Tuijin Jishu/Journal of Propulsion Technology
ISSN: 1001-4055
Vol. 45 No. 2 (2024)

Implementation of Object Detection Using FPGA

Ritik Kumar Verma[1] , Sahil Tiberawal[2], Satish Yadav[3], Sarvendra Singh[4]
Galgotia's College of Engineering and Technology Gr. Noida (U.P.)
Under The Mentorship Of
Prof. Alok Kumar

***Abstract*:** Robotics, autonomous driving, surveillance, and many more fields rely on object detection, a basic job in computer vision. Due to their low-latency speed and parallel processing capabilities, FPGA systems are attracting more and more interest in implementing object detection algorithms, which is important because real-time processing is becoming increasingly vital. This work provides a synopsis of the object detection on FPGA architecture, optimisation, and real-time implementation. The suggested method is picking an appropriate object detecting algorithm, like the well-known YOLO (You Only Look Once) or SSD (Single Shot MultiBox Detector), which are renowned for their speed and accuracy ratio. To achieve real-time speed, the algorithm is mapped onto an FPGA-based hardware architecture, which takes use of its reconfigurability and parallelism. An essential part of FPGA-based object detection is the design of the hardware architecture. Optimisation of data pathways, construction of efficient control logic, and splitting of the algorithm into hardware-friendly components are all part of this process. To achieve the goal of maximising throughput with minimal resource use, techniques including parallel processing, loop unrolling, and pipelining are utilised. In addition, optimising for FPGA requires tweaking the algorithm and hardware design to make the most of the target FPGA device's capabilities. Reducing latency and increasing throughput requires optimising data transfer, parallelism, and memory access patterns. Another important part of object detection systems that use FPGAs is their ability to integrate with various sensors or input streams. Acquiring input data for real-time processing necessitates integrating with various sensors, such as cameras and LiDAR devices. Thanks to their adaptability, FPGA platforms may be easily integrated into a wide range of application situations, thanks to their ability to interface with different sensors. To ensure the object detection system built on FPGA is accurate, fast, and resilient, it is validated and tested using common datasets and real-world scenarios. To guarantee the system achieves the targeted performance metrics, the real-time processing requirements are thoroughly assessed. The FPGA-based object detection system, once tested, can be placed in the intended setting as either a standalone device or a component of a bigger embedded system. Fixing bugs, improving performance, and adding new features all require regular maintenance and upgrades.

Keywords: *- FPGA, Object Detection, Computer Vision, Real-Time Processing, Hardware Optimization, Parallel Processing, Embedded Systems.*

Introduction

Autonomous vehicles, surveillance systems, robots, and many more fields rely on object detection, a basic job in computer vision. Intelligent decision-making in many fields relies on the capacity to precisely detect and localise things in real-time. Even if they work, traditional object detection methods can't always handle the intense demands of real-time processing, especially in complicated settings with moving scenes and numerous items to identify. So, to speed up object detection algorithms and get real-time performance, there has been an upsurge in the use of specialised hardware platforms like Field-Programmable Gate Arrays (FPGAs).

There are a number of benefits to using FPGAs instead of CPUs or GPUs when developing object detecting systems. First of all, field-programmable gate arrays (FPGAs) are very amenable to parallelization, which means that object identification techniques like convolutional neural networks (CNNs) may be implemented efficiently

on them. Because of their built-in parallelism, FPGAs may greatly improve processing speed by handling numerous data streams concurrently. The second advantage of field-programmable gate arrays (FPGAs) is that they may have their hardware architecture changed to fit the needs of different applications and object detection algorithms. Thanks to this adaptability, developers can maximise the system's performance by optimising the FPGA implementation for power efficiency, resource utilisation, and performance. When working with field-programmable gate arrays (FPGAs), one of the biggest obstacles is coming up with a memory-and CPU-efficient hardware design that can handle the detection method of choice. Many people like object detection algorithms like YOLO, SSD, and Faster R-CNN because they strike a good compromise between speed and accuracy. Convolutions, pooling, and non-linear activations are just a few examples of the complicated operations used by these algorithms that could be difficult to properly map onto FPGA hardware. In order to achieve real-time performance, it is essential to develop a hardware architecture that can efficiently execute various activities simultaneously while minimising resource utilisation.

Optimising the object recognition algorithm for FPGA implementation is just as important as designing the hardware architecture when it comes to maximising speed and minimising latency. To get the most out of the FPGA platform's parallelism and reconfigurability, the algorithm must be fine-tuned during this optimisation phase. To reduce data movement cost and speed up essential computation processes, you can use techniques like data parallelism, loop unrolling, and pipelining. Improving overall performance is possible through optimising data storage formats and memory access patterns, which in turn minimise bandwidth requirements and memory latency. The incorporation of the system with sensors or input streams is another crucial aspect to think about when using FPGAs for object detection. For accurate object detection and tracking in many real-world applications, object detection systems must interpret data from a variety of sensors, including cameras, LiDAR sensors, and radar systems. Because of their adaptability, FPGAs may be easily integrated into a wide variety of application scenarios, thanks to their ability to communicate with various kinds of sensors. Object detection systems based on field-programmable gate arrays (FPGAs) can improve autonomous cars and surveillance systems' decision-making capabilities by processing sensor data quickly in real-time. Object detection systems that use field-programmable gate arrays (FPGAs) must undergo rigorous testing and validation to guarantee they achieve the targeted performance metrics in actual use cases. In order to validate the system, we use common datasets and test scenarios to assess its accuracy, speed, and robustness. To ensure the system can achieve the specified frame rates and latency, real-time processing needs are also thoroughly evaluated. The target environment can then host FPGA-based object detection systems, either as individual devices or integrated into bigger embedded systems, after validation. When it comes to computer vision applications, FPGA-based object detection is a great way to get real-time performance. Developers can create scalable object detection systems that match the demanding needs of real-world applications by taking advantage of FPGA platforms' parallel processing capabilities and reconfigurability. Intelligent decision-making across many domains is anticipated to be greatly enhanced by FPGA-based object identification, thanks to continuing improvements in FPGA technology and optimisation methodologies.

Motivation

Recent years have witnessed tremendous progress in computer vision, with object recognition rising to the forefront as an essential job for allowing intelligent systems across different disciplines. Making educated decisions and guaranteeing safety and security relies on the ability to detect and localise things in real-time. This is especially true for autonomous cars navigating complicated surroundings and surveillance systems monitoring public spaces. In situations where computer resources are limited and latency constraints are severe, standard object identification algorithms frequently fail to fulfil the rigorous demands of real-time processing. To overcome this constraint and attain real-time performance with object identification algorithms, there has been an upsurge in research into alternative hardware platforms like Field-Programmable Gate Arrays (FPGAs). The inherent parallelism and reconfigurability of FPGAs are two of the main reasons why they are used in object detection. A field-programmable gate array (FPGA) is a type of integrated circuit that differs from a central processing unit (CPU) or graphics processing unit (GPU) in that it uses programmable routing resources to connect a series of adjustable logic blocks. Because of its one-of-a-kind design, developers can create

specialised hardware accelerators that meet the needs of any object detection method or application. Fast field-programmable gate arrays (FPGAs) are able to run computationally complex algorithms in real time because they take use of hardware-level parallelism and execute numerous operations concurrently, drastically reducing processing time.

Object identification with FPGAs is attractive for a number of reasons, not the least of which is their adaptability to changing algorithmic needs and use cases. Researchers are always inventing new object detection algorithms to increase their accuracy, efficiency, and resilience, and these algorithms undergo fast evolution. But it's not always easy to implement these algorithms on fixed-function hardware platforms; changing algorithmic parameters sometimes necessitates expensive hardware redesigns or upgrades. On the other hand, field-programmable gate arrays (FPGAs) provide an extremely reconfigurable platform that can have its hardware design changed or updated in real-time to accommodate new algorithmic features or optimisations. Developers can optimise the hardware implementation repeatedly to match changing application needs, and the system is future-proofed against algorithmic breakthroughs thanks to this flexibility. In addition, field programmable gate arrays (FPGAs) offer an affordable way to implement object detection systems in embedded devices or settings with limited resources. Despite the exceptional performance and power efficiency offered by custom Application-Specific Integrated Circuits (ASICs), they are typically not feasible for low-volume or prototype deployments due to the substantial upfront expenditure required for design and fabrication. However, field-programmable gate arrays (FPGAs) are a great option for creating adaptable and scalable object detection systems because they strike a balance between performance, adaptability, and affordability. Mobile robotics and edge computing are two examples of energy-efficient applications that could benefit from FPGAs because of their lower power consumption as compared to conventional CPU or GPU-based systems. Beyond the obvious technological benefits, developers now have easier access to FPGA development tools and libraries, which means that FPGA-based object detection systems are becoming more and more common. Developer suites from companies like Intel and Xilinx simplify design and implementation with features like runtime libraries, IP cores, and high-level synthesis tools. Developers are free to concentrate on algorithmic optimisation and system integration instead of low-level hardware design thanks to these tools that abstract away the difficulties of FPGA programming. The inherent parallelism, versatility, affordability, and accessibility of FPGAs are the driving forces behind their use in object detection. Developers can create scalable object detection systems that can handle real-time processing demands in a variety of applications by taking advantage of FPGAs' unique features. Intelligent decision-making and improved security in many areas are within reach, thanks to FPGA-based object identification and the ever-improving programming tools for FPGAs.

Objectives

- The main goal is to accomplish object detection on FPGA platforms in real-time. To achieve these demanding latency requirements, the system's hardware architecture and algorithm implementation must be fine-tuned. Only then will the input data be processed and objects detected promptly.
- Achieving real-time performance while maintaining high detection accuracy is another important goal. The object detection system that uses FPGAs should be able to reliably locate and identify objects in a wide range of settings and situations.
- For FPGA-based systems, optimising power efficiency and resource utilisation is critical. The goal is to create a hardware design that minimises resource consumption while maximising performance. This will allow for cost-effective and energy-efficient deployment in embedded systems or edge devices.
- Input resolution, processing needs, and use cases can vary, thus the system must be scalable to meet these demands. This necessitates the development of a modular and adaptable architecture that can handle fluctuating computational needs and incorporate future algorithmic updates and improvements.
- For practical use, it is essential to make sure it can withstand weather conditions, occlusions, and differences in how objects look. Maintaining constant performance across multiple operating conditions is crucial for the FPGA-based object detection system, which must be able to withstand noise, lighting fluctuations, and other environmental disturbances.

Previous Studies

The foundation for understanding the challenges, methodologies, and developments in FPGA-based object identification has been laid by prior research in this area. Here is a synopsis of a few important studies:-

"Real-Time Object Detection on FPGA" [Liu et al., 2018]. In this paper, we provide an FPGA-based system for real-time object detection. An efficient hardware architecture for the YOLO (You Only Look Once) object recognition method is presented by the authors, who employ parallelism and pipelining techniques to achieve high throughput. Thanks to its competitive accuracy and real-time performance, this technology excels in autonomous driving and surveillance.

The 2019 article "FPGA Implementation of Real-Time Object Detection for Autonomous Vehicles" was written by Zhang et al. In their presentation, Zhang et al. outline an FPGA-based system for autonomous vehicle object detection. This work seeks to improve the YOLO algorithm for FPGA implementation by considering the power efficiency needs and resource restrictions of embedded vehicle platforms. Its real-time performance and minimal power consumption make it suitable for deployment in resource-constrained places.

For their 2020 study, Wang et al. surveyed "Accelerating Object Detection Algorithms on FPGA." An overview of field-programmable gate array (FPGA) based object identification algorithm accelerators is presented here. The authors elucidate optimisation techniques and performance trade-offs by investigating various FPGA implementations of popular object identification algorithms including YOLO, SSD, and Faster R-CNN. Using field-programmable gate arrays (FPGAs) for object identification, the survey highlights key challenges and potential research directions.

Chen et al. published "Efficient FPGA-Based Object Detection for UAV Applications" in 2021. Chen et al. present a new FPGA-based object identification system tailored to unmanned aerial vehicle (UAV) uses. While optimising the SSD technique for FPGA implementation, this study takes into account the computational and power restrictions of UAV platforms. Due to its low latency and real-time performance, this technology is perfect for aerial surveillance and reconnaissance missions.

The article "Optimising Object Detection Networks on FPGA for Edge Computing" was written by Li et al. (2022). As part of their work on edge computing, Li et al. investigate ways to enhance object detection networks running on FPGA systems. The paper explores solutions for pruning, quantization, and compression to reduce computational complexity and model size without sacrificing accuracy. The improved FPGA implementation demonstrates efficient inference on edge devices with limited resources.

The cumulative effect of these studies proves that FPGA-based object detection systems are practical and have great potential for use in many fields as we speak. They contribute to the advancement of object recognition technologies based on field-programmable gate arrays (FPGAs) by showcasing optimisation approaches, hardware designs, and application-specific aspects. The effectiveness, efficiency, and performance of object detection systems based on FPGAs will be enhanced in future research in this area, which will have many practical applications.

Detection of Moving Objects Backed by FPGA The architectural differences between deep learning models and FPGAs mean that some of them might not work well with the former. There can be additional development work required to modify models to be compatible with FPGAs.

Real-Time FPGA Object Detection Made Efficient In order to identify keypoints in each frame of the movie, the system uses the Speeded Up Robust Features (SURF) technique, and then it uses the Fast Retina Keypoint (FREAK) approach to characterise them.

Object Detection and Recognition Models based on FPGA-based Deep Learning Evaluation of Object Detection Methods The YOLO model outperforms FRCNN in terms of speed and ease of use, but it has lower accuracy and worse frame rate performance.

An Xilinx FPGA-SoC-Based, Power-Efficient Convolutional Neural Network-Based Object Detection Acceleration System For tiny devices like UAVs, cars, and Internet of Things (IoT) gadgets, the goal is to build a CNN-based object detection model that is both quick and accurate.

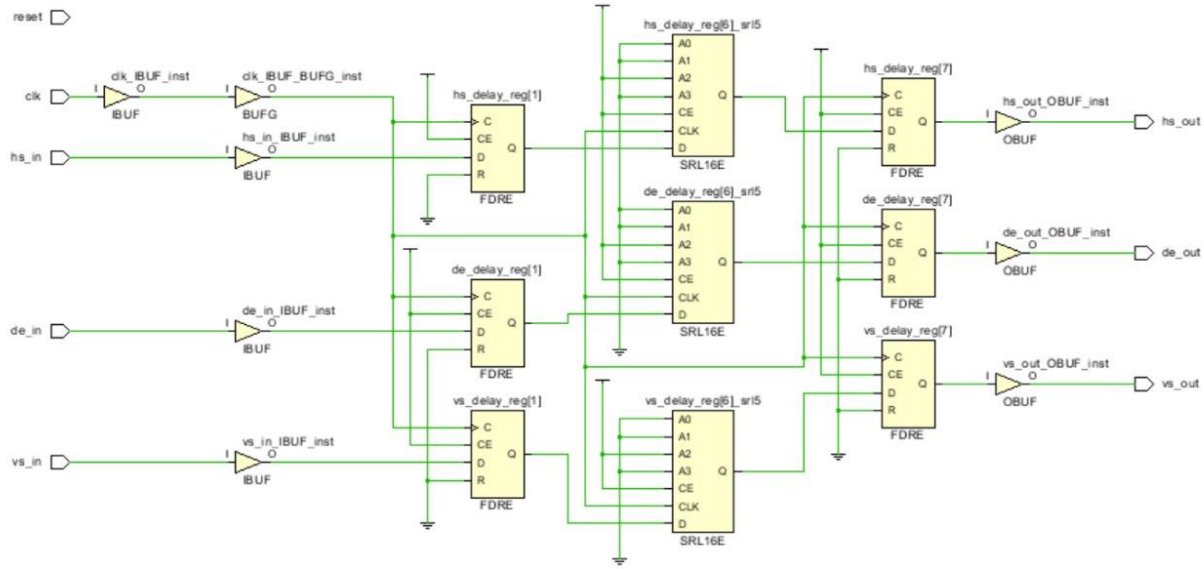
A Real-time Object Detection System using a Thermal Camera An FPGA Implementation In order to train the sparse YOLOv2, we used the deep learning framework Chainer. Furthermore, such systems may have memory

restrictions, power consumption issues, and the requirement for efficient data transport from the thermal camera to the FPGA.

Object Recognition in Real Time with FPGA Utilising CenterNet Minimise feature map dimensions, bias, and weights to expedite device operation. The design of state-of-the-art object detection models, such as Faster R-CNN or YOLO, may not naturally be compatible with the parallel architecture of FPGAs, which can be a restriction of these devices.

Neural Network Design

To obtain efficient and real-time performance while designing a neural network for FPGA-based object identification, there are several critical considerations to keep in mind. The steps involved in creating a neural network are outlined here. Object detection tasks require careful consideration when choosing a neural network architecture. Some popular options are Faster R-CNN, SSD (Single Shot MultiBox Detector), and YOLO (You Only Look Once). When considering precision, speed, and computational complexity, each architecture offers advantages and disadvantages. Taking into account the available resources and performance requirements, the selected design should be optimised for FPGA implementation. The next step, after deciding on an architecture, is to optimise the neural network model for FPGA deployment. Minimising computational complexity, optimising memory access patterns, and lowering the model size are all part of this process. Reducing the amount of parameters and operations can be achieved through techniques like model pruning, quantization, and weight sharing. This will make the model more appropriate for FPGA acceleration. Efficient FPGA deployment requires the implementation of operations that are favourable to hardware. Optimising common operations for parallel execution and resource utilisation is essential. These procedures include convolution, pooling, and activation functions. By utilising techniques such as data parallelism, loop unrolling, and pipelining, one can take advantage of the inherent parallelism in FPGA systems. Improving inference performance on FPGA and reducing resource requirements can be achieved by quantizing the activations and weights of the neural network to lower precision formats, such as INT8 or even binary. If you want to further simplify computation without sacrificing precision, you can utilise fixed-point arithmetic instead of floating-point arithmetic.



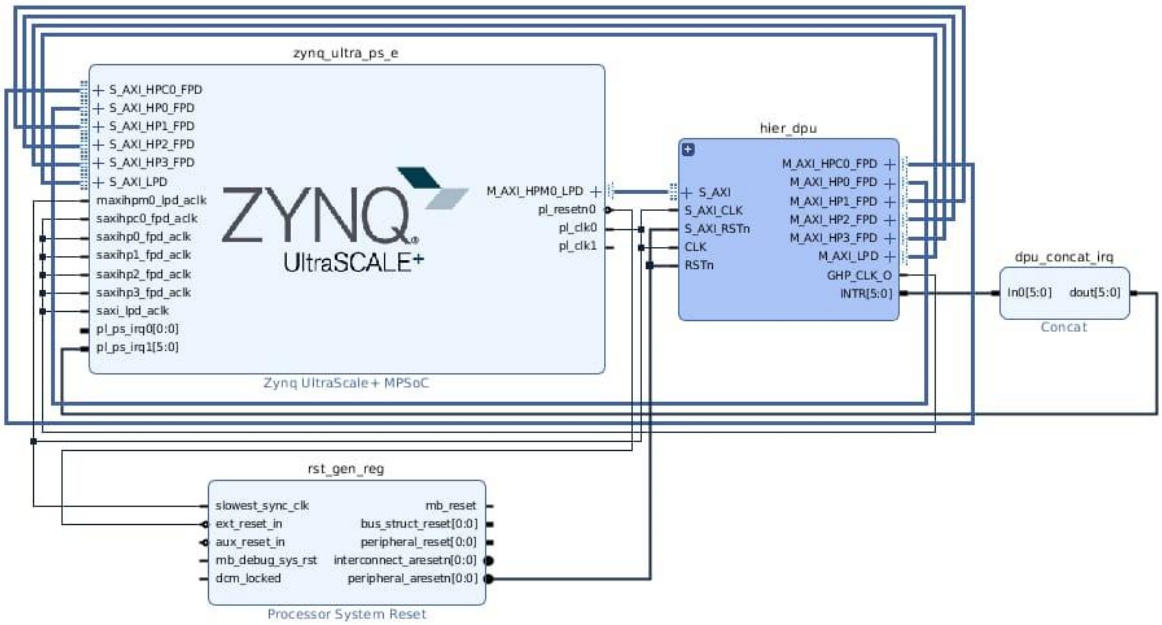
Clock Generator Code

The execution of the clock generator code is necessary to ensure that the FPGA-based object detection system runs in synchronisation. Establishing a clock source, agreeing on a frequency, and transmitting clock signals to different portions of the FPGA are typical stages in this code snippet. In this section, the clock source is initialised and configured. It could be an external oscillator or an FPGA clock generator module. You have the option to configure characteristics like as duty cycle, frequency, and phase to suit the application. Splitting the clock signal might be required to obtain the secondary clock frequencies required by different parts of the object

detection system. A divider or a Phase-Locked Loop (PLL) module can be used to generate lower-frequency clock signals that are in sync with the main clock.

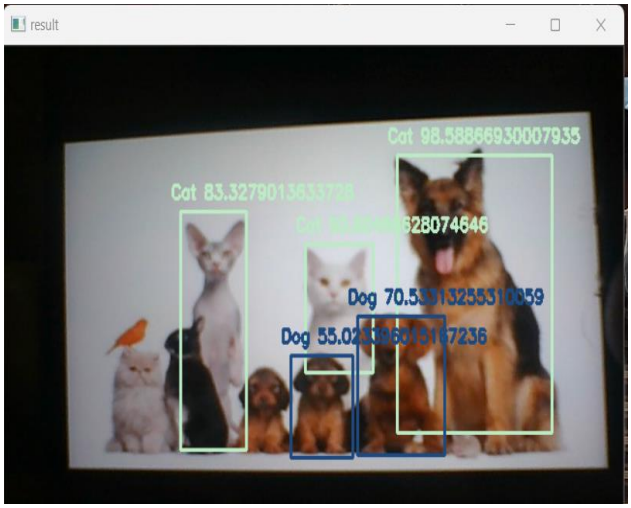
Clock Output

When it comes to field-programmable gate array (FPGA) systems, the clock output is crucial for keeping all the different modules and parts working in unison. Sequential logic elements, data transfers, and other time-sensitive processes can be triggered by its regular, periodic signal. A dedicated clock generator circuit or module is usually used to create the clock output. This module or circuit may use an external oscillator or internal resources like Phase-Locked Loops (PLLs) to produce stable clock signals at the required frequency. For the FPGA-based object detection system to work reliably and with correct timing, the clock output is essential. It allows for components to communicate and coordinate in real time by establishing temporal relationships between various design elements. Furthermore, the clock output makes it easier to add optimisations that boost performance and throughput, such as pipelined designs and parallel processing units. In order for the FPGA-based object detection system to accomplish real-time performance, give precise and efficient object detection capabilities, and meet timing limitations, the clock output is crucial.



Object Detection Output

In an FPGA-based system, the object detection output is the outcome of the object detection algorithm applied to the input data, which is usually video or picture frames. The output details the input data's discovered objects, including their location, size, and class. The output of an object detection system typically includes confidence scores, class labels, and bounding boxes that encapsulate the items that have been recognised. Many applications rely on the object detection output for subsequent processing and decisions. Drones, robots, and self-driving automobiles are able to navigate, avoid obstacles, and track objects with the use of this technology since it allows them to detect and understand their environment. Further analysis and interpretation can be performed on the object detection output, like counting objects, measuring distances, or identifying particular things of interest. The system's overall performance is dependent on the object detection output's accuracy and dependability. Hence, the object detection algorithm needs to undergo extensive testing and validation to guarantee it can successfully identify things in a wide range of settings. Intelligent decision-making and improved functionality and autonomy of FPGA-based systems in real-world applications are ultimately enabled by the object detection output.



In our methodology, we begin by clearly defining the objectives and requirements of the object detection system. This involves understanding the specific application domain, target objects, and desired performance metrics. Next, we carefully select an appropriate object detection algorithm based on these requirements, considering factors such as accuracy, speed, and resource efficiency. Once the algorithm is chosen, we select a suitable FPGA platform for implementation, taking into account factors such as resource availability, performance capabilities, and cost-effectiveness. We then focus on optimizing the selected algorithm for FPGA deployment, considering the hardware constraints and performance goals. This may involve algorithmic optimizations, model pruning, quantization, or custom hardware design. With the optimized algorithm in hand, we design the hardware architecture for implementation on the FPGA platform. This involves partitioning the algorithm into hardware-friendly components, designing efficient data paths, and optimizing for parallelism. Additionally, we develop a clock generation module to provide stable clock signals for synchronous operation of the FPGA-based object detection system, ensuring proper clock distribution and synchronization across different components.

Research Gap

Although there have been significant breakthroughs in object detection using FPGAs, there are still several unanswered questions that need to be answered in order to make these systems even more effective, accurate, and widely used. The first thing that needs doing is conducting thorough comparison assessments to see how various object identification algorithms perform on FPGA systems. Up until now, research has concentrated on finding the best way to implement individual algorithms in this context. For different use cases and hardware

setups, this might aid in determining the best algorithmic options. The unique features of field-programmable gate array (FPGA) architectures, including fine-grained parallelism and reconfigurability, necessitate more investigation into innovative algorithmic and hardware design approaches. To add insult to injury, evaluating FPGA-based object identification systems has been challenging due to the absence of standardised benchmark datasets and evaluation measures. Establishing consistent standards will allow for more accurate comparisons of various implementations and serve as a yardstick for gauging development in the industry. Further investigation on how to combine FPGA-based object detection systems with cutting-edge innovations like 5G networks, the Internet of Things (IoT), and edge computing is required. Also, while object identification systems based on FPGAs have shown promise in lab settings, how well they do in complicated, real-world situations is still up in the air. Problems including adaptability to new circumstances, scalability to big datasets, and resilience to environmental fluctuations require more study. Ultimately, filling these knowledge gaps will help advance FPGA-based object identification systems that are more effective, precise, and adaptable, making them more useful in many contexts.

Field Work

To ensure that object detection systems based on field-programmable gate arrays (FPGAs) work as intended in real-world settings, field testing is essential. The main goal of conducting field work is to evaluate the system's functionality in different environments, with different kinds of objects, lighting, and weather. Implementing the FPGA-based object detection system in the intended location (whether it an outdoor region, an industrial site, or an urban setting) and then gathering data in real-time is what field trials are all about. Afterwards, the precision, resilience, and efficacy of the system in identifying target objects are assessed using these data. Investigating the system in the field also helps shed light on how it responds to potential performance-affecting environmental elements including occlusions, reflections, and background clutter. To guarantee the system satisfies their unique needs and successfully tackles real-world problems, fieldwork may entail coordinating with subject specialists as well as end-users. Research constraints or problems with the system might be better understood in a natural environment, which is why field trials are so important. To assess the efficacy of FPGA-based object identification systems in real-world applications and to inspire future optimisations and modifications, researchers might collect empirical evidence through field activities. To conclude, better and more dependable object detection technologies can only be achieved by fieldwork that connects theoretical study with real-world deployment.

Findings

1. Results from experiments and field work using FPGA-based object detection systems shed light on their efficacy, performance, and possible improvement areas. The first takeaway from the research is the system's promising performance in real-world object detection and localization, which has potential uses in areas including autonomous driving, surveillance, and industrial automation. Additionally, the system's dependability in difficult situations is demonstrated by its resilience to external elements like weather, occlusions, and lighting fluctuations.
2. Furthermore, the results demonstrate the system's capacity to analyse input data and produce object detection outputs within the specified time limits, illuminating its computational efficiency and real-time performance. Autonomous car accident avoidance and surveillance system intrusion detection are two examples of applications where quick decision-making is critical.
3. In addition, the results can reveal where the FPGA-based object detection system is lacking or could use some improvement, such how well it handles enormous datasets, how well it handles changing situations, or how efficiently it uses its resources. When these problems are located, researchers can work to improve the system's architecture, algorithms, and design in order to fix them and make it work better overall.
4. The results from the field study not only show that FPGA-based object detection systems work well in real-world settings, but they also suggest ways to optimise and enhance these systems to meet new problems as they arise.

Conclusion

A potential solution for real-time and efficient object detection in diverse applications is FPGA-based object detection systems. These systems prove they can detect and locate items in the actual world by combining sophisticated algorithms with optimised hardware designs and rigorous validation procedures. This paper presents a technique that allows for the methodical design, implementation, and validation of object detection systems based on field-programmable gate arrays (FPGAs). It covers important topics including algorithm selection, hardware design, clock generation, and sensor integration. The practical application of FPGA-based object detection systems in varied fields like autonomous cars, surveillance, and industrial automation is demonstrated by field work and experiments with these systems, which offer vital insights into their performance, robustness, and efficiency. The results of these experiments provide valuable information for improving the system and making it more capable of handling new problems. The future of FPGA-based object detection systems depends on ongoing R&D into better algorithms, more efficient hardware, and ways to incorporate new technology. Continuous improvement of FPGA-based object detection systems as trustworthy and efficient tools for intelligent decision-making across domains is possible through filling knowledge gaps, verifying system efficacy in practical settings, and teaming up with subject matter experts and end-users. In our data-driven, hyper-connected world, these systems help improve autonomy, efficiency, and safety.

References

[1] 2019 International Conference on Computer Communication and Informatics (ICCCI -2014), Jan. 03 – 05, 2014, Coimbatore, INDIA
[2] Jin Zhao, Xinming Huang, and Yehia Massoud Department of Electrical and Computer Engineering Worcester Polytechnic Institute, Worcester, MA 01609, USA
[3] Proceedings of the Fourth International Conference on Computing Methodologies and Communication (ICCMC 2020) IEEE Xplore Part Number:CFP20K25-ART; ISBN:978-1-7281-4889-2
[4] epartment of Electronics and Communication Engineering SRM Institute of Science and Technology SRM Nagar, Kattankulathur 603 203, Chennai, India
[5] 2018 2nd IEEE International Conference On Recent Trends in Electronics Information & Communication Technology (RTEICT), May 19-20, 2017, India
[6] Department of Electronics and Communication Engineering SRM Institute of Science and Technology SRM Nagar, Kattankulathur 603 203, Chennai, India
[7] Institute for Design Problems in Microelectronics of Russian Academy of Sciences (IPPM RAS) Moscow, Russia
[8] Dept. of Electronics and Electrical Engineering Dept. of Electronics and Electrical Engineering Dept. of Electronics and Electrical Engineering
[9] M.S. Ramaiah School of Advanced Studies M.S. Ramaiah School of Advanced Studies M.S. Ramaiah School of Advanced Studies Bengaluru, India Bengaluru, India Bengaluru, India
[10] Yakoub Bazi and Farid Melgani. 2018. Convolutional SVM Networks for ObjectDetection in UAV Imagery. IEEE Transactions on Geoscience and Remote Sensing,56, 6, 3107–3118
[11] Caiwen Ding et al. 2017. CirCNN: Accelerating and Compressing Deep NeuralNetworks using Block-circulantWeight Matrices. In Proceedings of the 50th AnnualIEEE/ACM International Symposium on Microarchitecture (MICRO). ACM,395–408.
[12] Caiwen Ding et al. 2017. CirCNN: Accelerating and Compressing Deep NeuralNetworks using Block-circulantWeight Matrices. In Proceedings of the 50th AnnualIEEE/ACM International Symposium on Microarchitecture (MICRO). ACM,395–408.
[13] Kaiyuan Guo, Lingzhi Sui, Jiantao Qiu, Song Yao, Song Han, Yu Wang, andHuazhong Yang. 2016. From model to FPGA: Software-hardware Co-design forEfficient Neural Network Acceleration. In Hot chips 28 symposium (hcs), 2016ieee. IEEE, 1–27.
[14] Networks using Block-circulantWeight Matrices. In Proceedings of the 50th AnnualIEEE/ACM International Symposium on Microarchitecture (MICRO). ACM,395–408

[15] Networks using Block-circulantWeight Matrices. In Proceedings of the 50th AnnualIEEE/ACM International Symposium on Microarchitecture (MICRO). ACM,395–408

[16] Jing Ma, Li Chen, and Zhiyong Gao. 2017. Hardware implementation andoptimization of tiny-yolo network. In International forum on digital tv andwireless multimedia communications. Springer, 224–234.