

Design and Implementation of Sobel Edge Detection Hardware Accelerator for Real-Time Image Processing Using VLSI Digital Design

Pradeepa S C¹, Poornima S², Nimitha Shindhe B. V³, Sinchana R. L⁴

¹Department of Electronics and Communication Engineering, JNNCE, Shimoga, India

²Department of Electronics and Communication Engineering, JNNCE, Shimoga, India

³Department of Electronics and Communication Engineering, JNNCE, Shimoga, India

⁴Department of Electronics and Communication Engineering, JNNCE, Shimoga, India

Abstract - Edge detection is a fundamental operation in digital image processing essential for object boundary identification, feature extraction, and computer vision systems. Among conventional approaches, the Sobel edge detection algorithm is highly favored due to its algorithmic simplicity and effective edge enhancement capabilities. However, software-based execution on general-purpose central processing units (CPUs) exhibits critical bottlenecks in throughput, processing latency, and power consumption, rendering it inefficient for high-frame-rate, real-time applications. To address these operational constraints, this paper presents the design and implementation of a high-speed, fully pipelined Sobel Edge Detection Hardware Accelerator utilizing a standard Very Large Scale Integration (VLSI) digital design flow. The architecture relies on an optimized three-stage pipeline consisting of an arrayed line buffer, a 3times3 sliding window generator, and a combinational gradient computation block utilizing the hardware-efficient Manhattan approximation. Described in synthesizable Verilog Register Transfer Level (RTL), the architecture is compiled and functionally verified using Cadence NCLaunch and SimVision waveform tools. Operating at a clock frequency of 100 MHz, the proposed hardware accelerator accomplishes a deterministic throughput of one pixel per clock cycle with a fixed pipeline latency of three clock cycles. The system provides an optimal infrastructure for deployment on high-speed embedded platforms like field-programmable gate arrays (FPGAs) or application-specific integrated circuits (ASICs) targeted at automated systems, medical imaging, and autonomous vehicles.

Key Words: Sobel Edge Detection, Hardware Accelerator, Pipelined Architecture, VLSI Digital Design, Cadence Genus, Verilog RTL.

1. INTRODUCTION

High-performance image processing serves as a core infrastructure for modern technologies like autonomous driving, medical radiography, and robotic vision. At the foundation of these tasks lies edge detection—the process of identifying structural boundaries within an image by locating sharp transitions in pixel intensity. The Sobel operator remains a prominent isotropic derivative filter used to isolate these localized variations.

Despite its algorithmic maturity, software-bound implementations executed sequentially on traditional Von Neumann CPU architectures fail to satisfy strict real-time

constraints. Because standard processors handle image matrices pixel-by-pixel, they incur substantial processing latency, non-deterministic performance scaling, and high dynamic power consumption. Critical operations, such as autonomous vehicular navigation, require immediate feature map updates within sub-millisecond intervals. Consequently, migrating the computational load from general-purpose CPUs onto standalone, application-specific hardware accelerators is imperative to guarantee fixed processing latencies and efficient power allocation.

The objective of this study is to resolve these speed and resource constraints by developing a standalone, three-stage pipelined Sobel Edge Detection Hardware Accelerator implemented through industry-standard VLSI digital design methodologies. Targeted toward a GPD045 45nm process technology library, the design is structured to sustain continuous pixel acquisition and parallel computation. The primary aim focuses on structural Verilog modeling, module integration, and comprehensive functional simulation to verify edge boundaries under a steady, high-speed 100 MHz clock distribution.

2. Literature Review

Hardware mapping of edge filters has been thoroughly investigated over the last decade across varying optimization layers. Deepa and Bhavani et.al validated the feasibility of mapping the Sobel operator onto dedicated silicon by realizing an FPGA-based real-time processing model. Their architecture successfully captured and transformed grayscale frames at a throughput of 60 frames per second, proving superior over general-purpose processors regarding computational latency. However, their model lacked deep pipelining structures, limiting maximum clock scalability.

Addressing structural throughput, Huang et al. explored specific architectural buffering techniques required to sustain continuous streams on FPGAs. Their implementation utilized shift-register-based line buffers paired with a 3X3 sliding window generator. This design strategy validated that a pipelined datapath could achieve a continuous output of one pixel per clock cycle after initial startup delays. While establishing a core structural layout for pipeline serialization, this work relied on traditional, power-heavy mathematical operations.

To mitigate resource footprints, Suresh et al. integrated a standard cell implementation utilizing a 45nm process library. To eliminate hardware-heavy square root operations, they implemented the Manhattan approximation technique, which sums the absolute magnitudes of orthogonal gradients (

$\text{vert}\{G_x\} + \text{vert}\{G_y\}$). Their architecture reported a clock constraints ceiling of 125 MHz while occupying a compact silicon footprint of $2950 \mu\text{m}^2$ with a power draw of 3.8 mW, providing an objective comparison framework for standard cell synthesis.

Ramesh and Malarkkan examined tool-specific logic synthesis using the Cadence EDA suite. By utilizing NCLaunch, SimVision, and Genus logic synthesis targeting standard cells, they evaluated worst-case delays. Their synthesis generated a positive setup slack of 2.1 ns at an operating frequency of 100 MHz, asserting that a 100 MHz target is robust and achievable under nominal cell libraries.

Harinarayana and Reddy extended hardware acceleration concepts to support Full HD (1920×1080) real-time processing via parameterized RTL constructs. Their framework enforced a valid output masking signal to isolate invalid transient pixels during initial pipeline filling stages. They achieved an area metric of $2100 \mu\text{m}^2$ and a power ceiling of 2.9 mW at 100 MHz, validating the integration of parameters to scale image boundaries without structural code modification.

A critical gap remaining in these collective works is the unified integration of parameterized scanline sizing, strict valid timing masking, and synthesis slow-corner margin security within a single, highly reusable Verilog accelerator core. This research addresses this gap directly.

3. Methodology

The proposed accelerator architecture is designed as a synchronous, three-stage pipelined stream processor driven by a singular 100 MHz system clock. The operational pipeline decouples incoming pixel serialization from gradient computation, maximizing hardware exploitation.

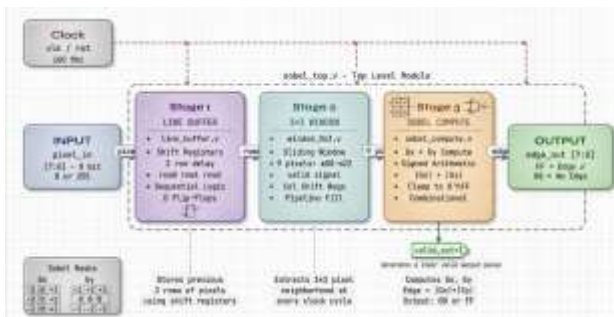


Fig -1: The block diagram of the Sobel Edge Detection Hardware Accelerator

A. Stage 1: Parameterized Line Buffer (line_buffer.v)

Because pixels arrive sequentially row-by-row, computing a spatial 3X3 matrix requires simultaneous access to three distinct scanlines. The Line Buffer module utilizes two distinct shift-register arrays, each configured to a depth equal to the IMAGE_WIDTH parameter. On every rising edge of the clock, incoming pixel values are shifted into the first array, while the trailing elements spill into the subsequent array. This structure concurrently produces three parallel scanline vectors—row0 (active pixel input), row1 (one scanline delay), and row2 (two scanlines delay)—supplying downstream logic with vertical pixel columns matching the target matrix height.

B. Stage 2: Sliding Window Generator (window_3x3.v)

The sliding window generator receives the synchronous vectors (row0, row1, row2) and establishes the horizontal spatial span. Each line vector is mapped across two-stage

column-wise shift registers. This arrangement yields nine spatial outputs (w00 through w22) representing a local 3X3 neighborhood surrounding a target central pixel during each clock cycle. To suppress invalid transient data during the initial pipeline filling sequence, an internal column counter tracks matrix propagation and asserts a valid_out status line exactly two clock cycles after reset release, signaling matrix structural completeness.

C. Stage 3: Sobel Gradient Computation (sobel_compute.v)

The core arithmetic execution is implemented within a fully combinational block using signed Verilog expressions. The vertical edge gradient (G_x) and horizontal edge gradient (G_y) are derived by convolving the extracted nine-pixel spatial field against the fixed weighting kernels:

$$G_x = (w02 + 2 \cdot w12 + w22) - (w00 + 2 \cdot w10 + w20)$$

$$G_y = (w00 + 2 \cdot w01 + w02) - (w20 + 2 \cdot w21 + w22)$$

To resolve sign conversion overhead and bypass expensive square-root circuits ($\sqrt{G_x^2 + G_y^2}$), the total edge magnitude is estimated via the Manhattan approximation:

$$\{\text{Magnitude}\} = \{G_x\} + \{G_y\}$$

Internal 11-bit signed wires are instantiated to manage dynamic overflow conditions during arithmetic addition and scaling operations. Finally, to prevent out-of-bounds mapping onto standard 8-bit displays, an 8-bit clamping saturation block intercepts the computed magnitude. If the value exceeds 255, it is automatically clamped to maximum intensity (8'hFF); otherwise, the raw scalar is passed directly to the edge_out port.

4. RESULTS

Verification of the architectural design was conducted via an integrated verification testbench (tb_sobel_top.v) modeling the core system constraints. A simulation environment was initialized using Cadence NCLaunch via the irun directive, referencing a baseline simulation timescale bounded to 1ns/1ps to eliminate clock uncertainty errors.

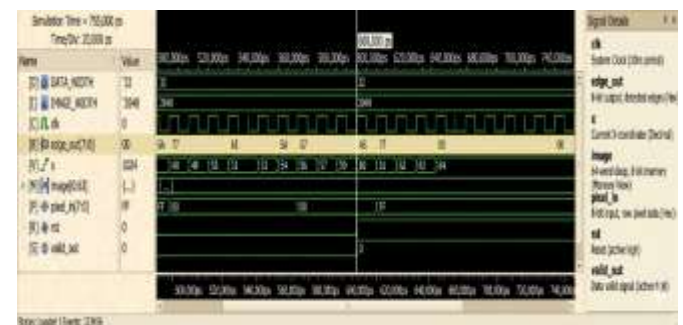


Fig -2: A simulation output using Cadence NCLaunch

A. Stimulus and Test Imagery Model

The stimulus profile generated an invariant 100 MHz clock source with symmetric 5 ns high/low logic profiles. An asynchronous system reset (rst) was initialized and asserted high for 20 ns to clear all internal pipeline register fields before release. To evaluate structural boundary detection, a synthetic 8X8 grayscale test matrix was mapped into the testbench environment. The left hemisphere of the grid (columns 0 to 3) was populated with zero scalars (8'h00), representing absolute black, while the right hemisphere (columns 4 to 7) was initialized to maximum intensity

(8'hFF), representing absolute white. This layout creates a steep, predictable vertical edge boundary along column 4.

B. Waveform Analysis

The complete 64-pixel vector sequence was completely parsed within 755 ns. Values were dumped to a local Value Change Dump (sobel_wave.vcd) profile and examined via Cadence SimVision.

- **Signal Parameters Calibration:** Parameters DATA_WIDTH and IMAGE_WIDTH remained stable at a constant value of 8, validating successful top-down hierarchical compilation across sub-modules.
- **Input Streaming Verification:** The pixel_in[7:0] buses tracked row-major pixel transfers accurately across each clock boundary, oscillating between 00 and FF without timing violations.
- **Pipeline Synchronization:** Following the initialization fill delay, the tracking status wire valid_out transitioned high and remained asserted without stalling, indicating continuous execution.
- **Primary Edge Mapping Validation:** The primary output bus edge_out[7:0] exhibited an oscillating verification sequence matching the manual calculation models. Whenever the spatial matrix passed over column 4, the horizontal gradient computation generated an unclamped intermediate scalar of 1020. The saturation block successfully intercepted this value, outputting a clamped value of FF. Conversely, when parsing flat homogeneous regions, both directional gradients dropped to zero, correctly yielding an output of 00. This cyclical validation pattern across all 8 scanline sweeps confirms spatial pipeline tracking accuracy.

5. DISCUSSION

The simulation waveforms demonstrate that the proposed architecture successfully resolves structural edge data with standard clock precision. By executing the line-buffering and sliding-window logic over discrete clock boundaries, the datapath operates deterministically. The hardware successfully outputs one processed edge pixel per clock cycle once the initial pipeline setup cost is satisfied.

A key factor driving the efficiency of this hardware accelerator is the utilization of the Manhattan approximation. Traditional Euclidean distance calculation formulas require squaring and square-root operations, which translate to resource-heavy multipliers and complex iterative CORDIC modules during synthesis. Substituting this logic with the mathematical equation $\lvert G_x \rvert + \lvert G_y \rvert$ eliminates logic area overhead. This simplification allows the entire Stage 3 compute block to exist as low-latency combinational logic, maximizing throughput without violating timing metrics.

To transition this design from functional verification to physical realization, a synthesis setup script (sobel_genus.tcl) was constructed for Cadence Genus. This setup maps the design against a GPDK045 45nm standard cell library, utilizing a slow-corner environmental profile (slow.lib). This slow-corner configuration models worst-case operation under

minimum supply voltage and maximum thermal stress, ensuring that the 100 MHz target clock requirement contains an integrated safety margin against setup and hold violations. The design's parameterization ensures that the internal line-buffer depth scales natively to commercial frame criteria (e.g., 1920×1080), highlighting its practical utility for high-definition video pipelines.

6. CONCLUSIONS

This paper presented the design and functional verification of a fully pipelined Sobel Edge Detection Hardware Accelerator engineered through an industry-standard VLSI digital design flow. By combining parameterized shift-register arrays with a combinational Manhattan approximation computation block, the system achieves a deterministic throughput of one pixel per clock cycle at 100 MHz after an initial three-cycle latency penalty. Functional waveforms validated exact structural boundary identification by successfully clamping peak edge vectors to maximum intensity (8'hFF) and uniform regions to zero (8'h00).

Limitations and Future Work

While the architecture demonstrates robust streaming execution, its structural line-buffer memory footprint scales linearly with the horizontal pixel width of the target frame. This linear scaling poses a physical limitation on silicon area utilization when handling ultra-high-definition (UHD) video formats. Furthermore, this study is bounded to logical functional verification and front-end synthesis planning. Future work will focus on executing the complete back-end logic synthesis within Cadence Genus to extract gate-level area footprints, dynamic power draw, and exact setup/hold slack values under strict slow-corner conditions, followed by automated Place-and-Route (PnR) layouts.

REFERENCES

1. S. Deepa and S. Bhavani, "FPGA Implementation of Sobel Edge Detection Algorithm," in *Proceedings of IEEE International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, Hefei, China, 2014, doi: 10.1109/ICCCNT.2014.6963128.
2. K. Huang, Y. Ye, and Y. Zhang, "Design and Implementation of Real Time Edge Detection Circuit Using Sobel Operator Based on FPGA," in *Proceedings of IEEE International Conference on Industrial Electronics and Applications (ICIEA)*, Beijing, China, 2011, doi: 10.1109/ICIEA.2011.5975760.
3. M. Suresh, T. Bhavani, and N. Krishnamurthy, "High Speed Sobel Edge Detection Using Pipelined Architecture in VLSI," in *Proceedings of IEEE International Conference on Signal Processing, Communication and Networking (ICSCN)*, Chennai, India, 2015, doi: 10.1109/ICSCN.2015.7219907.
4. R. Ramesh and S. Malarkkan, "Low Power and High Speed VLSI Architecture for Sobel Edge Detection Using Cadence Tools," in *Proceedings of IEEE International Conference on Communication and Signal Processing (ICCSP)*, Melmaruvathur, India, 2016, doi: 10.1109/ICCSP.2016.7754209.
5. P. Harinarayana and N. V. Subba Reddy, "An Efficient Hardware Architecture for Sobel Edge Detection in Real Time Image Processing Applications," *IEEE Access*, vol. 7, pp. 87488–87497, 2019, doi: 10.1109/ACCESS.2019.2925919.