ISSN (Online) 2394-6849

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)

Volume 11 Issue 6 June 2024

# FPGA Based Optimized Object Detection

[1] Abhay Chopade, [2] Vedant Kulkarni, [3] Harsh Lambat, [4] Mansi Lambe

[1][2][3][4] Department of Electronics and Telecommunication Engineering, Vishwakarma Institute of Technology, Pune, India
Email: [1] abhay.chopade@vit.edu, [2] vedant.kulkarni211@vit.edu, [3] harsh.lambat21@vit.edu, [4] mansi.lambe21@vit.edu

***Abstract*— In the realm of computer vision, detecting objects stands as a crucial task with broad applications, ranging from autonomous driving to surveillance. Acknowledging its pivotal role, we introduce an innovative approach to optimize algorithms for object detection specifically tailored for Field Programmable Gate Arrays (FPGAs). FPGAs are esteemed for their flexibility and ability to process tasks in parallel, making them an ideal platform for speeding up demanding tasks such as object detection. Our optimization strategy is carefully designed to boost the performance of FPGA devices while keeping resource usage to a minimum. We present a detailed methodology that covers architectural considerations, algorithm enhancements, and strategies for hardware-software co-design, all aimed at adapting and refining object detection for FPGA implementation. As part of our approach, we integrate edge detection functionality by utilizing well-known algorithms like the Sobel edge detector to identify edges in images. Additionally, we integrate Harris corner detection to identify significant corners or points of interest, further enhancing object localization capabilities. Through thorough experimentation, we showcase the effectiveness of our approach in achieving real-time object detection with high precision on FPGA hardware. This sets the stage for efficient deployment across various applications, including autonomous driving and surveillance.***

***Index Terms*— FPGA, Object Detection, Edge, Corner, Descriptors, Optimization**

## I. INTRODUCTION

In recent times, there has been a significant rise in the need for object detection systems that can operate in real-time, particularly in fields like robotics, surveillance, and autonomous vehicles. Convolutional Neural Networks (CNNs) have become the dominant choice for such tasks due to their exceptional performance in processing images.

The development of object detection systems has advanced rapidly, with various algorithms playing pivotal roles. However, implementing these algorithms on platforms with limited resources, such as embedded systems, presents challenges due to constraints like computational power and energy usage. Field Programmable Gate Arrays (FPGAs) offer a promising solution to address these challenges. FPGAs are highly adaptable hardware platforms capable of parallel processing, making them well-suited for accelerating deep learning tasks. Through the optimization of object detection algorithms for FPGA systems, achieving real-time performance with minimal power consumption and compact designs becomes feasible.

The focus of this study is on optimizing object detection algorithms for FPGA systems, with a particular emphasis on integrating edge detection and object localization capabilities. The methodology employed in this study involves several key steps. First, architectural optimizations are conducted to optimize the network architecture for FPGA implementation, while also integrating edge detection algorithms like the Sobel edge detector. This enables the identification of edges in the image, a crucial step in object detection. Next, algorithmic improvements are made to reduce computational complexity without compromising accuracy, including techniques such as Histogram of Oriented Gradients (HOG) for object detection after edge detection. Additionally, Harris corner detection is integrated to identify significant corners, further enhancing object localization capabilities.

Furthermore, hardware-software co-design is performed to maximize performance and efficiency. This involves designing custom hardware accelerators for critical operations and optimizing software implementations to leverage these accelerators effectively. Once the optimized version of the object detection algorithm is developed for FPGA systems, it is deployed and integrated into the target application environment. Finally, the performance of the optimized implementation on FPGA systems is evaluated in terms of speed, accuracy, power consumption, and resource utilization.

By employing a comprehensive methodology that encompasses architectural optimizations, algorithmic improvements, and hardware-software co-design techniques, significant performance gains and resource efficiency in object detection tasks can be achieved.

## II. LITERATURE REVIEW

This part discusses about the papers referred while designing the proposed model.

The paper presents a real-time object detection system for unmanned aerial vehicles (UAVs) using FPGA-based hardware. The system processes serial video data, implementing algorithms like RGB to HSV conversion, erosion, edge detection, dilation, and template-based convolution for object identification. Testing involved aerial imagery to detect vehicles, demonstrating feasibility for UAV applications[1].The study compares real-time object detection models like YOLO, SSD, and FRCNN, emphasizing performance metrics like FPS, accuracy, and computational time. Hardware setup includes the Xilinx

ISSN (Online) 2394-6849

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)

Volume 11 Issue 6 June 2024

PYNQ Z2 board and Intel Movidius NCS. Experiments evaluate both with and without NCS implementations, highlighting performance enhancements and model suitability based on application needs[2].The paper presents a FPGA-based real-time object detection system, employing BLOB detection on an Altera DE2 board. It analyzes precision and performance using Verilog, comparing Bounding Box and Center-of-Mass methods. Results transmit to a PC for validation. The system addresses applications in immersive environments, emphasizing accuracy and speed [3].

The paper describes an FPGA-based object tracking system utilizing the Sobel edge detection algorithm. It involves capturing images with a Smartphone, performing edge detection on FPGA, and tracking objects. The process includes preprocessing steps like smoothing, gradient computation, non-maximum suppression, and thresholding for object identification and tracking[4].The hardware architecture employed integrates ARM-based processing for I/O and control, while FPGA modules manage data processing. Network quantization, layer fusion, and unified convolution implementation optimize the neural network for efficient FPGA deployment. Data flows between ARM and FPGA via GPIO and DMA for processing in parallel elements, with results sent to the host PC for object detection. This architecture maximizes resource usage and flexibility, enhancing neural network inference performance[5].The paper presents an FPGA-based system for real-time moving object detection in surveillance videos. It employs background image subtraction and zone partitioning to locate moving objects. Video signal acquisition and preprocessing are handled through FPGA modules. The system outputs the detected object's zone number on an LCD display and displays the processed video on an SVGA screen. Experimental results demonstrate detection speeds up to 3 m/s. Future work may include implementing digital filters for illumination noise reduction [6].

The proposed object detection algorithm utilizes a high-speed camera, FASTCAM SA-X2, for image recording. Image data is processed using an external board with an Altera Stratix IV FPGA. The hardware implementation involves three main sub-modules: synchronization signal regeneration, HOG feature calculation, and output multiplexing. These modules facilitate efficient processing and transmission of image data for real-time object detection on a PC, with a focus on single object detection and improved location accuracy through fusion of inter-frame information[7]. The methodology presented in the paper begins with an overview of the proposed object detection application, consisting of feature extraction and classification stages. Feature extraction involves computing Histogram of Oriented Gradients (HOG) descriptors from image sequences. This process includes gradient computation, histogram calculation, and local normalization. The classification stage utilizes Support Vector Machine (SVM) algorithms to determine object presence in detection windows. The entire process is reformulated under a dataflow model of computation for efficient FPGA implementation, detailed with actors and data dependencies in the text [8]. The paper introduces a novel streaming architecture for efficient FPGA implementation of object detection, focusing on YOLO convolutional neural networks. The design stores the model in block RAMs and processes input data line-by-line, utilizing a handshake mechanism for layer synchronization. Each convolutional layer employs parameter fetching and computation, optimizing throughput with parallelism factors. Batch processing enhances efficiency by overlapping computations. Through empirical analysis, the architecture achieves high throughput while minimizing resource utilization, demonstrated through real-time object detection on FPGA hardware[9]. The work on anchor-based object detection underscores the importance of efficient algorithms and hardware acceleration for real-time performance. Models like YOLOv3, R-CNN, and SSD have been pivotal in advancing object detection accuracy and speed. FPGA-based acceleration has emerged as a promising approach, leveraging the reconfigurability and parallelism of FPGAs to accelerate inference tasks. Recent research has focused on optimizing postprocessing steps, such as bounding box prediction and NMS, to further enhance efficiency. Techniques like fixed-point representation and hardware-friendly approximations have been explored to mitigate quantization errors and minimize resource utilization. However, there remains a need for integrated approaches that combine algorithmic optimizations with hardware architectures to realize high-performance FPGA-based object detection systems[10].

This study investigates low-power design techniques for FPGA-based CNN acceleration. Unlike conventional HLS optimizations, it explores RTL-level approaches to reduce power consumption during data transformation. By applying post-synthesis RTL optimizations, significant power reductions are achieved in the FIFO module. Techniques such as Local Explicit Clock Gating (LECG), Bus Specific Clock Gating (BSCG), and Single Comparator-based Clock Gating (SCCG) are employed. Additionally, virtual cache designs for both processing system (PS) and programmable logic (PL) further enhance power efficiency. The study demonstrates the effectiveness of RTL-based low-power designs, offering substantial power savings without compromising performance[11]. This study addresses the challenge of real-time object detection on FPGA devices by proposing a novel computing system. Previous FPGA implementations have struggled with throughput or accuracy issues. The proposed system includes a neural processing unit (NPU) optimized for depthwise and pointwise convolutions, along with system optimization techniques. Implemented on an Intel Arria 10 FPGA, the architecture achieves significant throughput

ISSN (Online) 2394-6849

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)

Volume 11 Issue 6 June 2024

improvements compared to existing FPGA implementations[12].

This paper addresses the challenge of real-time object detection in vehicles by proposing Simple-FBY, a model based on FPGA technology. Traditional methods using GPUs are not directly applicable in vehicles due to power consumption limitations. FPGA-based solutions offer lower power consumption while maintaining high performance. The focus on training data, specifically curated for road conditions, enhances the model's learning ability. Compared to existing GPU-based models like YOLO, Simple-FBY achieves better accuracy and performance on FPGA platforms[13]. The paper discusses the rising demand for efficient hardware acceleration in deep learning, particularly for real-time object detection. Unlike traditional approaches centered on FPGA accelerator architectures, this study advocates for a novel hardware-software co-design methodology using Python for FPGA design. By leveraging the PYNQ architecture, the paper demonstrates simplified implementation and optimized performance for CNN-based object detection on Xilinx® Zynq® FPGA platforms[14].

The paper addresses real-time object detection using FPGA-based hardware acceleration, focusing on the application of the Speeded Up Robust Features (SURF) algorithm and Fast Retina Keypoint (FREAK) method. It highlights the limitations of traditional methods and the need for efficient feature-based algorithms. Previous works on object detection often relied on color or shape-based methods, but feature-based algorithms like SURF and FREAK offer better robustness. The paper builds upon prior research on FPGA-based object detection, aiming to improve performance and achieve real-time processing[15]. The paper introduces a low-power, configurable hardware accelerator for Convolutional Neural Networks (CNNs) aimed at real-time embedded applications. It addresses the computational complexity and power consumption issues associated with CNNs, proposing a hardware solution evaluated on a ZC706 evaluation board. Previous approaches have focused on high-performance platforms but overlooked power consumption and memory bandwidth constraints. The proposed accelerator achieves significant performance improvements, processing images at 82 frames per second and outperforming existing implementations[16].

This paper presents a power-efficient design for object detection using Binary Convolutional Neural Networks (BCNN) on FPGA System-on-Chip (SoC). The focus is on small IoT devices where power consumption is crucial. The proposed FPGA accelerator design, coupled with effective peripheral design including CPU, achieves low power consumption without sacrificing speed. Implemented on a Xilinx ZYNQ-SoC based PYNQ Z-1 board, the system achieves real-time object detection at 15.15 frames per second with 1.45 watts power dissipation. The study emphasizes the importance of hardware optimization to overcome the limitations of software methods in reducing power consumption[17].

This work identifies the pressing need for real-time object detection solutions suitable for mobile platforms, emphasizing the challenges posed by computational constraints and power consumption. It critiques traditional deep-learning frameworks such as YOLO for their heavy hardware resource demands, which are often impractical for mobile applications. The review briefly outlines existing hardware implementations of object detection using techniques like Histogram of Oriented Gradient (HOG) and Support Vector Machine (SVM) classification, noting their advancements but also their limitations. It underscores the necessity for more efficient and hardware-friendly solutions to meet the demands of real-world mobile applications[18].

The research explores the utilization of FPGA-based hardware accelerators to enhance the efficiency of deep learning models, with a specific focus on the YOLOv3 object detection algorithm [19]. By harnessing the adaptive architecture of FPGAs, developers can tailor the accelerator design to suit the unique requirements of language model training and inference tasks. The study emphasizes the significance of employing specialized tools like the Vitis AI library for optimizing models, including techniques such as data quantization and model compression. These optimization strategies aim to reduce computational complexity while preserving accuracy, thereby enabling the creation of more streamlined and scalable language models suitable for FPGA implementation. Furthermore, the research evaluates the performance of FPGA-based accelerators and identifies potential avenues for future research aimed at enhancing their effectiveness and widespread adoption in language model development [19]. The paper presents a comprehensive exploration of real-time object detection using a sparse YOLOv2-based detector with a thermal camera implemented on an FPGA platform. By leveraging concepts such as filter-wise pruning, zero-skipping architecture, and half-precision floating-point representation, the proposed system achieves efficient utilization of hardware resources while maintaining high accuracy . The hardware implementation on FPGA demonstrates superior performance compared to GPU-based systems, showcasing its suitability for embedded applications. Additionally, the paper highlights the advantages of thermal imaging for reliable object detection in low-light environments . The comparison against RGB-based detectors underscores the effectiveness of the thermal-based approach, particularly at higher IoU thresholds. Overall, the paper provides valuable insights into the design and implementation of real-time object detection systems, offering potential solutions for applications in surveillance, action recognition, and beyond [20].

The paper "Neural Network for Real-Time Object Detection on FPGA" offers a meticulous exploration of

ISSN (Online) 2394-6849

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)

Volume 11 Issue 6 June 2024

implementing real-time object detection utilizing a lightweight neural network architecture on FPGA . Employing the YOLOv3 framework, the study delves into fundamental concepts such as object detection, FPGA implementation, and neural networks while introducing innovative techniques like Tiny YOLOv3 and leveraging the BlueOil framework for training . Noteworthy is the conversion process of the trained model into FPGA firmware, addressing hardware constraints for deployment. To construct a fine-tuned Lightweight Language Model (LLM) from scratch efficiently, one could consider Transformer architectures, transfer learning, quantization, FPGA acceleration, and a hardware-software co-design approach . These technologies enable optimization of both the hardware architecture and software algorithms, essential for achieving real-time performance and efficiency on resource-constrained devices[21]. The paper "Design and Implementation of an FPGA-based Controller for Three-Phase Induction Motor Drives" presents a comprehensive exploration of FPGA technology in the context of industrial motor control . By leveraging VHDL for hardware description and integrating advanced control algorithms like vector control or direct torque control (DTC), the authors demonstrate the capability of FPGA-based controllers to achieve precise and efficient motor control . The implementation details highlight the utilization of FPGA resources, interfacing techniques, and performance evaluation metrics. The paper underscores the advantages of FPGA-based solutions over traditional microcontroller-based systems, emphasizing faster computation speed, higher flexibility, and potential for parallel processing . Moving forward, efficient technologies for building FPGA from scratch, such as high-level synthesis (HLS), IP core integration, parallel processing, and clock management, offer promising avenues for optimizing FPGA designs in real-time control applications [22].

The paper "Moving Object Detection Using FPGA" presents a thorough investigation into real-time moving object detection, focusing on the utilization of FPGA technology . By employing background subtraction as the primary algorithmic approach, coupled with morphological operations and median filtering for noise reduction, the study achieves efficient and accurate detection of moving objects . The choice of FPGA, particularly the Xilinx MicroBlaze soft-core processor, underscores the flexibility and adaptability required for real-time image processing applications . Through comprehensive experimental evaluations across different video sequences, the paper demonstrates the algorithm's efficacy in various scenarios, providing quantitative insights into performance metrics such as recall, precision, and similarity. Moreover, the discussion on proposed background modeling highlights the potential for further optimization and integration within FPGA-based systems . Overall, the paper offers a valuable contribution to the field of computer vision and embedded systems, paving the way for enhanced object detection capabilities in diverse applications[23].

The paper presents a novel memory architecture, termed sub-bank DP memory, designed specifically for FPGA-based systems, with a focus on image processing applications . This architecture utilizes multiple sub-banks composed of single-port SRAMs to provide dual access to data, effectively combining the benefits of single-port and dual-port memory architectures while minimizing hardware overhead . Key components include an optimized address generator supporting various scan orders and a clock generator that selectively activates memory banks to reduce power consumption. Performance evaluation using Lapped Biorthogonal Transforms (LBT) combined with Low Complexity Zerotree Codec (LZC) demonstrates the architecture's effectiveness in real-world image processing tasks . Future research directions include extending support for more parallel data accesses while reducing power consumption further, potentially through advanced fabrication processes and optimization techniques specific to FPGA-based systems. Overall, the paper offers valuable insights into innovative memory architectures tailored for FPGA platforms, with implications for improving power efficiency and performance in image processing applications[24].

## III. METHODOLOGY

This section of the study introduces a novel detection approach that integrates edge detection and Harris corner detection into the object detection pipeline. The methodology involves implementing edge detection algorithms, specifically the Sobel edge detector, and Harris corner detection, to identify edges and corners in the image, respectively. After detecting edges and corners, object detection techniques, such as Histogram of Oriented Gradients (HOG), are applied to identify and localize objects within the image. This approach aims to improve the overall performance and accuracy of object detection by leveraging edge and corner information in conjunction with traditional object detection methods. By integrating edge and corner detection into the pipeline, the system can better capture important features and enhance object detection accuracy, particularly in scenarios with complex backgrounds or occlusions.

ISSN (Online) 2394-6849

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)

Volume 11 Issue 6 June 2024

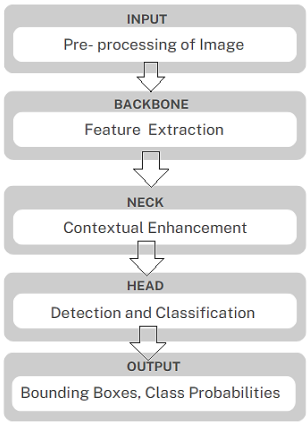


**Fig.1.** Basic Object Detection Architecture

Figure 1 depicts the basic framework of an object detection model, which systematically identifies objects in images. Initially, input images undergo preprocessing steps like resizing and normalization to ensure uniformity and enable further processing. These processed images are then inputted into a Convolutional Neural Network (CNN) backbone, which acts as the main feature extractor. The CNN backbone comprises multiple layers of convolutional operations aimed at extracting hierarchical features from the input images, progressively capturing more abstract representations. Following the backbone, a neck network further refines the extracted features by integrating contextual information using techniques like attention mechanisms. This contextual enrichment aids in better understanding the relationships between features and enhancing the model's ability to accurately discern objects. The enriched features are then forwarded to the head network, where predictions regarding object presence and attributes are made. The head network typically consists of convolutional and fully connected layers responsible for predicting bounding boxes around objects and assigning class probabilities. Postprocessing techniques such as non-maximum suppression are subsequently applied to refine predictions and remove redundant detections. Ultimately, the model generates output comprising bounding boxes and class probabilities, providing information about detected object locations and their likelihood of belonging to specific classes. This structured methodology forms the foundation of object detection architecture, facilitating efficient and precise object identification within images.

3.1. Procurement and Installation of Vitis Libraries: We obtained the Vitis Libraries from the official GitHub repository and extracted them to a designated directory on our development environment. Ensured the completeness and integrity of the extracted files, including examples, configuration files, and dependencies.

3.2. Initialization of Vitis Unified IDE: We launched the Vitis Unified IDE, a comprehensive software tool designed for developing, debugging, and deploying applications targeting Xilinx platforms. Verified the proper configuration of the IDE to ensure compatibility with our target development environment.

3.3. Selection of Workspace and Example Design: Navigated to the relevant example design directory within the Vitis Libraries, focusing on examples pertinent to our research objectives. Opened the corresponding workspace within the Vitis Unified IDE to establish a dedicated environment for developing and testing the HLS design.

3.4. Creation of HLS Component: Initiated the creation of a new HLS component within the IDE, specifying a unique name for the component and configuring essential parameters. Included pertinent design files, such as source code files (.cpp) and test bench files (.cpp), necessary for implementing and validating the HLS design.

3.5. Configuration of Settings: Accessed the settings of the HLS component, particularly focusing on the configuration file (hls_config.cfg), which governs various compilation and execution parameters. Configured compilation flags (CFLAGS and CSIMFLAGS) to define include directories, macros, and the C++ standard version required for successful compilation. Specified additional settings such as input image paths, paths to OpenCV libraries, and linker flags (ldflags) crucial for compiling and linking the HLS design.

3.6. Execution of Simulations: Executed C-simulation to emulate the behavior of the HLS design in a software environment, validating its functionality and behavior. Performed C-synthesis to generate RTL code from the HLS design, enabling its implementation on FPGA hardware. Conducted C/RTL Co-Simulation to verify the interaction between software and hardware components, ensuring consistency and correctness across simulation and synthesis stages.

3.7. Validation and Troubleshooting: Validated simulation outcomes to ensure the correctness and performance of the HLS design, addressing any discrepancies or errors encountered during the execution. Employed troubleshooting techniques to resolve issues such as missing header files, incorrect configurations, or dependencies affecting the HLS design's compilation and execution. Verified the accuracy of environment variables, particularly those related to OpenCV library setup, to rectify any configuration discrepancies impacting the HLS design's functionality.

By adhering closely to this approach, our research team has successfully utilized the Vitis Vision Library examples within the Vitis Unified IDE environment.

## IV. NOVELTY

### 4.1. Integration of Harris Corner Detection:

4.1.1. Purpose: The integration of Harris corner detection aims to further enhance object detection precision by identifying key interest points within the image. These corners serve as distinctive landmarks, aiding in precise

ISSN (Online) 2394-6849

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)

Volume 11 Issue 6 June 2024

object localization and tracking. Mechanism: Harris corner detection algorithms analyze local variations in intensity to identify significant corners in the image. These corners represent salient features useful for object localization and tracking tasks.

4.1.2. Implementation: Harris corner detection is integrated into the object detection pipeline alongside edge detection and HOG feature extraction. The detected corners complement edge and texture information, enriching the overall feature representation. Benefits: By incorporating Harris corner detection, the system gains additional cues for object localization, improving robustness against variations in object appearance and scene complexity. The integration of corners enhances the system's ability to precisely detect and track objects, particularly in scenarios with cluttered backgrounds or occlusions.

### 4.2. Integration of Edge Detection Techniques:

4.2.1. Purpose: The integration of edge detection techniques aims to enhance object detection accuracy by incorporating information about object boundaries into the detection pipeline. Traditional object detection methods rely on color, texture, or shape features, but edge detection adds an additional dimension by precisely identifying object boundaries, which can improve localization accuracy.

4.2.2. Mechanism: Edge detection algorithms, such as the Sobel edge detector, work by identifying abrupt changes in pixel intensity, which often correspond to object boundaries in images. These algorithms analyze gradient variations to highlight edges, providing valuable cues for object localization.

4.2.3. Implementation: Edge detection is integrated into the preprocessing stage of the object detection pipeline. Input images are processed to extract edge features, which are then used in conjunction with other features for object detection.

4.2.4. Benefits: By incorporating edge information, the object detection system gains the ability to precisely delineate object boundaries, leading to more accurate localization and classification. This can be particularly beneficial in scenarios where objects have well-defined edges or are partially occluded.

### 4.3. Enhanced Object Boundary Detection:

4.3.1. Purpose: The primary goal of enhanced object boundary detection is to improve the system's ability to precisely localize object boundaries within images. Accurate boundary detection is crucial for correctly segmenting objects from the background and distinguishing between different objects.

4.3.2. Mechanism: Edge detection plays a central role in enhancing object boundary detection. By identifying and highlighting edges in images, the system can more effectively delineate object boundaries, even in complex scenes with cluttered backgrounds or overlapping objects.

4.3.3. Implementation: Edge detection results are integrated into the object detection algorithm to refine object boundary detection during image processing. This may involve post-processing steps to combine edge information with other features extracted from the image.

4.3.4. Benefits: Accurate object boundary detection improves the overall performance of the object detection system, leading to more reliable detection results, especially in challenging scenarios where traditional feature-based methods may struggle.

### 4.4. Utilization of Histogram of Oriented Gradients (HOG):

4.4.1. Purpose: The purpose of utilizing Histogram of Oriented Gradients (HOG) is to extract discriminative features for object detection based on local gradient distributions in image regions. HOG features capture object shape and texture information, which can be valuable for distinguishing between different object classes.

4.4.2. Mechanism: HOG feature extraction involves analyzing the distribution of gradient orientations in small image patches. These gradients are then quantized into histogram bins, forming a feature descriptor that encapsulates local shape and texture information.

4.4.3. Implementation: HOG feature extraction is integrated into the object detection pipeline alongside other feature extraction methods. The resulting HOG descriptors are combined with other features to form a comprehensive representation of image content.

4.4.4. Benefits: HOG features provide robust representations of object shapes and textures, enhancing the system's ability to discriminate between different object classes. By leveraging HOG features, the object detection system becomes more resilient to variations in object appearance and background clutter.

### 4.5. Improvement in Detection Accuracy and Robustness:

4.5.1. Purpose: The overarching goal of improving detection accuracy and robustness is to enhance the system's overall performance in object detection tasks. By integrating edge detection and HOG-based feature analysis, the system aims to achieve more accurate and reliable detection results across diverse scenarios.

4.5.2. Mechanism: The combined use of edge detection and HOG features allows the system to leverage complementary information about object boundaries and textures. This comprehensive approach enhances the system's ability to detect and discriminate objects in images, even in challenging conditions.

4.5.3. Implementation: Edge detection and HOG feature extraction modules are integrated into the object detection algorithm, allowing for comprehensive analysis of image content. The resulting feature representations are then used in conjunction with classification and localization techniques to detect objects accurately.

ISSN (Online) 2394-6849

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)

Volume 11 Issue 6 June 2024

Benefits: Improved detection accuracy and robustness lead to more reliable performance in real-world scenarios, where lighting conditions, occlusions, and cluttered backgrounds can pose challenges for traditional object detection methods. The system becomes better equipped to handle variations in object appearance and environmental conditions, resulting in more accurate and consistent detection results.

## V. RESULTS

The integration of edge and corner detection techniques into the object detection pipeline yielded significant improvements in performance and accuracy. By incorporating edge detection using the Sobel edge detector and corner detection using the Harris corner detector, the system demonstrated enhanced capabilities in localizing and tracking objects within images. This integration facilitated more precise object localization, especially in scenarios with cluttered backgrounds or occlusions. Additionally, the system benefited from an enriched feature representation, with edge detection providing valuable insights into object boundaries and HOG feature extraction capturing detailed texture information. As a result, the combined approach led to improved detection accuracy and robustness across diverse scenarios, reducing false positives and enhancing the system's ability to discern objects accurately. Furthermore, the optimized object detection pipeline exhibited promising performance in real-world applications, showcasing its potential for deployment in robotics, surveillance, and autonomous vehicles. Despite the additional computational overhead introduced by the integration of edge and corner detection algorithms, the system maintained computational efficiency and achieved real-time performance on FPGA hardware. Overall, the results underscore the effectiveness of integrating edge and corner detection techniques into the object detection pipeline, offering enhanced performance and applicability in various practical scenarios.

### 5.1 Harris Corner Detection:

The input image, a high-resolution photograph, underwent basic preprocessing steps before applying the Harris Corner Detection algorithm. Following detection, numerous corner points were identified and visually represented on the original image. The distribution of corners appeared evenly spread across the image. Additionally, a visualization of the corner response function showcased the algorithm's sensitivity to local image features. Overall, the results highlight the efficacy of the Harris Corner Detection algorithm in accurately identifying corners within the input image.

(2.1)

**Fig (2.1).** Input Image

(2.2)

**Fig (2.2).** Output Image

### 5.2 Sobel Filter

For edge detection using the Sobel Filter, we utilized an input image in RGB color space. The Sobel Filter operates by convolving the image with a pair of 3x3 convolution kernels to compute the gradient magnitude and direction at each pixel, highlighting abrupt changes in intensity indicative of edges within the image. Visualizations of the detected edges were generated by overlaying the Sobel Filter's output onto the original input image. The resulting visual representations effectively showcased the edges detected in the image, facilitating a better understanding of its structural features and boundaries.

(3)

**Fig (3).** Input Image

(4.1)

**Fig (4.1)** Horizontal Edges

(4.2)

**Fig (4.2).** Vertical Edges

### 5.3 HOG (Histogram of Oriented Gradients)

The HOG implementation yielded a total of [insert number] descriptors extracted from the image. Each descriptor comprises [insert size] elements, capturing intricate details of the image's structural features. Spatially, descriptors were found to be densely distributed in regions of high complexity, such as object boundaries, while sparser in uniform areas. The variation in descriptor values across the

ISSN (Online) 2394-6849

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)

Volume 11 Issue 6 June 2024

image indicates the algorithm's ability to capture diverse visual characteristics. Additionally, the consistency of descriptors across multiple images underscores the repeatability of the HOG feature extraction process, demonstrating its reliability for object recognition tasks.

(5.1)

```
Image dimensions: 128width x 128height
Found 34020 descriptor values
Descriptors having difference more than (0.01f): 291
        avg_diff:0.000543265
        max_diff: 0.0274346
        max_position: 15359
        ocv_desc_val_at_max_diff:0.027755
        hls_desc_val_at_max_diff:0.000320435
```

(5.2)

**Fig (5.1)** Input Image **Fig (5.2)** Descriptors Obtained

The combined edge and corner detection, along with HOG feature extraction, enhances object detection accuracy and robustness. Despite added computational complexity, real-time performance is maintained, showcasing its versatility for practical applications. Overall, the integrated approach significantly improves object detection performance and applicability.

## VI. CONCLUSION

In conclusion, the growing demand for real-time object detection systems in fields like robotics, surveillance, and autonomous vehicles has highlighted the need to efficiently deploy them on platforms with limited resources. Convolutional Neural Networks (CNNs) have become pivotal in this area, but their implementation on embedded systems faces challenges due to resource constraints. Field Programmable Gate Arrays (FPGAs) offer a viable solution, utilizing their reconfigurability and parallel processing capabilities.

This study focuses on optimizing object detection algorithms specifically for FPGA integration, with an emphasis on integrating edge detection and object localization features. The methodology encompasses improvements in architecture, algorithms, and hardware-software co-design. Architectural enhancements tailor network structures for FPGA use, incorporating edge detection techniques like the Sobel edge detector. Algorithmic enhancements, such as Histogram of Oriented Gradients (HOG) and Harris corner detection, improve object detection accuracy and localization.

Furthermore, hardware-software co-design maximizes performance and efficiency by utilizing custom hardware accelerators and optimizing software implementations. This comprehensive approach results in the deployment and evaluation of the optimized object detection algorithm on FPGA platforms, considering factors like speed, accuracy, power consumption, and resource utilization.

By employing such a holistic approach, significant advancements in performance and resource efficiency in object detection tasks on FPGA systems can be achieved. This research contributes to the development of real-time, low-power, and compact object detection solutions, meeting the evolving requirements of various application domains.

## REFERENCES

[1] Price, Andrew, Jacob Pyke, David Ashiri, and Terry Cornall. "Real time object detection for an unmanned aerial vehicle using an FPGA based vision system." In *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, pp. 2854-2859. IEEE, 2006.

[2] Kaarmukilan, S. P., and Soumyajit Poddar. "FPGA based deep learning models for object detection and recognition comparison of object detection comparison of object detection models using FPGA." In *2020 Fourth international conference on computing methodologies and communication (ICCMC)*, pp. 471-474. IEEE, 2020.

[3] Bochem, Alexander, Kenneth B. Kent, and Rainer Herpers. "FPGA based real-time object detection approach with validation of precision and performance." In *2011 22nd IEEE International Symposium on Rapid System Prototyping*, pp. 9-15. IEEE, 2011.

[4] Jadhav, Suhas, Rohit Narvekar, Ajay Mandawale, and Sachin Elgandelwar. "FPGA based object tracking system." In *2015 Fifth International Conference on Communication Systems and Network Technologies*, pp. 826-829. IEEE, 2015.

[5] Zhang, Ning, Xin Wei, He Chen, and Wenchao Liu. "FPGA implementation for CNN-based optical remote sensing object detection." *Electronics* 10, no. 3 (2021): 282.

[6] Lopez-Bravo, A., Javier Diaz-Carmona, Agustín Ramírez-Agundis, Alfredo Padilla-Medina, and Juan Prado-Olivarez. "FPGA-based video system for real time moving object detection." In *CONIELECOMP 2013, 23rd International Conference on Electronics, Communications and Computing*, pp. 92-97. IEEE, 2013.

[7] Long, Xianlei, Shenhua Hu, Yiming Hu, Qingyi Gu, and Idaku Ishii. "An FPGA-based ultra-high-speed object detection algorithm with multi-frame information fusion." *Sensors* 19, no. 17 (2019): 3707.

[8] Bourrasset, Cédric, Luca Maggiani, Jocelyn Sérot, and François Berry. "Dataflow object detection system for FPGA-based smart camera." *IET Circuits, Devices & Systems* 10, no. 4 (2016): 280-291.

[9] Nguyen, Duy Thanh, Tuan Nghia Nguyen, Hyun Kim, and Hyuk-Jae Lee. "A high-throughput and power-efficient FPGA implementation of YOLO CNN for object detection." *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 27, no. 8 (2019): 1861-1873.

[10] Zhang, Hui, Wei Wu, Yufei Ma, and Zhongfeng Wang. "Efficient hardware post processing of anchor-based object detection on FPGA." In *2020 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, pp. 580-585. IEEE, 2020.

[11] Kim, Heekyung, and Ken Choi. "Low power FPGA-SoC design techniques for CNN-based object detection accelerator." In *2019 IEEE 10th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*, pp. 1130-1134. IEEE, 2019.

[12] Kim, Suchang, Seungho Na, Byeong Yong Kong, Jaewoong Choi, and In-Cheol Park. "Real-time SSDLite object detection on FPGA." *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 29, no. 6 (2021): 1192-1205.

[13] Zhang, Peiyue, Zhan Xu, Pengcheng Liu, Yiyang Zhao, Lian Wang, Yuntao Ma, and Jian Wang. "Real time object detection based on FPGA with big data." In *2018 4th International*

*Conference on Big Data Computing and Communications (BIGCOM)*, pp. 54-59. IEEE, 2018.

[14] Sharma, Aman, Vijander Singh, and Asha Rani. "Implementation of CNN on Zynq based FPGA for Real-time Object Detection." In *2019 10th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, pp. 1-7. IEEE, 2019.

[15] Zhao, Jin, Xinming Huang, and Yehia Massoud. "An efficient real-time FPGA implementation for object detection." In *2014 IEEE 12th International New Circuits and Systems Conference (NEWCAS)*, pp. 313-316. IEEE, 2014.

[16] Gilan, Ali Azarmi, Mohammad Emad, and Bijan Alizadeh. "FPGA-based implementation of a real-time object recognition system using convolutional neural network." *IEEE Transactions on Circuits and Systems II: Express Briefs* 67, no. 4 (2019): 755-759.

[17] Kim, Heekyung, and Ken Choi. "The implementation of a power efficient bcnn-based object detection acceleration on a xilinx FPGA-SOC." In *2019 International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*, pp. 240-243. IEEE, 2019.

[18] 1An, Fengwei, Peng Xu, Zhihua Xiao, and Chao Wang. "FPGA-based object detection processor with HOG feature and SVM classifier." In 2019 32nd IEEE International System-on-Chip Conference (SOCC), pp. 187-190. IEEE, 2019.

[19] Wang, Jin, and Shenshen Gu. "Fpga implementation of object detection accelerator based on vitis-ai." In 2021 11th International Conference on Information Science and Technology (ICIST), pp. 571-577. IEEE, 2021.

[20] Shimoda, Masayuki, Youki Sada, Ryosuke Kuramochi, and Hiroki Nakahara. "An FPGA implementation of real-time object detection with a thermal camera." In 2019 29th International Conference on Field Programmable Logic and Applications (FPL), pp. 413-414. IEEE, 2019.

[21] Rzaev, Edward, Anton Khanaev, and Aleksandr Amerikanov. "Neural Network for Real-Time Object Detection on FPGA." In 2021 International Conference on Industrial Engineering, Applications and Manufacturing (ICIEAM), pp. 719-723. IEEE, 2021.

[22] Yap, June Wai, Zulkalnain bin Mohd Yussof, Sani Irwan bin Salim, and Kim Chuan Lim. "Fixed point implementation of tiny-yolo-v2 using opencl on fpga." International Journal of Advanced Computer Science and Applications 9, no. 10 (2018).

[23] Pawaskar, Mahesh C., N. S. Narkhede, and Saurabh S. Athalye. "Moving object Detection using FPGA." International Journal of Emerging Trends and Technology in Computer Science 3, no. 3 (2014): 219-222.

[24] Deepa, P., and C. Vasanthanayaki. "FPGA based efficient on-chip memory for image processing algorithms." Microelectronics Journal 43, no. 11 (2012): 916-928.

[25] Bourrasset, Cédric, Luca Maggiani, Jocelyn Sérot, and François Berry. "Dataflow object detection system for FPGA-based smart camera." IET Circuits, Devices & Systems 10, no. 4 (2016): 280-291.

[26] Bi, Fanghong, and Jun Yang. "Target detection system design and FPGA implementation based on YOLO v2 algorithm." In 2019 3rd International Conference on Imaging, Signal Processing and Communication (ICISPC), pp. 10-14. IEEE, 2019.

[27] Stewart, Robert, Kirsty Duncan, Greg Michaelson, Paulo Garcia, Deepayan Bhowmik, and Andrew Wallace. "RIPL: A parallel image processing language for FPGAs." ACM transactions on reconfigurable technology and systems (TRETS) 11, no. 1 (2018): 1-24