to hardware-accelerated solutions. FPGA-based Sobel edge detection architectures have demonstrated notable improvements in processing

speed and throughput compared to CPU-based approaches, especially in applications such as vehicle detection and object identification [3]. Optimized two-dimensional convolution architectures for FPGA platforms have further contributed to reduced processing latency and faster image computation [4]. In addition, FPGA-based eye-gaze tracking systems highlight the effectiveness of hardware acceleration for real-time vision-driven human–machine interaction [5].

As the demand for high-speed and low-latency image processing continues to grow, hardware–software co-design approaches using FPGA platforms have gained widespread interest. Configurable image processing systems developed using Verilog HDL on FPGA devices have shown the ability to deliver reliable real-time performance, particularly in medical and clinical imaging applications [6]. Furthermore, hybrid filtering techniques that combine edge detection with noise reduction have been successfully implemented on FPGA, achieving deterministic operation with high throughput [7].

Recent studies have also focused on improving design efficiency and system-level integration. FPGA–Simulink-based accelerated image processing frameworks have been proposed to simplify development while preserving real-time performance [8]. Gaussian filtering implemented on FPGA has proven effective for image pre-processing and noise suppression in real-time environments [9]. Additionally, Verilog-based FPGA systems designed for medical image enhancement further demonstrate the advantages of hardware acceleration in achieving efficient and dependable image processing solutions [10].
Overall, existing research strongly indicates that FPGA-based architectures offer substantial benefits in terms of processing speed, determinism, and real-time capability, making them highly suitable for implementing edge detection and image processing algorithms in modern embedded vision systems.

III. METHODOLOGY

A. System Design

The proposed system is implemented on a Xilinx Zynq System-on-Chip (SoC), which combines an ARM-based Processing System (PS) with programmable FPGA logic (PL) to enable real-time hardware acceleration for Sobel edge detection. The input image is used directly in bitmap (BMP) format, allowing raw pixel values to be accessed without compression or decoding, thereby simplifying the data flow and reducing processing overhead. The image is stored in external DDR memory connected to the PS, from where an AXI DMA controller streams the pixel data to the custom image-processing hardware in the PL using the AXI4-Stream interface.

The programmable logic includes dedicated hardware blocks such as line buffers, convolution units, gradient computation modules, and magnitude calculation circuits to implement the Sobel filter in a fully pipelined manner. As pixels stream through the hardware, edge detection is performed continuously in real time. The processed output is transferred back to DDR memory through the DMA, and completion interrupts are sent to the PS for further display, analysis, or storage. Proper clock management ensures synchronization between the PS and PL domains. This architecture achieves low-latency and high-throughput edge detection by effectively combining software control with FPGA parallelism, making it suitable for real-time embedded image-processing applications.

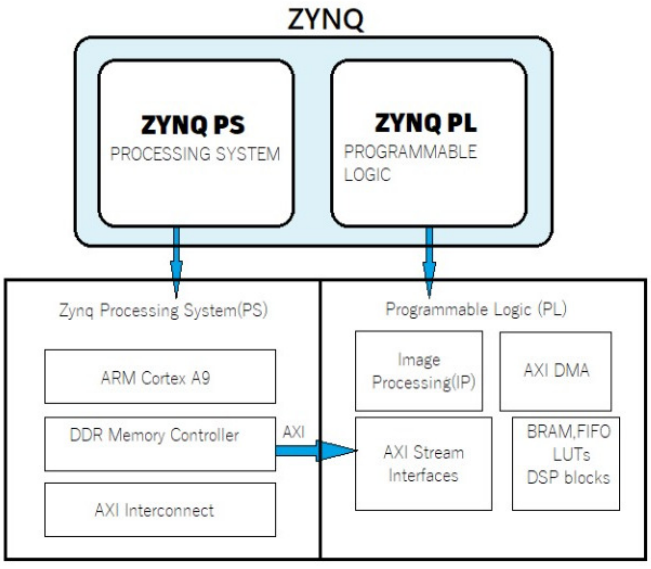

Fig -1: Zynq SoC Architecture

B. *Algorithms*

The Sobel operator is a classical edge detection technique that calculates the spatial gradient of an image to highlight regions with rapid intensity variations, typically corresponding to object boundaries. It uses two orthogonal 3 × 3 convolution masks, as illustrated in Fig-2, where one kernel detects horizontal transitions and the other identifies vertical intensity changes. By convolving the input grayscale image with these masks, the horizontal and vertical gradient components, denoted as Gx and Gy, respectively, are obtained.
The gradient magnitude at each pixel is then computed using the standard expression,

$$G(x, y) = \sqrt{(Gx^2 + Gy^2)} \quad \text{.....equation(1)}$$

-1	0	1
-2	0	2
-3	0	3

Gx

1	2	3
0	0	0
-1	-2	-3

Gy

Fig-2: Sobel convolution kernels

However, for real-time or hardware-based implementations, a simplified form such as,

$$G(x,y) = |Gx| + |Gy| \quad \text{.....equation(2)}$$

is frequently adopted reduce computational complexity 3 to while retaining acceptable detection performance.

The direction of the edge can also be calculated using $\tan^{-1}(Gy/Gx)$, although many applications focus solely on the magnitude of the output for visualization and post-processing. The kernels shown in Figure 2 inherently perform smoothing owing to their larger size compared to operators such as t he Roberts Cross. This smoothing effect results in a higher noise tolerance, making Sobel filters suitable for real-world images. However, it also causes the edges to appear thicker. To improve computational efficiency, a pseudo-convolution implementation, such as the kernel arrangement shown in Figure 3, can be used, enabling a fast estimation of the gradient magnitude in a single pass.

P_1	P_2	P_3
P_4	P_5	P_6
P_7	P_8	P_9

Fig-3: Pseudo-convolution kernels are applied to achieve rapid approximation of the gradient magnitude.

Using this kernel the approximate magnitude is given by:

$$|G|=|(P_1+2\times P_2+P_3)-(P_7+2\times P_8+P_9)|+|(P_3+2\times P_6+P_9)-(P_1+2\times P_4+P_7)| \quad \text{.....equation(3)}$$

A related method, the Prewitt operator, shown in equation(3), uses similar masks but offers reduced isotropy and slightly poorer edge localization than the Canny operator. Nevertheless, the Sobel operator remains widely adopted because of its simplicity, computational efficiency, and suitability for FPGA-based acceleration and embedded real-time vision systems.

Steps for FPGA-Based Sobel Filtering:

1. Import the BMP image and store it in external DDR memory connected to the Zynq PS.
2. Initialize the AXI DMA for high-speed data transfer between DDR and PL.
3. Configure the custom spatial filter IP in PL via AXI4-Lite registers.
4. Stream pixel data from DDR to PL through AXI4-Stream using DMA (MM2S).
5. Use line buffers in PL to create a 3×3 sliding window for convolution.
6. Apply Sobel or spatial filter kernels with MAC units to compute intensity values.
7. Transfer processed data back to DDR via DMA (S2MM).
8. Generate an interrupt (IRQ_F2P) to signal frame completion.
9. Read and display, store, or transmit the processed image from DDR.
10. Validate output visually and through simulation for correctness, latency, throughput, and resource usage.

The experimental setup is based on implementing spatial filtering and Sobel edge detection hardware on the Xilinx platform, where the input BMP image is loaded into external DDR memory and processed in real time by a custom accelerator implemented in the Programmable Logic.

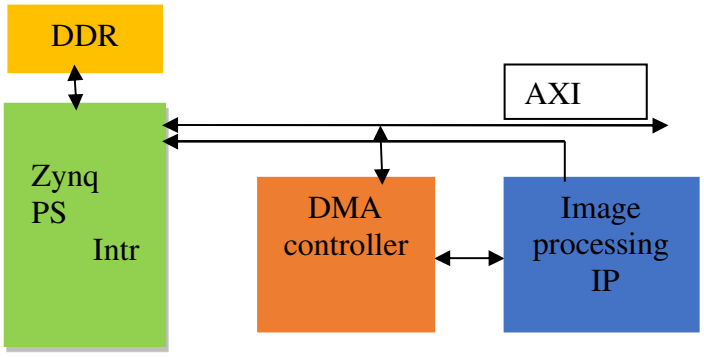

Fig-4: Target System Architecture

The design was developed and implemented using Xilinx Vivado 18.2, which was used for block design, RTL development, synthesis, implementation, and IP integration. Vitis/SDK was used to write and execute the embedded software required for DMA configuration, interrupt handling, and system control. The hardware platform includes DDR

memory, AXI interconnects for communication between the PS and PL, and an interrupt mechanism to signal frame completion.

Simulation and verification were conducted within the Vivado environment, and real-time testing was performed using test images provided by a host system. This setup enabled the evaluation of the system in terms of throughput, latency, filtering accuracy, and FPGA resource utilization.

IV. IMPLEMENTATION

The proposed Sobel edge detection system was implemented on a Xilinx Zynq SoC (system on chip) platform using a hardware/software co-design approach. The programmable logic (PL) performs real-time spatial filtering and gradient computation, whereas the processing system (PS) handles system configuration, memory management, and control flow. The following subsections describe the hardware and software platforms used, Verilog HDL development flow, Vivado toolchain workflow, and experimental setup used for validating the system.

A. Hardware and Software.

The proposed system was implemented on the Xilinx Zynq System-on-Chip (SoC), which combines a dual-core ARM Cortex-A9 processor with FPGA programmable logic on a single platform. This heterogeneous architecture enables compute-intensive Sobel edge detection to be executed in hardware, while system configuration and data management are handled by the ARM processor through DMA control.

Xilinx Vivado 2018.2 was used as the development environment, providing an integrated flow for design entry, synthesis, IP integration, simulation, and hardware debugging. The IP Integrator simplified system-level integration, while the Vivado Logic Analyzer supported real-time signal observation during hardware execution.

The Sobel accelerator was developed using Verilog HDL at the RTL level, allowing efficient parallel computation and low-latency processing. AXI interfaces were employed for system communication, with AXI4-Lite used for control and AXI-Stream for high-speed pixel data transfer between DDR memory, DMA, and the custom hardware core. External DDR memory was used to store image frames, supporting high-resolution processing without FPGA memory constraints.

B. Verilog HDL Design Flow

The Verilog design follows a structured hierarchical methodology to ensure reusability, modularity, and synthesis efficiency. The expandable architecture divides the system into well-defined blocks that communicate through standardized signals and generate outputs in a continuous streaming format.

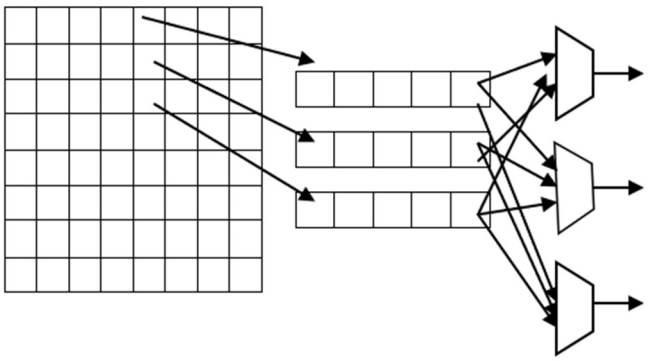

Fig-5: Line Buffer

At the lowest level, line buffer modules store three consecutive image rows to enable 3×3 window-based Sobel processing, using shift registers and block RAM to support continuous high-speed data flow. As pixels stream in, a nine-pixel window is generated and passed to the convolution stage, where horizontal and vertical Sobel kernels are applied in parallel to compute the gradient components G_x and G_y within a single clock cycle. The resulting edge magnitude is then calculated using either an exact or approximate method, with scaling and saturation logic to prevent overflow and preserve contrast. Finite state machines control data flow and synchronization between modules, and the final processed pixel stream is transmitted through the AXI-Stream output interface. Functional correctness was validated through simulation prior to synthesis and hardware deployment.

C. Vivado Workflow

The FPGA design was developed using Xilinx Vivado 2018.2 following a structured workflow. System components such as the Zynq Processing System, AXI DMA, interconnects, GPIO, and the custom Sobel IP were integrated using the IP Integrator, allowing graphical configuration of clocks, resets, and AXI interfaces.

The Verilog RTL design was then synthesized, where Vivado mapped logic to FPGA resources and reported timing and utilization. This was followed by placement and routing, during which timing constraints were analyzed and optimized to achieve timing closure at the target clock frequency. After successful implementation, the FPGA bitstream was generated and programmed onto the device.

Finally, hardware validation was performed using ARM-based software to initialize DDR memory, configure DMA, and control data transfer through memory-mapped I/O, ensuring efficient and continuous image processing with minimal CPU involvement.

D. Experimental Setup

The system was validated using grayscale bitmap images stored directly in external DDR memory, removing the need for intermediate conversion and minimizing latency. Images are loaded into DDR via JTAG, SD card, or external communication, while the ARM processor configures the AXI DMA engine to stream pixel data directly to the Sobel hardware accelerator. The programmable logic performs real-time edge detection on a pixel-per-cycle basis using line buffers, and the processed output is returned via DMA to DDR memory for storage or display. This setup allows results to be compared with software implementations in terms of edge sharpness, gradient clarity, noise suppression, and processing speed.

Hardware acceleration with DMA significantly reduces processor workload, enabling the ARM core to handle supervisory tasks instead of pixel-level processing. The combination of Verilog RTL design, Vivado implementation, and the Zynq SoC's parallel architecture provides low-latency, deterministic performance suitable for embedded vision applications. Additionally, the system maintains flexibility for future enhancements, including threshold adaptation, multi-kernel support, and chained image processing stages, while achieving real-time Sobel edge detection efficiently.

V. RESULTS AND ANALYSIS

The proposed Sobel-based edge detection architecture was successfully implemented on a Xilinx Zynq SoC platform using Verilog HDL and Vivado 2018.2 software. The system was verified at multiple stages, including simulations, software–hardware integration, and real-time hardware execution.

A. Simulation Results

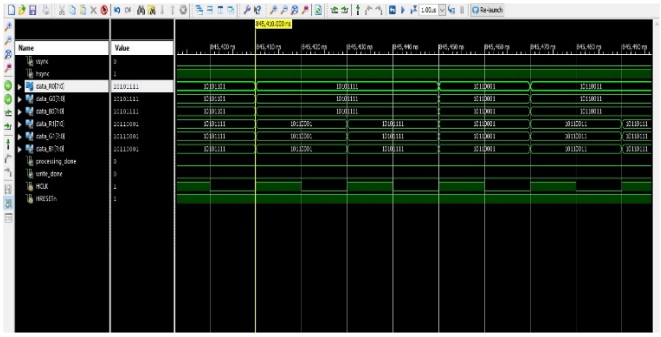

Fig-6: Simulation Output of Sobel Filter

Before hardware testing, each unit of the design was validated through functional simulations using the Vivado Simulator.

Test benches were created to evaluate the following:-

The simulation waveforms confirmed that the architecture produced accurate gradient values for each pixel stream. After pipeline filling, the system processes one pixel per clock cycle, demonstrating a fully parallel and pipelined hardware execution. The control logic and DMA signalling were verified to ensure a proper data handshake between the Processing System (PS) and Programmable Logic (PL).

Fig-7: Input Image & Output Image of Edge Detection

The design was synthesized, implemented, and deployed on a Zynq-7000 FPGA. Images in bitmap format were transferred through the AXI DMA from the DDR memory to the custom Sobel accelerator and written back after processing.

The results demonstrate that the Sobel filter implementation effectively identifies edges across diverse image types, preserving critical boundary information while maintaining computational efficiency.

B. Resource Utilization

The implementation results generated in Vivado 2018.2 show moderate usage of FPGA resources. The main resource contributions come from:

- LUTs and registers used for pixel datapaths and pipeline staging.
- DSP units used for multiply–accumulate operations
- BRAM blocks used to implement the three- line buffer.

Resource	Utilization	Available	Utilization %
LUT	1685	53200	3.17
LUTRAM	1154	17400	6.63
FF	261	106400	0.25
BRAM	0.50	140	0.36
IO	23	200	11.50

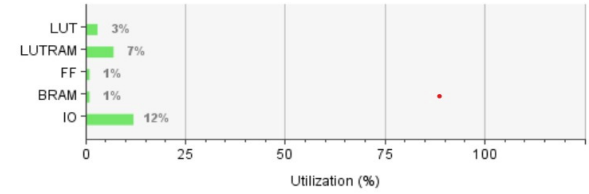

Fig-8: Utilization Report of Sobel Filter

The moderate utilization leaves sufficient programmable logic further enhancements such as multi-filter support, noise reduction, or more advanced algorithms like Canny or Laplacian filtering.

C. Power Report

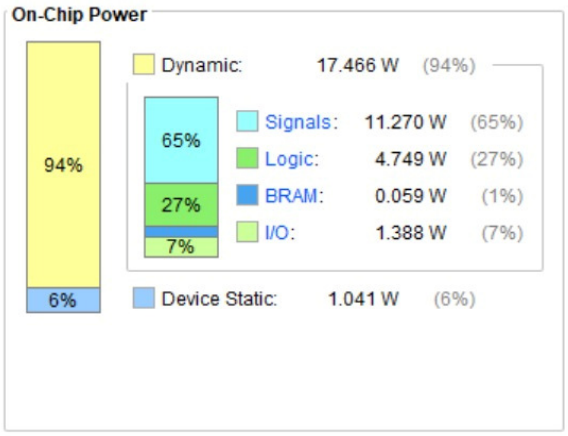

Fig-9: On-Chip Power Report of Sobel Filter

The on-chip power analysis shows that the total power consumption is dominated by dynamic power, which accounts for approximately 94% of the overall power usage, while static device power contributes only 6%. Among the dynamic components, signal activity is the major contributor at 65%, followed by logic power at 27%, indicating high switching activity due to continuous pixel processing. I/O power accounts for 7%, reflecting data movement through AXI interfaces, whereas BRAM power consumption is minimal at 1%, showing efficient memory utilization. This power distribution confirms that the design effectively exploits FPGA parallelism but also highlights the need for thermal management during prolonged operation.

D. Comparison with Conventional Methods

Parameter	**CPU-Based Sobel**	**FPGA-Based Sobel (Proposed)**
Execution Style	Sequential processing	Parallel and pipelined processing
Throughput	Low to moderate	High (one pixel per clock cycle)
Latency	High	Very low
Real-Time Capability	Limited	Yes (real-time)
Processor Load	High CPU utilization	Minimal processor involvement
Power Efficiency	Lower	Higher (energy efficient)
Noise Handling	Moderate	Better due to stable hardware timing

Table -1: Comparison of CPU based & FPGA-Based Sobel Filter

VI. CONCLUSIONS

This work presents a Sobel edge detection accelerator implemented on the Xilinx Zynq SoC, featuring DMA-based streaming, AXI memory communication, and a fully pipelined Verilog engine achieving one pixel per clock cycle. Hardware validation confirms accurate edge extraction across various image features. Prolonged operation causes notable temperature rise, highlighting the need for thermal management and power optimization in future designs. Overall, the system delivers low-latency, high-throughput edge detection, reducing processor load while preserving FPGA resources for additional processing stages.

Future Scope :

Future enhancements could improve performance, flexibility, and reliability. Thermal management strategies, algorithmic upgrades like Gaussian smoothing and non-max suppression, and support for multiple edge operators selectable at runtime can boost efficiency and edge quality. Dynamic frequency scaling and direct integration with MIPI or CSI-2 cameras would enable live image acquisition while reducing processor load. These improvements would make the system more robust, scalable, and suitable for diverse real-time embedded vision applications.

REFERENCES

[1] Y. Cao, N. Wei, X. Zhu and J. Ma, "Image Processing Algorithm Design for Low-Light EBCMOS Devices Based on FPGA," 2023 3rd International Conference Information Engineering on and Electronic Computer

Communication (EIECC), Wuhan, China, 2023, pp. 1-6, doi: 10.1109/EIECC60864.2023.10456650.

[2] I. Chiuchisan, "An approach to the Verilog based system for medical image enhancement," 2022 E-Health and Bioengineering Conference (EHB), Iasi, Romania, 2022, pp. 1-4, doi: 10.1109/EHB.2022.7391461.

[3] V. S. Seleznev, E. O. Antonova, A. V. Iluhin, R. A. Gematudinov and L. Y. Isaeva, "Implementation on Sobel Field-Programmable Gate Array Detector for Identification of Vehicles," 2021 Intelligent Technologies and Electronic Devices in Vehicle and Road Transport Complex (TIRVED), Moscow, Russia, 2021, 10.1109/TIRVED53476.2021.9639132. pp.

[4] C. T. Hai, O. C. Pun and T. W. Haw, "Accelerating video and image processing design for FPGA using HDL coder and Simulink," 2015 IEEE Conference on Sustainable Utilization And Development In Engineering and Technology (CSUDET), Selangor, Malaysia, 2021, pp. 1-5, doi: 10.1109/CSUDET.2021.7446221.

[5] D. A. Padilla et al., "Implementation of eye gaze tracking technique on FPGA-based on-screen keyboard system using Verilog and MATLAB," TENCON 2020 - 2020 IEEE Region 10 Conference, Penang, Malaysia, 2020, pp. 2771 2776, doi: 10.1109/TENCON.2017.8228333.

[6] M. Sreenivasulu and T. Meenpal, "Efficient Hardware Implementation of 2D Convolution on FPGA for Image Processing Application," 2019 IEEE ICECCT, Coimbatore, India, 2019, pp. 1-5, doi: 10.1109/ICECCT.2019.8869347.

[7] Sankaranarayanan, V. et al. (2018). Real-time edge detection and noise reduction using hybrid filters on FPGA. Journal of Real-Time Image Processing, 17(5), 1531-1545.

[8] Aparna, S. et al. (2017). FPGA-based real-time edge detection. International Journal of Computer Science and Engineering, 5(3), 113–120.

[9] Zhou, S. et al. (2015). FPGA-based Gaussian filter for real-time image processing. IEEE Transactions on Circuits and Systems for Video Technology, 25(3), 428–436.

[10] Chiuchisan, "An approach to the Verilog based system for medical image enhancement," 2015 E-Health and Bioengineering Conference (EHB), Iasi, Romania, 2015, pp. 1-4, doi: 10.1109/EHB.2015.7391461.

