electronics

MDPI

Article

EOS: Edge-Based Operation Skip Scheme for Real-Time Object Detection Using Viola-Jones Classifier

Cheol-Ho Choi *, Joonhwan Han, Hyun Woo Oh, Jeongwoo Cha and Jungho Shin

Electro Optics R&D Center, Hanwha Systems Co., Ltd., 188, Pangyoyeok-ro, Bundang-gu, Seongnam-si 13524, Gyeonggi-do, Republic of Korea; joonhwan.han@hanwha.com (J.H.); hyunwoo.oh@hanwha.com (H.W.O.); jeongwoo.cha@hanwha.com (J.C.); jh.hoya.shin@hanwha.com (J.S.)
* Correspondence: cheoro1994@hanwha.com; Tel.: +82-31-8091-7680

Abstract: Machine learning-based object detection systems are preferred due to their cost-effectiveness compared to deep learning approaches. Among machine learning methods, the Viola-Jones classifier stands out for its reasonable accuracy and efficient resource utilization. However, as the number of classification iterations increases or the resolution of the input image increases, the detection processing speed may decrease. To address the detection speed issue related to input image resolution, an improved edge component calibration method is applied. Additionally, an edge-based operation skip scheme is proposed to overcome the detection processing speed problem caused by the number of classification iterations. Our experiments using the FDDB public dataset show that our method reduces classification iterations by 24.6157% to 84.1288% compared to conventional methods, except for our previous study. Importantly, our method maintains detection accuracy while reducing classification iterations. This result implies that our method can realize almost real-time object detection when implemented on field-programmable gate arrays.

Keywords: object detection; machine learning; Viola-Jones classifier; operation skip scheme

Academic Editors: Aryya Gangopadhyay, Yue Zhang, Bin Chen, Jinlin Guo and Xueliang Liu

Received: 18 December 2024
Revised: 2 January 2025
Accepted: 18 January 2025
Published: 20 January 2025

Citation: Choi, C.-H.; Han, J.; Oh, H.W.; Cha, J.; Shin, J. EOS: Edge-Based Operation Skip Scheme for Real-Time Object Detection Using Viola-Jones Classifier. *Electronics* **2025**, *14*, 397. https://doi.org/10.3390/electronics14020397

1. Introduction

Driver assistance systems leveraging both inside and outside cameras are increasingly integrated into vehicles nowadays. In modern vehicles equipped with advanced driver assistance systems (ADASs), the primary electronic control unit (ECU) either manages individual sensors or consolidates collected data to deliver optimal information to drivers and passengers [1]. Consequently, vehicle manufacturers are now requesting functionalities such as object detection and recognition to be performed by cores present within individual sensors in an embedded environment. Particularly for in-cabin systems, manufacturers aim to lower sensor supply costs to reduce overall vehicle manufacturing expenses.

For in-cabin systems among ADASs, the European New Car Assessment Programme (EURO NCAP) mandates that cameras used in automotive child presence detection (CPD) systems by 2025 must execute object detection and recognition functions at minimal cost [2]. Due to these requirements, various artificial intelligence (AI)-based CPD systems are being considered to meet EURO NCAP standards [3–5]. As illustrated in Figure 1, deep learning techniques typically entail complex classifier architectures comprising feature extraction and classification processes [6–10]. However, the intricate and deep architecture poses challenges for implementation on embedded platforms. Furthermore, given the characteristics of these architectures, high-performance processors and memory resources (such as DDR or Flash) are necessary to achieve desired processing speeds.

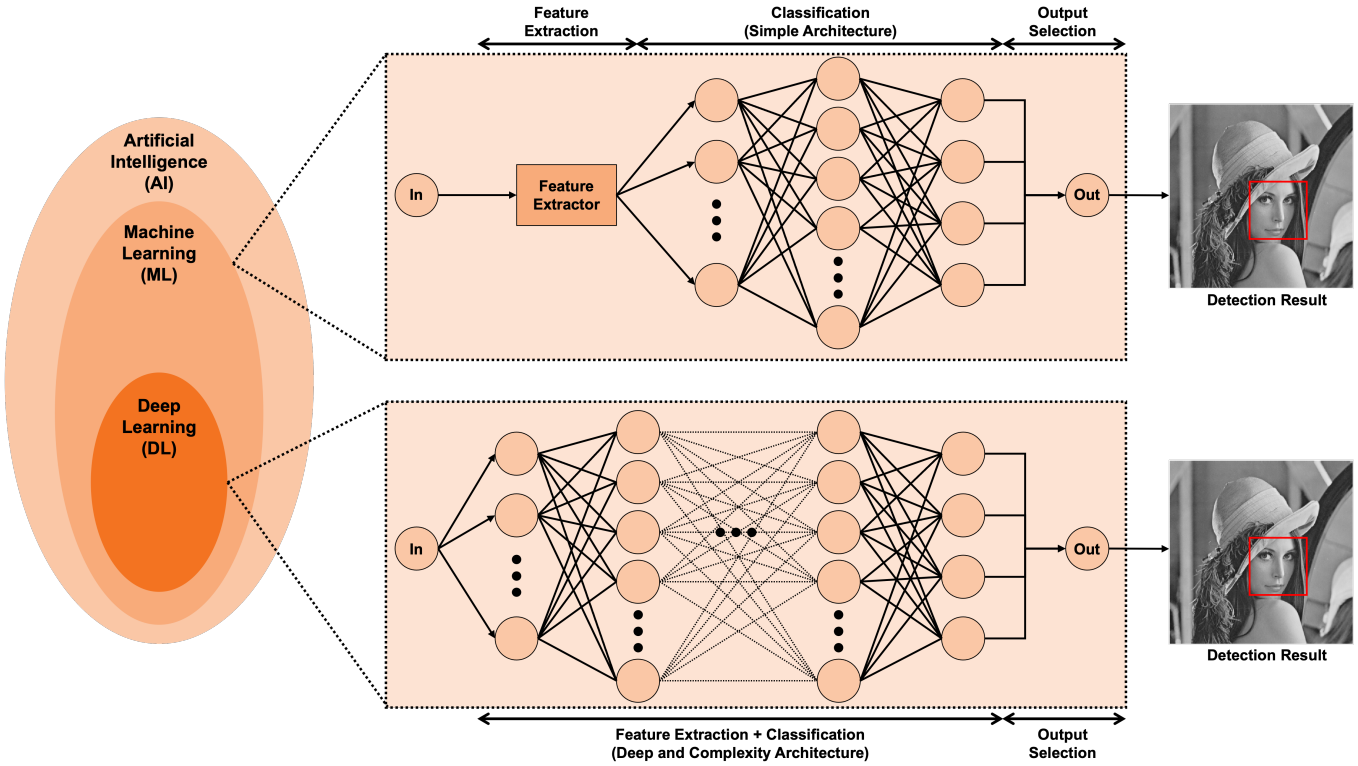


Figure 1. Analyzing machine learning and deep learning approaches in artificial intelligence research and development: A comparative perspective (Red box: Detected result).

Since various original equipment manufacturing (OEM) companies (such as Hyundai, Toyota, and Volkswagen) require embedded platform-based CPD systems, these drawbacks emerge as critical factors in the development of commercial products. In CPD systems, machine learning techniques are being explored more than deep learning, due to the feasibility of utilizing low-cost processors, reducing mass production costs. Among the machine learning-based algorithms for object detection and recognition, the Viola-Jones classifier, commonly referred to as the Haar cascade classifier, is widely preferred [11,12]. This preference stems from its simple classification structure and parameters, as well as its utilization of a cascade classifier architecture, which enables faster processing speeds compared to other techniques [13]. However, similar to other machine learning methods, the processing speed decreases as the input image resolution increases and the number of classification repetitions for detection rises [14].

To improve processing speed, in our previous study [15], we proposed an enhanced object detection algorithm using the Viola-Jones classifier with an edge-based skip scheme and edge component calibration method. However, the method proposed in the previous study performs edge calibration operations at each step, which may affect detection performance as edge components are excessively added. Additionally, the previous study failed to demonstrate objective performance using public datasets. Therefore, in this paper, we propose an improved method to overcome the limitations of the previous study's approach. Furthermore, we present experimental results using the publicly available face detection dataset and benchmark (FDDB) [16] database to evaluate the performance of the proposed method compared to conventional methods before implementing a field-programmable gate array (FPGA)-based object detection system.

2. Related Work

2.1. Traditional Viola-Jones Classifier

Viola proposed an algorithm using a cascade classifier architecture with Haar-like features for object detection [11,12]. Although the algorithm was initially proposed for face detection, it is being used in various object detection systems, such as those for vehicles and pedestrians, due to its simple architecture that facilitates implementation on embedded platforms [17–19].

The Viola-Jones classifier algorithm operates through five main steps: (1) Pyramidal Image Generation, (2) Window Generation, (3) Integral Image Generation, (4) Cascade Classification, and (5) Coordinate Merging. In the pyramidal image generation step, the algorithm creates multiple versions of the image, each resized to a smaller scale, but still larger than the predefined window size. This step is crucial because the Viola-Jones classifier uses a fixed-size window for object detection, and resizing the image allows the detection of objects of various sizes. Next, during window generation, a sliding window technique is applied to the pyramidal images. Since the window size is fixed, the pyramidal images allow the detection of differently scaled objects within the scene. In the integral image generation step, the integral image is calculated for each window. This process significantly reduces the computational complexity of memory access, as well as the addition and subtraction operations, enhancing the algorithm's efficiency. During the cascade classification step, the integral image is evaluated to determine whether an object is present in the window. If an integral image passes this classification, the corresponding coordinates are output as a detected object. If it fails, no coordinates are output for that window. Finally, during the coordinate merging step, the coordinates of all objects detected through the cascade classification are combined, ensuring that the final set of coordinates provides a clear and visually cohesive representation of detected objects.

Thus, the traditional Viola-Jones classifier algorithm is capable of detecting objects of various sizes in an input image by applying the sliding window technique across pyramidal images. However, a known limitation is that processing speed may decrease when the resolution of the input image is high, or when the number of false positives in background regions increases.

2.2. Viola-Jones Classifier with Skip Scheme Based on Intersection over Union (IoU)

To address the drawback of processing speed, Hyun proposed a skip scheme in 2021 to reduce the number of classification operations [20]. The concept of the skip scheme is to avoid performing object detection operations in regions that are eliminated when coordinates are merged in the input image using the same scale factor value generated by the image pyramid technique.

For instance, consider using the Intersection over Union (IoU) with an input window size of 20×20. When the input window at $(10, 10)$ is identified as an object, the object detection operation is not performed up to the coordinates with an IoU of 0.5 or more. Instead, the operation is conducted again at $(16, 10)$ when the window moves six pixels horizontally. Consequently, since $(11, 10)$ to $(15, 10)$ is recognized as an object region and detection operation is skipped, there is an advantage in reducing unnecessary operations to improve processing speed.

However, if no object exists in the input image, the processing speed remains the same as the Viola-Jones classifier algorithm. Additionally, if the number of objects in the input image is significantly small, the processing speed may slightly improve, but there is no significant difference compared to the Viola-Jones classifier. Therefore, the skip scheme proposed by Hyun only offers an advantage when there are many objects in the input image. This is because the number of regions identified as objects increases as the number

of skipped operations based on the IoU metric grows, thereby reducing unnecessary classification processes.

2.3. Viola-Jones Classifier with Wavelet Transform

In 2022, Choi proposed an edge component calibration method using two-dimensional (2-D) Haar wavelet transform to enhance processing speed while maintaining detection performance [14]. By employing the 2-D Haar wavelet transform, the resolution of the input image can be reduced, leading to improved processing speed due to the downsizing of the input image. However, when using the 2-D Haar wavelet transform to decrease the input image resolution, edge component information may be lost in the computed approximation image, known as the de-noised image. Consequently, the computed horizontal, vertical, and diagonal components obtained using the 2-D Haar wavelet transform are calibrated from the approximation image to preserve detection performance.

The utilization of the edge component calibration method based on the 2-D Haar wavelet transform results in a significant improvement in processing speed. Nevertheless, a large number of classification operations are still required for the object detection process due to unnecessary computations in background regions, referred to as non-object regions.

2.4. Our Previous Work

In our previous research, we introduced an edge-based operation skip scheme to enhance processing speed by reducing unnecessary classification operations on background regions [15]. Additionally, we proposed a revised edge component calibration method inspired by Choi's work to maintain detection performance compared to conventional methods. Therefore, we employed the 2-D wavelet transform for each image resizing step to generate pyramidal images and applied both the edge-based operation skip scheme and the revised edge component method.

By integrating these two image processing concepts, we demonstrated that our previously proposed algorithm achieved fast processing speed while maintaining detection performance compared to conventional methods. However, since only two test images were used in our previous study, there is uncertainty regarding the robustness of the proposed method. Furthermore, as the resolution of the input image used in the cascade classifier architecture decreases, there is a risk of excessively calibrating edge components. This occurs because the revised edge component calibration method using 2-D wavelet transform is applied to each pyramidal image. In other words, this factor could impact detection performance when experiments are conducted on various images.

3. Proposed Method

To address the processing speed limitation inherent in conventional methods, including those outlined in our previous work, we have proposed an enhanced edge-based operation skip scheme for the Viola-Jones classifier. Figure 2 illustrates the operation process of our proposed method, with each subsection providing detailed information on specific aspects of the operation process.

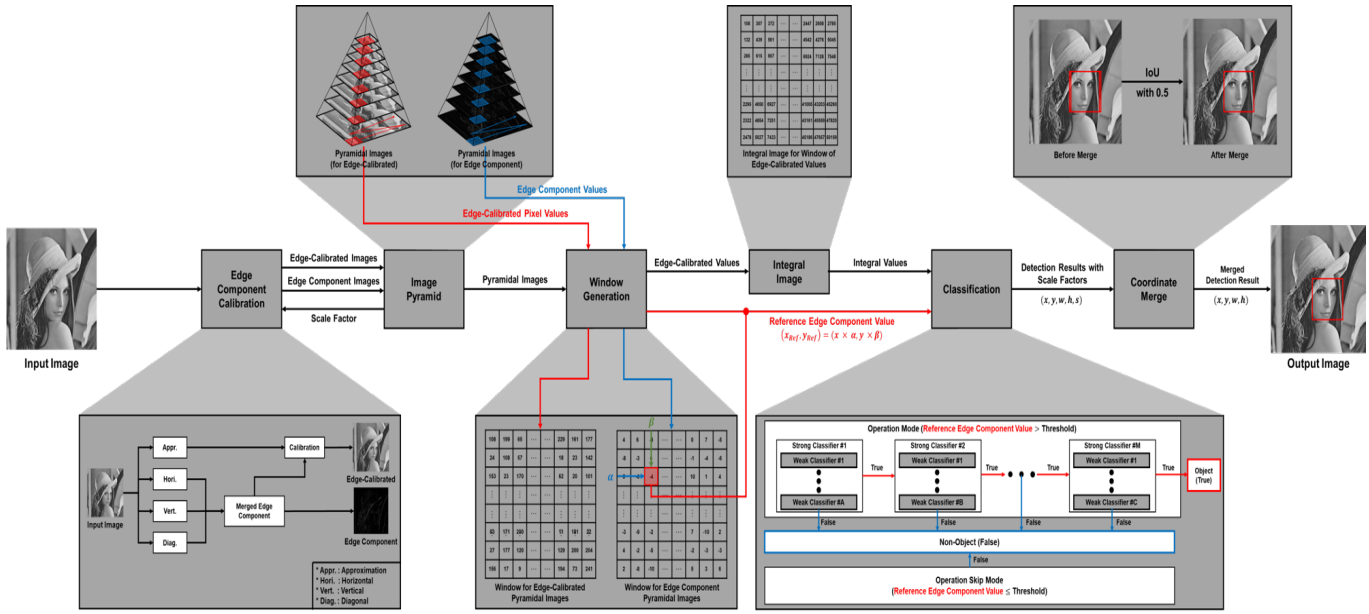


Figure 2. Process overview: Integrating the proposed edge-based skip scheme with the Viola-Jones classifier algorithm.

3.1. Edge Component Calibration

To maintain detection performance while reducing the input image resolution, we implement the concept of edge component calibration proposed by Choi [14]. In our previous work, we utilized N-level Haar wavelet transform to calibrate the edge component from each level de-noised image. However, employing N-level Haar wavelet transform might excessively calibrate the edge component, potentially compromising detection performance. Hence, in this study, we opt for a one-level 2-D Haar wavelet transform to decrease the input image resolution while incorporating edge component calibration to uphold detection performance.

When the scale factor is 1 (i.e., using the original image as input and employing 1-level 2-D Haar wavelet transform simultaneously), the edge component calibration process involves computing two types of images: (1) an edge-calibrated image and (2) an edge component image. For the edge-calibrated image, the process comprises three steps: (1) 2-D Haar wavelet transform (down-sampling), (2) merged edge component computation, and (3) edge component calibration. On the other hand, the edge component image involves the following three steps: (1) 2-D Haar wavelet transform (down-sampling), (2) rectified linear unit (ReLU), and (3) merged edge component computation.

3.2. Two-Dimensional (2-D) Wavelet Transform

For edge component calibration, a down-scaled image and edge component values are first required, and for this purpose, the proposed method utilizes 2-D Haar

wavelet transform. To achieve this goal, the 2-D Haar wavelet transform is applied using Equations (1)–(5).

$$T_i(x,y) = \begin{cases} \frac{1}{4} \times \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} & \text{if } i = SS, \\ \frac{1}{2} \times \begin{bmatrix} -1 & -1 \\ 1 & 1 \end{bmatrix} & \text{if } i = SW, \\ \frac{1}{2} \times \begin{bmatrix} -1 & 1 \\ -1 & 1 \end{bmatrix} & \text{if } i = WS, \\ \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} & \text{if } i = WW \end{cases} \tag{1}$$

$$I_{LL}(x,y) = \sum_n \sum_m T_{SS}(n,m) \cdot I(2x-n, 2y-m) \tag{2}$$

$$I_{LH}(x,y) = \sum_n \sum_m T_{SW}(n,m) \cdot I(2x-n, 2y-m) \tag{3}$$

$$I_{HL}(x,y) = \sum_n \sum_m T_{WS}(n,m) \cdot I(2x-n, 2y-m) \tag{4}$$

$$I_{HH}(x,y) = \sum_n \sum_m T_{WW}(n,m) \cdot I(2x-n, 2y-m) \tag{5}$$

where I_{LL} represents the approximation image, also called the de-noised image or low-low (LL) coefficient image, I_{LH} stands for the horizontal edge component image, referred to as the low-high (HL) coefficient image, I_{HL} denotes the vertical edge component image, termed as the high-low (HL) coefficient image, and I_{HH} signifies the diagonal edge component image, labeled as the high-high (HH) coefficient image. The original input image is represented as $I(2x-n, 2y-m)$, and $T_i(x,y)$ denotes the predefined kernel for 2-D Haar wavelet transform computation. Specifically, T_{SS} indicates the predefined kernel shape for I_{LL}, T_{SW} for I_{LH}, T_{WS} for I_{HL}, and T_{WW} for I_{HH}.

In general 2-D wavelet transform, four types of kernels are required for computing the de-noised image and three edge component images. Typically, 2-D Haar wavelet transform necessitates two transformation functions, namely scaling and wavelet functions, which act as low- and high-pass filters for the vertical and horizontal directions. However, employing these functions involves applying the transformation function sequentially to the horizontal and vertical directions. Consequently, processing time increases as the number of memory accesses rises, posing a critical disadvantage in reducing processing speed in embedded environments. To address this concern and enhance processing speed performance, we utilize a rewritten kernel shape as shown in Equation (1).

To compute the de-noised image as depicted in Equations (1) and (2), the rewritten kernel shape generated solely by the scaling function is applied to the original input image, as the approximation coefficient, also known as the LL coefficient image. Secondly, to compute the horizontal edge component image, as illustrated in Equations (1) and (3), the rewritten kernel shape generated by the scaling function with the wavelet function is applied. This is because the horizontal edge component image is referred to as the LH coefficient image. Thirdly, to compute the vertical edge component image, as demonstrated in Equations (1) and (4), the rewritten kernel shape generated by the wavelet function with the scaling function is applied. This is due to the vertical edge component image

being known as the HL coefficient image. Lastly, to compute the diagonal edge component image, as shown in Equations (1) and (5), the rewritten kernel shape generated solely by the wavelet function is applied. This is because the diagonal edge component image is referred to as the HH coefficient image. Consequently, by utilizing the 2-D Haar wavelet transform, the de-noised image, and the horizontal, vertical, and diagonal edge component images can be obtained.

3.2.1. Rectified Linear Unit (ReLU)

When using a 2-D Haar wavelet transform, the resulting image contains both negative and positive edge component values for horizontal, vertical, and diagonal components. To effectively apply the proposed edge-based operation skip scheme, reference edge component values are required. To achieve this goal, we need to generate an edge component image. If we directly utilize horizontal, vertical, and diagonal components that contain both negative and positive values, the edge values may become blurred (close to 0 pixel value) or disappear (have a pixel value of 0). Therefore, it is essential to extract only positive edge values (valid edge values when representing the edge image). ReLU effectively filters out negative values and retains only positive edge components, thus preventing blurring of edge values and ensuring sharper edge representation in the generated edge component image. The ReLU process for obtaining edge component images is as follows.

$$ReLU(I_{in}(x,y)) = \begin{cases} I_{in}(x,y) & \text{if } I_{in}(x,y) \geq R_T \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

$$I_{R.H}(x,y) = ReLU(I_{LH}) \tag{7}$$

$$I_{R.V}(x,y) = ReLU(I_{HL}) \tag{8}$$

$$I_{R.D}(x,y) = ReLU(I_{HH}) \tag{9}$$

where $I_{R.H}(x,y)$, $I_{R.V}(x,y)$, and $I_{R.D}(x,y)$ represent the rectified horizontal, vertical, and diagonal edge component images, respectively, $ReLU(\cdot)$ denotes the ReLU operation, and R_T is the threshold value for the ReLU operation. In the ReLU operation, the original pixel value is outputted when the pixel value of the input image exceeds the threshold value R_T. Conversely, the output value is set to 0 when the pixel value of the input image does not exceed R_T. Therefore, through the application of the ReLU operation, valid edge component values can be obtained for the edge component calibration process.

3.2.2. Merged Edge Component Computation

The objective of computing the merged edge component is to generate two edge component images, one by utilizing the edge component values computed using the 2-D wavelet transform and the other by utilizing the rectified edge component values applied with ReLU. The edge component image is generated by utilizing the non-rectified edge component values (the edge components computed by applying the 2-D wavelet transform) as shown in Equation (10).

$$I_M(x,y) = 2 \times (I_{LH}(x,y) + I_{HL}(x,y)) - I_{HH}(x,y) \tag{10}$$

where $I_M(x,y)$ represents the merged edge component image that will be used to generate the edge-compensated image.

As illustrated in Equation (10), generating the edge component image for creating the edge-calibrated image involves three arithmetic operations. First, the horizontal (LH coefficient image) and vertical (HL coefficient image) components are added. Second, the value

resulting from the addition of the horizontal and vertical components is multiplied by 2. Third, the diagonal (HH coefficient image) component is subtracted from the value obtained in the second step. The rationale behind multiplying the vertical and horizontal components by 2 is that Choi's study, which proposed the edge correction component, presented a value proportional to the diagonal component, so it was implemented accordingly.

The purpose of subtracting the diagonal component is to filter out excessive edge components that might have been included during the addition of vertical and horizontal components. If excessive or unnecessary edge components are included in generating the edge-corrected image, it can significantly impact accuracy performance. Thus, by subtracting the diagonal component from the combined vertical and horizontal components, only pixels likely to be edge components, which will be used for edge-calibrated image generation, are retained.

The rectified edge component image for the proposed edge-based operation skip scheme is generated by utilizing the rectified edge component values (obtained using the ReLU function), as depicted in Equation (11).

$$I_{ES}(x,y) = I_{R.H}(x,y) + I_{R.V}(x,y) + I_{R.D}(x,y) \tag{11}$$

where $I_{ES}(x,y)$ represents the merged edge component image that will be used for the proposed edge-based operation skip scheme.

As described in Equation (11), the merged edge component image for the proposed edge-based operation skip scheme is generated by summing all the rectified vertical, horizontal, and diagonal components. In contrast to Equation (10), where only individual rectified components are considered, the rationale for incorporating all rectified edge components is as follows. If the proposed edge-based operation skip scheme was to rely solely on a single edge component (such as only the vertical component), it might excessively skip the classification operation by disregarding edge information from the horizontal and diagonal directions. While such an approach could indeed improve processing speed substantially, it could also compromise accuracy performance, given the restricted areas where accurate classification operations would occur—a critical consideration. Hence, by utilizing the merged edge component image containing all rectified vertical, horizontal, and diagonal components, the proposed edge-based operation skip scheme can perform classification operations exclusively in regions where valid edge components are present, typically in areas with a high likelihood of containing objects. This decision is grounded in the fact that objects typically exhibit edge components across multiple directions, including vertical, horizontal, and diagonal.

3.2.3. Edge Component Calibration

The goal of the Edge Component Calibration step is to generate an image with calibrated edge components that can be used as input for object detection. The equation for generating an image with corrected edge components is as follows in Equation (12).

$$I_O(x,y) = I_{LL}(x,y) + I_M(x,y) \tag{12}$$

where $I_O(x,y)$ represents the output image, which is equivalent to the edge-calibrated image that will be used for image pyramid generation and the object detection process. The edge-calibrated image is generated by adding the merged edge component ($I_M(x,y)$) to the de-noised image ($I_{LL}(x,y)$, the approximation coefficient image).

3.3. Pyramidal Image Generation

After the edge component calibration process, two output images are obtained: the merged edge component image ($I_{ES}(x, y)$) for the proposed edge-based operation skip scheme (utilizing the ReLU function) and the edge component-calibrated image $I_O(x, y)$. Considering the various object sizes within the image, the image pyramid technique must be employed because the Viola-Jones classifier utilizes the sliding window technique with a predefined window size that is set during the training process. Additionally, to apply the proposed skip scheme, pyramidal images for the edge components of the same scale are required by applying the image pyramid technique. To achieve this objective, two types of pyramidal images are generated using Equations (13) and (14).

$$I_P = D(I_O, P) \tag{13}$$

$$E_P = \begin{cases} I_{ES} & \text{if } P = 1 \\ W(I_P, P) & \text{otherwise} \end{cases} \tag{14}$$

where $D(\cdot)$ represents the down-sampling function, $W(\cdot)$ represents the wavelet analysis function, P denotes the computation step for generating the pyramidal images, I_P denotes the output pyramidal edge component-calibrated image group, and E_P represents the output pyramidal edge component image group, respectively.

The down-sampling function ($D(\cdot)$) generally refers to a resize function (e.g., the *imresize* function in the MATLAB software tool (version: R2023, creator: MathWorks Inc., location: Natick, MA, USA)). Therefore, the edge-calibrated image (I_O) is used as the input parameter for the down-sampling function, and the generated pyramidal images are composed of images that are the result of sequentially reducing the edge-calibrated image. The wavelet analysis function ($W(\cdot)$) generates an edge image for each scale of the pyramidal image generated according to the computation step (P). Therefore, there are two cases for generating a pyramidal edge image that affect the scale. First, when P is 1, the previously generated edge image (I_{ES}) is placed at the beginning of the edge image pyramid group. Second, when P is not 1, the formula for obtaining the merged edge component image (I_{ES}) is applied in the same way. However, the difference when P is not 1 is that the same operation method as the wavelet filtering operation used in [21] is employed instead of the wavelet transform, where the image resolution is square (sliding window technique with stride 1 is used).

In this example, let us assume a scale factor of 2 for the image pyramid technique, a resolution of 320 $\times$ 240 for the edge component-calibrated image, and a 20 $\times$ 20 window size.

- In the first step ($P = 1$), the input image (edge component-calibrated image) is placed at the beginning of the image pyramid group.
- In the second step ($P = 2$), the resolution decreases by a factor of 2 in both the vertical and horizontal directions. This results in a down-scaled image with a resolution of 160 $\times$ 120, which is a 4-fold reduction from the original resolution. This down-scaled image is located in the second position of the image pyramid group.
- Therefore, in this example, images down-scaled up to the fourth step will belong to the image pyramid group. This is because the resolution of the image down-scaled to the fifth step is 20 $\times$ 15, which is smaller than the predefined window size of 20 $\times$ 20. Consequently, it cannot be utilized by the Viola-Jones classifier.

3.4. Window Generation

In window generation, the input window for images with a pyramidal architecture is created based on the predefined window size set during training. This input window facilitates the sliding operation in the classification process using the cascade classifier architecture, as the Viola-Jones classifier operates based on these predefined windows. Therefore, the input window can be generated using Equations (15) and (16).

$$W_{I.P} = I_P(x : x + (N-1), y : y + (M-1)) \tag{15}$$

$$W_{E.P} = E_P(x : x + (N-1), y : y + (M-1)) \tag{16}$$

where x and y represent the coordinates for the horizontal and vertical directions, $W_{I.P}$ is the window for the edge component-calibrated image, and $W_{E.P}$ is the window for the merged edge component image.

For instance, assuming a 20×20 predefined window size, with $x = 1$ and $y = 1$:

- The window for the edge component-calibrated image spans from $(1, 1)$ to $(20, 20)$, comprising a total of 400 pixel values.
- Similarly, for the merged edge component image, the window also ranges from $(1, 1)$ to $(20, 20)$, totaling 400 pixel values.
- Therefore, both types of windows, for the edge component-calibrated and merged edge component images, can be obtained.
- Subsequently, the sliding window operation can be performed by adjusting the coordinates along the horizontal or vertical directions.

3.5. Integral Image

After creating the two windows, the integral image is computed only for the window of the edge component-calibrated image. This is performed to improve processing speed by reducing the number of memory accesses and addition operations. In this paper, our proposed method utilizes a general integral image generation technique, given the software-based test environment [22]. In software environments, with the exception of hardware platforms like FPGA, the integral image can be generated using Equation (17).

$$II(x, y) = \sum_{x=1}^{N} \sum_{y=1}^{M} W_{I.P}(x, y) \tag{17}$$

By performing summation operations in both the horizontal and vertical directions for the window from the edge component-calibrated image, we can obtain the integral image where pixel values are accumulated diagonally.

In general, the general integral image generation technique, as depicted in Equation (17), reduces memory access when computing brightness differences for Haar-like features in the Viola-Jones classifier. However, it necessitates inefficient memory access due to the separable addition operation for each direction. To address this issue, an alternative approach, known as the pipeline architecture-based integral image generation technique, has been proposed to enhance processing speed and reduce latency while requiring similar hardware resources when implemented on an FPGA [23].

The advantage of utilizing the pipeline architecture-based integral image generation technique is that the process outlined in Equation (17), which is typically required to generate a new integral image for each window, is not necessary, except when generating the integral image for the window from the left reference. This efficiency is achieved by updating the integral image through the addition of newly updated pixel values (located in the new vertical line) to the pixel values of the previous generated integral image using the

sliding window technique, while discarding the pixel value that was received first (located in the frontmost vertical line). As a result of this process, the number of operations required to generate the integral image can be significantly reduced during the algorithm's execution.

Therefore, with an eye towards implementing the proposed algorithm on an FPGA platform in the future, we adopt an improved integral image generation technique for the integral image operation step.

3.6. Classification with Edge-Based Operation Skip Scheme

After generating the integral image, the two windows for the integral image II and the merged edge component image $W_{E.P}$ are utilized for classification based on the cascade classifier architecture. The classification process involves two operation steps: (1) extraction of reference edge component values and (2) classification based on the cascade classifier.

3.6.1. Reference Edge Component Value Extraction

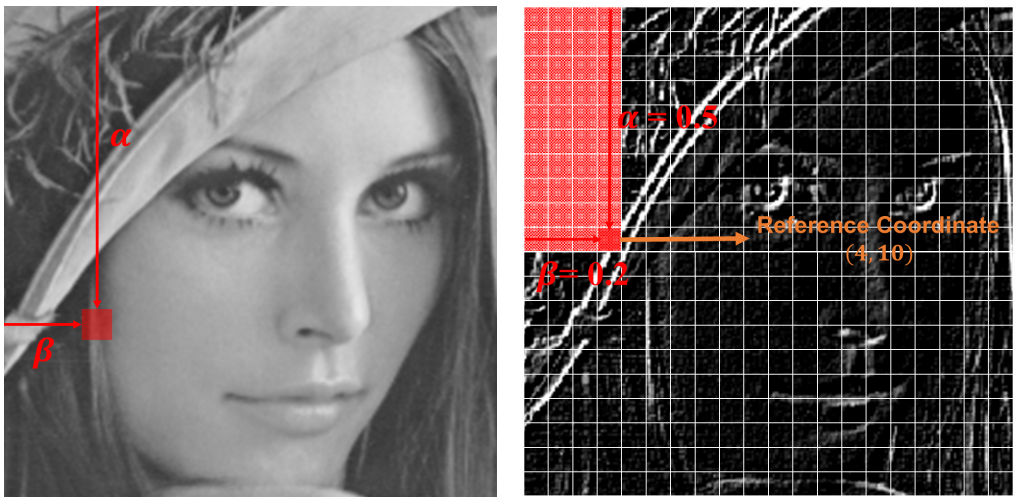
To improve processing speed, we propose an edge-based operation skip scheme. To utilize our proposed skip scheme, reference edge component values are required. The extraction of reference edge component values necessitates the merged edge component image and two hyper-parameters, α and β. The formula for the reference edge component value extraction process is as follows:

$$(x_R, y_R) = (N \times \alpha, M \times \beta) \tag{18}$$

$$E_R = W_{E.P}(x_R, y_R) \tag{19}$$

where α is the hyper-parameter for selecting the horizontal coordinate, β is the hyper-parameter for selecting the vertical coordinate, N and M are the input window size, x_R and y_R are the horizontal and vertical reference coordinates, respectively, and E_R is the extracted reference edge component value based on the reference coordinate (x_R, y_R).

For example, as shown in Figure 3, assuming a window size of 20×20 and hyper-parameters α and β set to 0.5 and 0.2, respectively, the reference coordinate is computed as $(10, 4)$. Therefore, the reference edge component value located at $(10, 4)$ in the window generated from the merged edge component image can be selected for the proposed edge-based operation skip scheme.



(**a**) (**b**)

Figure 3. Reference coordinate selection approach for the proposed edge-based operation skip scheme: (**a**) window for edge component-calibrated image and (**b**) window for merged edge component image.

3.6.2. Cascade Classifier-Based Classification

After selecting the reference edge component value, the extracted value and the pixel values of the window from the integral image II are entered into the cascade classifier to perform the classification function. The concept of classification with the proposed edge-based operation skip scheme is as follows:

$$D_O = \begin{cases} CascadeClassifier(II(x,y)) & \text{if } E_R > T_O \\ OperationSkip & \text{otherwise} \end{cases} \tag{20}$$

where $CascadeClassifier(\cdot)$ is the classification function using the cascade classifier architecture, T_O is the threshold value that determines whether or not to perform a classification function $CascadeClassifier(\cdot)$, $OperationSkip$ indicates that the classification operation is not performed, and D_O is the detection result containing width (w), height (h), coordinates x, and coordinates y, along with the scale factor (sf) at the step where the coordinates were detected.

As shown in Equation (20), the classification operation is not performed when the reference edge component value is less than the threshold value T_O. On the other hand, the classification operation is only performed when the reference edge component value is above the threshold value T_O. When the classification operation is performed, indicating that the reference edge component value is above the threshold value, the window from integral image II is entered into the cascade classifier architecture.

In the context of the classification operation within the cascade classifier architecture, the overall process is depicted in Figure 2, located within the classification box. When the input window at the corresponding coordinate successfully passes all the strong classifiers, each composed of various weak classifiers, the information required for the coordinate merging process is entered into the detection result group D_O. Conversely, when the corresponding coordinate fails to pass all the strong classifiers, the detection result group does not contain information for that coordinate, indicating that the region associated with that coordinate is a non-object region.

3.7. Coordinate Merging

After completing the classification operation for all pyramidal images, the detected coordinate groups, including width (w), height (h), x, y, and scale factor (sf) information, are used to compute the merged coordinates for drawing bounding boxes on the image. To accomplish this, we employ the Intersection over Union (IoU) for the merging process, as illustrated in Equations (21) and (22).

$$IoU(x,y) = \begin{cases} x & \text{if } IoU \geq T_I \\ (x,y) & \text{otherwise} \end{cases} \tag{21}$$

$$D'_O = IoU(D_O[a], D_O[b]) \tag{22}$$

where IoU denotes the function for the Intersection over Union (IoU) operation, T_I represents the threshold value for IoU operation, $D_O[a]$ denotes the detected coordinate information, $D_O[b]$ denotes the detected coordinate information used to compute the IoU area excluding $D_O[a]$, and D'_O represents the merged coordinate information.

As shown in Equation (21), when using the IoU function $IoU(x,y)$, two types of results can be obtained. First, when the IoU area value between the first and second input parameters exceeds the threshold value T_I, the IoU function computes the result using only the first input value. Otherwise, when the IoU area value is less than the threshold value

T_I, the IoU function computes the result using both the first and second input values. The threshold value for IoU can be set by the user. Generally, a threshold value of 0.5 is widely used not only for neighboring coordinate merging but also for performance evaluation based on ground truth [13,20,24]. Therefore, in our proposed method, we use 0.5 as the threshold value for IoU.

This merging operation based on IoU is performed separately for all detected coordinates. Consequently, if the IoU area result value exceeds 0.5 at least once, the number of merged coordinates becomes smaller than the number of coordinates before merging. Conversely, if the IoU area result value does not exceed 0.5, the number of merged coordinates remains the same as the number of coordinates before merging. (If only one coordinate is detected before merging, the merge operation is not performed, so the number of coordinates after merging remains the same).

4. Experimental Results

To compare the performances of the proposed and conventional methods, we conducted two types of comparisons: (1) a visual comparison and (2) a quantitative comparison. To ensure a fair performance comparison, we utilized the 'haarcascade_frontalface_alt.xml' file provided by the open computer vision library (OpenCV) library, as OpenCV offers the most fair trained parameters. In the XML file of OpenCV, the square window size and the number of strong classifier stages are 20 and 22, respectively.

Regarding our proposed method, we set the reference coordinate as $(10, 1)$, with the α and β values set to 0.5 and 0.05, respectively. The threshold value for the operation skip scheme was set to 0. As for the 2-D wavelet transform, we utilized the 2-D Haar wavelet transform, as previous works have also employed this technique [14,15].

For the objective performance comparison, we utilized four metrics, (1) precision, (2) recall, (3) F_1 score, and (4) number of classification iterations (NCIs). Concerning the NCIs, three parallelized weak classifiers are employed for each strong classifier in both the proposed and conventional methods. Regarding the other metrics (precision, recall, and F_1 score), they can be computed using Equations (23)–(25).

$$Precision = \frac{TP}{TP + FP} \tag{23}$$

$$Recall = \frac{TP}{TP + FN} \tag{24}$$

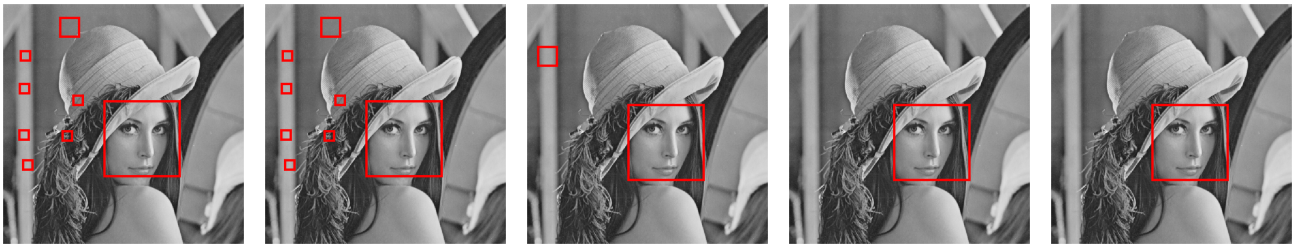
$$F_1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{25}$$

where TP denotes true positive, FP denotes false positive, and FN denotes false negative.

4.1. Visual Comparison

To conduct the visual comparison, we utilized two types of test images: (1) Lena and (2) Solvay Conference 1927.

Figure 4 displays the experimental results using the proposed and conventional methods for the 'Lena' test frame. In Figure 4a,b, when employing the Viola-Jones classifier and Hyun's method [20], numerous bounding boxes are detected in background regions (non-object regions). Conversely, when utilizing methods adopting the 2-D wavelet transform as shown in Figure 4c–e, improved visual detection performance is observed. Specifically, the number of detected bounding boxes in background regions is significantly reduced compared to Figure 4a,b. In other words, the utilization of 2-D wavelet transform-based methods results in a substantial reduction in false positives.



(**a**) (**b**) (**c**) (**d**) (**e**)

Figure 4. Experimental result using the proposed and conventional methods for 'Lena' test frame: (**a**) Viola-Jones classifier [11,12], (**b**) Hyun [20], (**c**) Choi [14], (**d**) our previous work [15], and (**e**) the proposed method (Red box: Detected result).

Table 1 shows the numerical performance comparison consisting of four metrics when using the proposed and conventional methods for the 'Lena' test frame. When employing both the proposed and conventional methods, the recall metric value is 1, indicating that the target object is perfectly detected. However, notable differences in numerical values are observed between the precision metric of the proposed method (including our previous work) and that of conventional methods (except for our previous work).

Table 1. Performance comparison of the proposed and conventional methods when using 'Lena' test frame.

Works	Precision	Recall	F_1	NCIs
[11,12]	0.1250	1.0000	0.2222	5,324,725
[20]	0.1250	1.0000	0.2222	5,317,377
[14]	0.5000	1.0000	0.6667	1,184,812
[15]	1.0000	1.0000	1.0000	812,910
Ours	1.0000	1.0000	1.0000	978,376

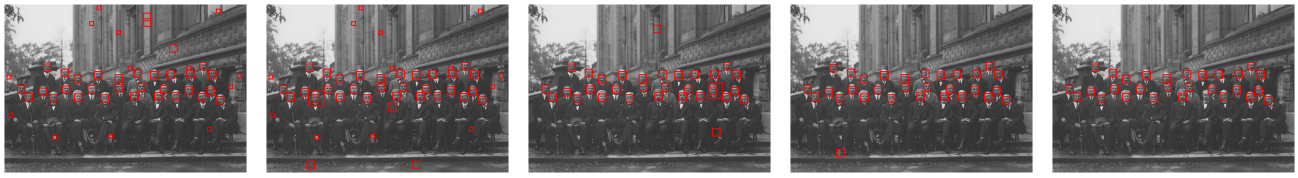
Specifically, when using the Viola-Jones classifier and Hyun's method [20], a precision metric value of 0.125 is observed. As depicted in Figure 4a,b, numerous false positive results in the background regions contribute to this difference between the precision and recall metrics. When utilizing Choi's method [14], a precision metric value of 0.5 is observed. In Figure 4c, only one false positive result appears in the background region. However, as this is a test frame with only one target object, the precision value is relatively lower compared to recall, at 0.5. Conversely, both our previous work and the current proposed method exhibit a precision metric value of 1. As shown in Figure 4d,e, no false positive results are present in the background regions, allowing precision to match the recall metric.

Based on the precision and recall metric values, the F_1 score can be computed using Equation (25). Excluding our previous work, conventional methods exhibit relatively low precision values, resulting in naturally low F_1 score values. Conversely, for our previous work and the method proposed in this paper, both precision and recall metric values are 1, leading to an F_1 score of 1 as well.

Regarding the NCIs metric, including our previous study, this exhibited values of about 5.325 M, 5.317 M, 1.185 M, and 0.813 M. In contrast, the NCIs metric when using the proposed method is 0.978 M. Comparing the proposed method in this paper with the conventional methods excluding our previous study, the NCI values of the proposed method are significantly reduced by 81.6258%, 81.6004%, and 17.4235%, respectively. However, when compared with our previous study, the NCIs metric value increased by about 20.3548%. Considering the four metric values for the 'Lena' test frame, it is evident that the detection performance can be excellent while requiring less time to detect objects com-

pared to conventional methods (excluding our previous study). This is because a smaller NCIs value means a shorter time required to detect objects (the proposed method requires only 978,376 clocks in the classification process, while the Viola-Jones classifier requires 5,324,725 clocks).

In the case of the 'Lena' test frame, there is an issue with the number of target objects, as there is only 1. Therefore, we used the 'Solvay Conference 1927' as the test frame for the second visibility test because the number of target objects in the image is large (number of target objects: 29). Figure 5 displays the experimental results using the proposed and conventional methods for the 'Solvay Conference 1927' test frame. In Figure 5a,b, when using the Viola-Jones classifier and Hyun's method [20], there are many false positives on background regions, similar to the experimental results using 'Lena' as shown in Figure 4a,b. Conversely, when using the methods adopting the 2-D wavelet transform as shown in Figure 5c–e, they exhibited better visual performance on the background regions, meaning the false positives were significantly reduced compared with Figure 5a,b, similar to the experimental results of Figure 4c–e.



(**a**) (**b**) (**c**) (**d**) (**e**)

Figure 5. Experimental result using the proposed and conventional methods for 'Solvay Conference 1927' test frame: (**a**) Viola-Jones classifier [11,12], (**b**) Hyun [20], (**c**) Choi [14], (**d**) our previous work [15], and (**e**) proposed method (Red box: Detected result).

Table 2 presents the numerical performance comparison when using the proposed method and conventional methods for the 'Solvay Conference 1927' test frame. When utilizing the conventional methods, the recall metric value is 1, consistent with the experimental results in Table 1. Conversely, when using the proposed method, the recall metric value is 0.9655, which is lower than that of the conventional methods. This indicates that the proposed method detects only 28 out of 29 target objects, resulting in lower performance compared to the conventional methods.

Table 2. Performance comparison of the proposed and conventional methods when using 'Solvay Conference 1927' test frame.

Works	Precision	Recall	F_1	NCIs
[11,12]	0.6444	1.0000	0.7838	25,200,943
[20]	0.6304	1.0000	0.7733	25,076,126
[14]	0.9355	1.0000	0.9667	5,655,980
[15]	0.9667	1.0000	0.9831	3,582,521
Ours	1.0000	0.9655	0.9825	4,114,420

However, in terms of the precision metric, the proposed method achieves a value of 1. This is due to the absence of false positives when using the proposed method, as opposed to the experimental results using the conventional methods. Therefore, considering the F_1 score, which comprehensively assesses precision and recall metric values, it can be inferred that the proposed method exhibits superior performance compared to the conventional methods (excluding our previous study).

In terms of the NCIs metric, the values observed, including our previous study, were approximately 25.20 M, 25.08 M, 5.66 M, and 3.58 M. When utilizing the proposed

method, the NCIs metric is 4.11 M. Comparing the proposed method with conventional methods, excluding our previous study, revealed significant reductions in the NCIs metric values, amounting to 83.6736%, 83.5923%, and 27.2554%, respectively. Nonetheless, when compared to our previous study, the NCIs metric increased by approximately 14.8471%. An analysis of the experimental results for the 'Solvay Conference 1927' test frame indicates that the proposed method enhances processing speed while maintaining detection performance, as reflected in the F_1 score, excluding our previous study.

4.2. Quantitative Comparison

In the Visual Comparison sub-section, we observed the experimental results for two test frames, 'Lena' and 'Solvay Conference 1927'. However, there is an issue that the performance of the proposed method may only show good results for these two test frames. Therefore, to perform a quantitatively fair and object performance comparison, we utilized the FDDB public dataset. The FDDB public dataset consists of a total of 2845 test images, with 5171 target objects. Table 3 presents the precision, recall, F_1 score, and mean NCI values when applying the proposed method and conventional methods to the FDDB public dataset (IoU threshold to 0.5).

Table 3. Performance comparison of the proposed and conventional methods when using FDDB public dataset with IoU threshold as 0.5.

Works	Precision	Recall	F_1	NCIs
[11,12]	0.5425	0.5364	0.5394	2,891,828
[20]	0.5423	0.5362	0.5393	2,885,842
[14]	0.5512	0.5316	0.5412	608,839
[15]	0.5233	0.4879	0.5050	428,473
Ours	0.5519	0.5283	0.5398	458,969

As illustrated in Table 3, the precision values for the conventional methods are 0.5425, 0.5423, 0.5512, and 0.5233, respectively, when using a threshold value of 0.5 for IoU. In contrast, the proposed method achieves a precision of 0.5519, outperforming all other methods under the same IoU threshold. Numerically, the proposed method improves precision by 1.7327%, 1.7702%, 0.1270%, and 5.4653% compared to the conventional methods.

For the recall metric, the conventional methods yield values of 0.5364, 0.5362, 0.5316, and 0.4879, respectively, when using a 0.5 IoU threshold. However, the proposed method shows a recall value of 0.5283, ranking second lowest among all methods using the same threshold. Numerically, the proposed method exhibits a reduction in recall by 1.5101%, 1.4733%, and 0.6208% compared to the conventional methods (excluding our previous study), with decreases of 0.0081%p, 0.0079%p, and 0.0033%p, respectively. Yet, compared to our previous study, the proposed method demonstrates an improvement of 8.2804%.

Considering both precision and recall, the proposed method significantly reduces false positives detected in the background, showcasing the best precision. However, there is a slight reduction in the object detection success rate compared to conventional methods (excluding our previous study). Nevertheless, compared to our previous study, there is a significant improvement in the recall metric, indicating that the proposed method overcomes previous limitations.

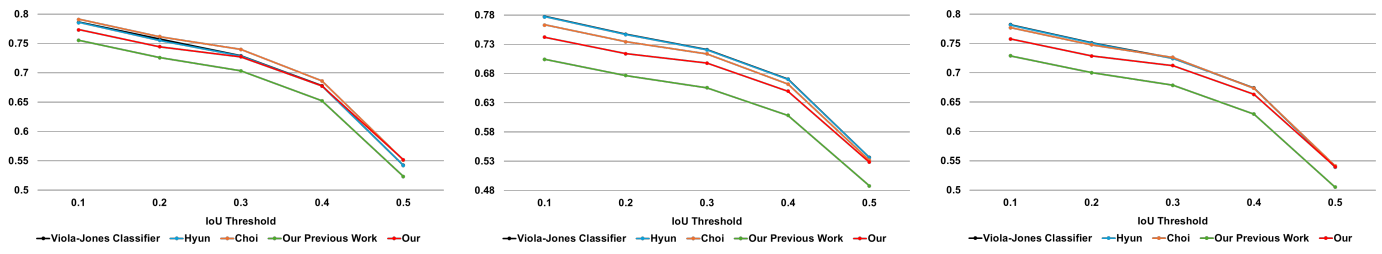
In terms of the F_1 score metric, the conventional methods yield values of 0.5394, 0.5393, 0.5412, and 0.5050, respectively, when using a 0.5 IoU threshold. In contrast, the proposed method achieves a value of 0.5398, ranking among the top two methods under the same threshold. Given the varying priorities of precision and recall, the proposed method's performance presents a viable option for various applications.

Regarding the mean NCIs, the conventional methods exhibit values of 2.892 M, 2.886 M, 0.608 M, and 0.428 M, respectively, while the proposed method achieves a value of 0.459 M, ranking second best. Numerically, the proposed method shows improvements of 84.1288%, 84.0958%, and 24.6157% compared to the conventional methods (excluding our previous study). However, compared to our previous study, the proposed method has a higher mean NCIs value by 7.1174%.

There may be an issue where the detection performance of the proposed method shows good performance only when the IoU threshold is 0.5. Therefore, we conducted an experiment to compare the detection performance between the proposed method and conventional methods as the IoU threshold was changed. Table 4 presents the results of the precision, recall, and F_1 score when the IoU threshold was varied from 0.1 to 0.5 for the FDDB dataset. The reason why Table 4 does not show values exceeding 0.5 is because the performance of the precision, recall, and F_1 scores of both the conventional methods and the proposed method approaches 0 exponentially. Figure 6 graphically represents the trends of the precision, recall, and F_1 score values from Table 4.

Table 4. Experimental result using the proposed and conventional methods for FDDB public dataset with various IoU threshold values.

IoU Threshold Value	Method	Precision	Recall	F_1 Score
0.1	Viola-Jones classifier [11,12]	0.7866	0.7778	0.7822
	[20]	0.7862	0.7774	0.7818
	[14]	0.7916	0.7635	0.7773
	[15]	0.7554	0.7403	0.7290
	Proposed	0.7738	0.7422	0.7577
0.2	Viola-Jones classifier [11,12]	0.7557	0.7472	0.7515
	[20]	0.7551	0.7467	0.7509
	[14]	0.7616	0.7345	0.7478
	[15]	0.7258	0.6767	0.7004
	Proposed	0.7444	0.7140	0.7289
0.3	Viola-Jones classifier [11,12]	0.7291	0.7210	0.7250
	[20]	0.7286	0.7204	0.7244
	[14]	0.7399	0.7136	0.7265
	[15]	0.7034	0.6558	0.6788
	Proposed	0.7274	0.6978	0.7123
0.4	Viola-Jones classifier [11,12]	0.6781	0.6705	0.6743
	[20]	0.6777	0.6701	0.6739
	[14]	0.6862	0.6618	0.6738
	[15]	0.6524	0.6083	0.6296
	Proposed	0.6775	0.6499	0.6634



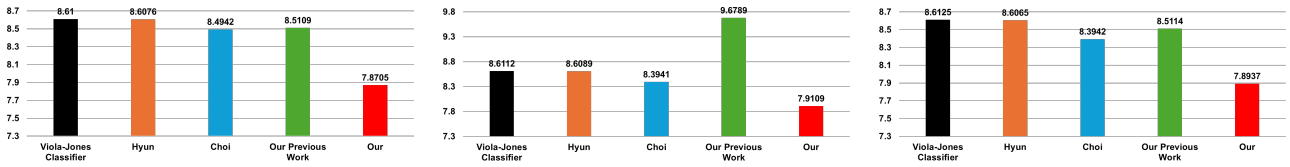
(**a**) (**b**) (**c**)

Figure 6. Experimental result using the proposed and conventional methods for FDDB public dataset with various IoU threshold values: (**a**) precision, (**b**) recall, and (**c**) F_1 score.

As shown in Figure 6 and Table 4, the performances of the precision, recall, and F_1 score of the proposed method and the conventional methods improve as the IoU threshold for evaluation with the ground truth decreases (closer to 0.1). Conversely, the performances of the precision, recall, and F1 score of the proposed method and the conventional methods approach 0 as the IoU threshold increases.

What we should pay attention to here is how much the performance of the precision, recall, and F_1 score of the proposed method differs from the conventional method for each IoU threshold. Unlike when the IoU threshold is 0.5, the performance of the precision and F_1 score of the proposed method is not ranked high when the IoU threshold is between 0.1. and 0.4. However, Figure 6 shows that the proposed method has a similar tendency to the conventional methods, except for our previous study. Compared to the method proposed in our previous study, the method proposed in this paper can be confirmed to show superior performance in all IoU threshold values.

As a result, it is true that the performance of the proposed method can show the best performance relative to the conventional methods when the IoU threshold is 0.5. However, it is necessary to confirm how much the performance decreases depending on the IoU threshold and whether it can show uniform performance. Therefore, we calculated the mean performance decrease depending on the IoU threshold, and as shown in Figure 7, it demonstrates the mean performance decrease as the IoU threshold changes.



(**a**) (**b**) (**c**)

Figure 7. Mean performance degradation of proposed and conventional methods using the FDDB public dataset at various IoU threshold values: (**a**) precision, (**b**) recall, and (**c**) F_1 score.

First, in the case of the Viola-Jones classifier, the performances of the precision, recall, and F_1 score metrics decrease by a mean of 8.6100%, 8.6112%, and 8.6125%, respectively, when the IoU increases by 0.1. Second, in the case of Hyun's method [20], when the IoU increases by 0.1, the performances of the precision, recall, and F_1 score metrics decrease by a mean of 8.6076%, 8.6089%, and 8.6065%, respectively. Third, in the case of Choi's method [14], when the IoU increases by 0.1, the performances of the precision, recall, and F_1 score metrics decrease by a mean of 8.4942%, 8.3941%, and 8.3942%, respectively. Fourth, in the case of the method proposed in our previous study [15], when the IoU increases by 0.1, the performances of the precision, recall, and F_1 score metrics decrease by a mean of 8.5109%, 9.6789%, and 8.5114%, respectively. Finally, in the case of the proposed method, the performances of the precision, recall, and F_1 score decrease by a mean of 7.8705%, 7.9109%, and 7.8937%, respectively.

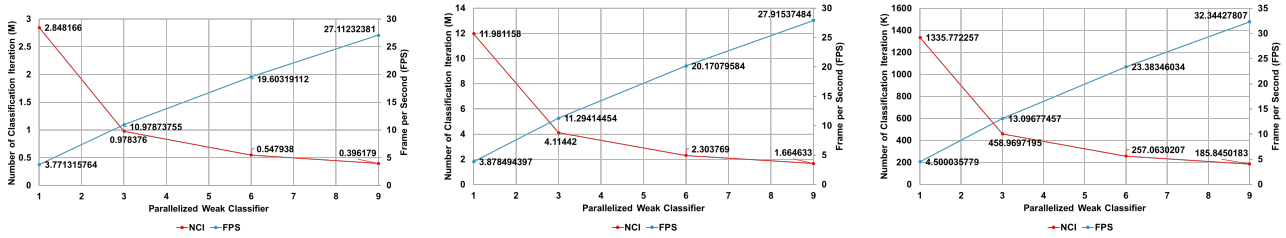
The reason why conventional methods (excluding our previous study) show higher performance when the IoU threshold is lower is as follows: Conventional methods generally perform object classification operations across all areas of the image. As a result, multiple bounding boxes are generated before the merging process. This means that bounding boxes which do not exceed the IoU threshold are not merged during the merging process, leading to multiple detection results for the same object area (i.e., two or more bounding boxes may be displayed for the same object after merging). Therefore, even after the merging process, un-merged bounding boxes remain, which results in relatively higher precision and recall values compared to the proposed methods. On the other hand, the proposed method performs object classification tasks only in regions that are more likely to be estimated

as objects compared to conventional methods. By avoiding unnecessary operations in non-object regions, the NCIs can be reduced, along with a decrease in the number of false positives. However, a relatively smaller number of bounding boxes are generated before the merging process. Consequently, the number of un-merged bounding boxes for the same object region is smaller than that of conventional methods (i.e., only one bounding box is displayed for the visually identical object after merging). Therefore, when the IoU threshold is lower, the performance of the proposed method tends to be lower compared to conventional methods.

It is true that the proposed method showed the best value compared to the conventional methods only when the IoU threshold was 0.5. However, when looking at the tendency and value of performance decreasing as the IoU threshold changed, it means that the proposed method can show relatively uniform detection performance compared to the conventional methods.

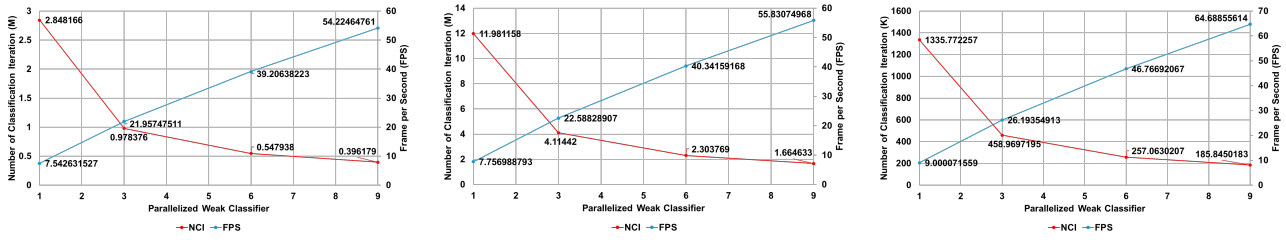
4.3. Processing Speed Estimation

The method proposed in this paper is designed considering system implementation in an FPGA, which is a preliminary step for ASIC design. Therefore, in future work, before implementing the proposed method on an FPGA, it is necessary to estimate the minimum processing speed required in the classification operation step. To achieve this, we derived the processing speed according to the number of parallelized weak classifiers for two test frames and the FDDB public dataset, as shown in Figures 8 and 9.



(**a**) (**b**) (**c**)

Figure 8. Experimental results using the proposed method with an operating frequency of 30 frames per second: (**a**) Lena, (**b**) Solvay conference 1927, and (**c**) FDDB public dataset.



(**a**) (**b**) (**c**)

Figure 9. Experimental results using the proposed method with an operating frequency of 60 frames per second: (**a**) Lena, (**b**) Solvay conference 1927, and (**c**) FDDB public dataset.

Parallelized weak classifiers refer to the extent to which weak classifiers utilized in the strong classifier stage, as shown in Figure 1, are parallelized and employed. For example, a parallelized weak classifier value of 1 indicates that each strong classifier stage will use the weak classifier only once per clock cycle. A parallelized weak classifier value of 3, 6, or 9 means that each strong classifier stage will utilize 3, 6, or 9 weak classifiers per clock cycle, respectively. The reason for setting the number of parallelized weak classifiers in this

experiment to 1, 3, 6, and 9 is to evaluate how processing speed performance improves with an increasing number of parallelized weak classifiers.

To obtain the minimum processing time required for the classification operation step, two factors are essential: (1) the NCIs and (2) the operating frequency. However, there is an issue where the image resolution of the two test frames ('Lena' and 'Solvay Conference 1927') and the FDDB public dataset inferred does not adhere to the standard specification. Therefore, we calculate the minimum processing time using an approximate operating frequency. The formula for obtaining the operating frequency is as follows:

$$f_r = \begin{cases} 12.5875 \times 10^6 & \text{where } r = 30, \\ 25.1750 \times 10^6 & \text{where } r = 60 \end{cases} \tag{26}$$

$$f_{op} = f_r \times \frac{R_{Target}}{(640 \times 480)} \tag{27}$$

where f_r is the reference operating frequency based on VGA (Video Graphic Array, 640 × 480 resolution), r represents the target FPS, R_{Target} represents the target resolution (e.g., 'Lena' has 512 × 512), and f_{op} represents the operating frequency for the target image.

For the 'Lena' test frame, the resolution is 512 × 512. Therefore, when using the above formula, operating frequencies of approximately 10.7413 MHz and 21.4827 MHz are required to receive input images at 30 FPS and 60 FPS, respectively. When using 10.7413 MHz as the operating frequency, as shown in Figure 8a, the classification operation shows a processing performance of approximately 27.11 FPS when using nine parallelized weak classifiers. When using 21.4827 MHz as the operating frequency, as shown in Figure 9a, the classification operation shows a processing performance of approximately 54.22 FPS when using nine parallelized weak classifiers.

For the 'Solvay Conference 1927' test frame, the resolution is 1280 × 886. Therefore, when using the above formula, operating frequencies of approximately 46.4689 MHz and 92.9377 MHz are required to receive input images at 30 FPS and 60 FPS, respectively. When using 46.4689 MHz as the operating frequency, as shown in Figure 8b, the classification operation shows a processing performance of approximately 27.92 FPS when using nine parallelized weak classifiers. When using 92.9377 MHz as the operating frequency, as shown in Figure 9b, the classification operation shows a processing performance of approximately 55.83 FPS when using nine parallelized weak classifiers.

For the FDDB public dataset, the resolution ranges from a minimum of 171 × 449 to a maximum of 450 × 450. Among this resolution range, the median resolution is 450 × 325, so we derived the operating frequency based on the 450 × 325 resolution. Therefore, when using the formula, operating frequencies of about 5.9926 MHz and 11.9852 MHz are required to receive input images at 30 FPS and 60 FPS, respectively. When using 5.9926 MHz as the operating frequency, as shown in Figure 8c, the classification operation shows a processing performance of approximately 32.34 FPS when using nine parallelized weak classifiers. When using 11.9852 MHz as the operating frequency, as shown in Figure 9c, the classification operation shows a processing performance of approximately 64.69 FPS when using nine parallelized weak classifiers.

In the case of the 'Lena' and 'Solvay Conference 1927' test frames, it is evident that the processing performance of 30 FPS or 60 FPS is not achieved at the given operating frequencies even with nine parallelized weak classifiers. On the other hand, for the FDDB public dataset, it was confirmed that the processing speed performance of 30 and 60 FPS can be achieved at the given operating frequencies when nine parallelized weak classifiers are used in the classification operation step.

However, when designing using the FPGA platform, the processing performance might be somewhat reduced due to the image pyramid generator and coordinate merger. The operating frequency value used in the processing speed estimation experiment can vary depending on the interval between line valid signals, potentially improving FPS performance in the future. Additionally, when designing using the FPGA platform, if the digital logic circuit is optimized considering the critical path, multiple operations can be performed within one clock, thus enhancing FPS performance.

The processing speed estimation experiment represents the worst-case scenario (only one operation per clock) in the classification operation when supposing the proposed method is implemented in an FPGA. If the proposed method is designed to overcome the worst-case scenario when implemented in a digital logic circuit, it can achieve real-time processing. Therefore, the key takeaway from this processing speed estimation experiment is that the proposed method can demonstrate near-real-time performance when using nine parallelized weak classifiers in the classification operation step, which is advantageous compared to the NCI metric values of the conventional methods.

5. Discussion

In the experimental environment using the FDDB public dataset and an IoU threshold of 0.5, it was confirmed that the proposed method reduces the NCIs by a minimum of 24.62% and a maximum of 84.13% compared to conventional methods (excluding the method proposed in our previous work). Compared to the method in our previous work, the NCIs result of the proposed method increased by 7.12%. Although the NCIs increased slightly compared to the previous work, the proposed method significantly reduces the NCIs compared to other conventional methods.

In addition, the experimental results using parallelized weak classifiers indicate that the proposed method cannot achieve real-time processing when fewer than nine parallelized weak classifiers are used. For instance, when images from the FDDB public dataset are input at 30 FPS and 60 FPS, the processing performance reaches up to 23.38 FPS at 5.9926 MHz and 46.76 FPS at 11.9852 MHz when six parallelized weak classifiers are utilized. However, near-real-time processing becomes feasible with nine parallelized weak classifiers, achieving 32.34 FPS at 5.9926 MHz and 64.69 FPS at 11.9852 MHz. These results confirm that the proposed method can effectively overcome the limitations of conventional Viola-Jones classifier-based methods, enabling real-time processing when at least nine parallelized weak classifiers are employed.

One of the most critical considerations when implementing a Viola-Jones classifier-based object detection system with nine parallelized weak classifiers on an FPGA is the amount of hardware resources utilized. If the edge compensation operator structure proposed in previous studies is incorporated into the Viola-Jones classification system before implementing it on an FPGA-based system, it is anticipated that the system will require one block memory (with a maximum size of 36 K) for window generation. Additionally, there will be a slight increase in slice register and look-up table (LUT) usage to perform the compensation operation, leading to a minor increase in power consumption. Regarding the classification process, it is natural for hardware resource consumption to increase as the number of parallelized weak classifiers rises. However, if the weak classifier structure from prior studies is employed, significant differences in hardware resource consumption are not expected. This is because the classifier IP, composed solely of an address selector and a coefficient comparator, is reused [20,25]. Therefore, while implementing the proposed method as an FPGA-based system may lead to a slight increase in hardware resource usage and power consumption compared to existing FPGA implementations, the trade-offs are

considered acceptable. The improvements in processing speed and overall performance outweigh these disadvantages, making the implementation worthwhile.

In experiments using the FDDB public dataset, when the IoU threshold ranged from 0.1 to 0.4, the precision of the proposed method decreased by 1.27% to 2.23% compared to the best-performing conventional methods. However, at an IoU threshold of 0.5, the precision improved by 1.73% compared to the best conventional methods. For the recall metric, the proposed method exhibited a decrease of 0.83% to 4.58% when the IoU threshold ranged from 0.1 to 0.5. Regarding the F_1 score, the proposed method showed a reduction of 0.03% to 3.13% compared to the best-performing conventional methods across the same IoU threshold range.

The experimental results of applying threshold values incrementally indicate a slight reduction in detection performance for the proposed method. However, when combined with the NCIs performance, this level of degradation is acceptable given the significant improvement in processing speed. Notably, the detection performance of the proposed method is substantially higher than that of our previous work. This finding confirms that the edge calibration method used in our previous work adversely affected detection performance. The edge calibration method proposed in this paper, however, minimizes its impact on detection performance.

Finally, the results suggest the necessity of further experiments on various algorithms used in merging bounding boxes. The performance differences observed depend significantly on the IoU threshold values, in addition to the algorithmic differences between the proposed and conventional methods.

6. Conclusions

In this paper, we propose an enhanced edge component correction method and an edge-based operation skipping scheme for the cascade classifier architecture, commonly known as the Viola-Jones classifier. By leveraging the improved edge component correction concept initially introduced by Choi [14], we achieved competitive detection accuracy compared to the conventional methods. Additionally, by adopting the proposed edge-based operation skipping scheme, we significantly reduce the number of computations involved compared to other conventional methods. The proposed edge-based operation skipping scheme significantly reduces the computational workloads compared to conventional methods, offering notable throughput improvement. Considering the FPGA implementation, we validated the feasibility of the near real-time operation across nine parallelized weak classifiers using test frames consisting of 'Lena' and 'Solvay Conference 1927', and the FDDB public dataset.

Based on the experimental results presented in this paper, future work will include the following experiments: (1) Analysis of average precision (AP) and receiver operating characteristic (ROC) curves, among others. (2) Performing a comparative analysis of classification performance based on hyper-parameters and threshold values for edge component extraction. (3) Building a dataset for CPD using a long-wave infrared (LWIR)-based thermal imaging system, and verifying that it meets the Euro NCAP CPD standard. (4) Implementing an object detection system using a register-transfer level (RTL)-based accelerator for real-time operation across various input image resolutions, and comparing the real-world performance in terms of static and dynamic power consumption, as well as hardware resource utilization.

Author Contributions: Conceptualization, C.-H.C. and H.W.O.; methodology, C.-H.C., J.H., and H.W.O.; software, C.-H.C.; validation, C.-H.C., J.C., and J.S.; formal analysis, C.-H.C., J.C., and J.S.; investigation, C.-H.C. and J.H.; data curation, J.H., H.W.O., and J.C.; writing—original draft preparation, C.-H.C.; writing—review and editing, C.-H.C., J.H., H.W.O., J.C., and J.S.; visualization,

C.-H.C. and J.H.; supervision, J.H. and J.C.; project administration, J.H. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data sharing is not applicable due to data security legal policy related to the Korean military industry.

Conflicts of Interest: All authors were employed by the company Hanwha Systems, Co., Ltd. However, this affiliation does not pose any other conflicts of interest regarding the research presented in this paper.

References

1. Makkena, Y.C.; Prashanth, P.; Tammana, P.; Chandrahas, P.; Pachamuthu, R. Real-Time Object Detection as a Service for UGVs Using Edge Cloud. In Proceedings of the 2024 16th International Conference on COMmunication Systems & NETworkS (COMSNETS), Bengaluru, India, 3–7 January 2024; pp. 303–305.
2. Choi, C.H.; Hong, S.; Jeong, E.J.; Han, J. Infrared Thermal Imaging for Embedded Child Presence Detection System: Feasibility and Performance Evaluation. In Proceedings of the 2024 IEEE International Conference on Consumer Electronics-Asia (ICCE-Asia), Danang, Vietnam, 3–6 November 2024; IEEE: Piscataway, NJ, USA, 2024; pp. 1–4.
3. Jaworek-Korjakowska, J.; Kostuch, A.; Skruch, P. SafeSO: Interpretable and explainable deep learning approach for seat occupancy classification in vehicle interior. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 19–25 June 2021; pp. 103–112.
4. Erlik Nowruzi, F.; El Ahmar, W.A.; Laganiere, R.; Ghods, A.H. In-vehicle occupancy detection with convolutional networks on thermal images. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, Long Beach, CA, USA, 15–20 June 2019; pp. 941–948.
5. Zeng, X.; Wang, F.; Wang, B.; Wu, C.; Liu, K.R.; Au, O.C. In-vehicle sensing for smart cars. *IEEE Open J. Veh. Technol.* **2022**, *3*, 221–242. [CrossRef]
6. Kok, C.L.; Ho, C.K.; Tan, F.K.; Koh, Y.Y. Machine learning-based feature extraction and classification of emg signals for intuitive prosthetic control. *Appl. Sci.* **2024**, *14*, 5784. [CrossRef]
7. Sak, H.; Senior, A.; Rao, K.; Beaufays, F. Fast and accurate recurrent neural network acoustic models for speech recognition. *arXiv* **2015**, arXiv:1507.06947.
8. Diwan, T.; Anirudh, G.; Tembhurne, J.V. Object detection using YOLO: Challenges, architectural successors, datasets and applications. *Multimed. Tools Appl.* **2023**, *82*, 9243–9275. [CrossRef] [PubMed]
9. Lai, S.; Xu, L.; Liu, K.; Zhao, J. Recurrent convolutional neural networks for text classification. In Proceedings of the AAAI Conference on Artificial Intelligence, Austin, TX, USA, 25–30 January 2015; Volume 29.
10. Liang, M.; Hu, X. Recurrent convolutional neural network for object recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 3367–3375.
11. Viola, P.; Jones, M. Rapid object detection using a boosted cascade of simple features. In Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2001, Kauai, HI, USA, 8–14 December 2001; Volume 1, pp. 511–518.
12. Viola, P.; Jones, M.J. Robust real-time face detection. *Int. J. Comput. Vis.* **2004**, *57*, 137–154. [CrossRef]
13. Khong, W.L.; Moung, E.G.; Chong, C.S. Viola-Jones Method for Robot Vision Purpose: A Software Technical Review. In Proceedings of the 2nd International Congress of Electrical and Computer Engineering, ICECENG 2023, Bandirma, Turkey, 22–25 November 2023; pp. 45–61.
14. Choi, C.H.; Kim, J.; Hyun, J.; Kim, Y.; Moon, B. Face detection using haar cascade classifiers based on vertical component calibration. *Hum.-Centric Comput. Inform. Sci.* **2022**, *12*, 1–17.
15. Choi, C.H.; Han, J.; Cha, J.; Shin, J.; Oh, H.W. Fast Object Detection Algorithm using Edge-based Operation Skip Scheme with Viola-Jones Method. In Proceedings of the 2024 IEEE 6th International Conference on AI Circuits and Systems (AICAS), Abu Dhabi, United Arab Emirates, 22–25 April 2024.
16. Jain, V.; Learned-Miller, E. FDDB: A benchmark for face detection in unconstrained settings. *Tech. Rep. Umass Amherst Tech. Rep.* **2022**, *2*, 1–11.

17. Mimboro, P.; Heryadi, Y.; Suparta, W.; Wibowo, A. Realtime Vehicle Counting Method Using Haar Cascade Classifier Model. In Proceedings of the 2021 International Conference on Advanced Mechatronics, Intelligent Manufacture and Industrial Automation (ICAMIMIA), Surabaya, Indonesia, 8–9 December 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 229–233.
18. Ashwini, B.; Yuvaraju, B.; Pai, A.Y.; Baliga, B.A. Real time detection and classification of vehicles and pedestrians using haar cascade classifier with background subtraction. In Proceedings of the 2017 2nd International Conference on Computational Systems and Information Technology for Sustainable Solution (CSITSS), Bengaluru, India, 21–23 December 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 1–5.
19. Aguilar, W.G.; Luna, M.A.; Moya, J.F.; Abad, V.; Ruiz, H.; Parra, H.; Angulo, C. Cascade classifiers based robust pedestrian detection. In Proceedings of the 2017 1st Latin American Conference on Intelligent Transportation Systems (ITS LATAM), Bogota, Colombia, 29 March 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 1–6.
20. Hyun, J.; Kim, J.; Choi, C.H.; Moon, B. Hardware architecture of a Haar classifier based face detection system using a skip scheme. In Proceedings of the 2021 IEEE International Symposium on Circuits and Systems (ISCAS), Daegu, Republic of Korea, 22–28 May 2021; pp. 1–4.
21. Choi, C.H.; Kim, Y.; Ha, J.; Moon, B. Haar Filter Hardware Architecture for the Accuracy Improvement of Stereo Vision Systems. In Proceedings of the 2021 18th International SoC Design Conference (ISOCC), Jeju Island, Republic of Korea, 6–9 October 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 401–402.
22. Kim, D.; Hyun, J.; Moon, B. Memory-efficient architecture for contrast enhancement and integral image computation. In Proceedings of the 2020 International Conference on Electronics, Information, and Communication (ICEIC), Barcelona, Spain, 19–22 January 2020; pp. 1–4.
23. Kim, J.; Hyun, J.; Moon, B. Low-cost Hardware Architecture for Integral Image Generation using Word Length Reduction. In Proceedings of the 2020 International SoC Design Conference (ISOCC), Yeosu, Republic of Korea, 21–24 October 2020; pp. 119–120.
24. Rezatofighi, H.; Tsoi, N.; Gwak, J.; Sadeghian, A.; Reid, I.; Savarese, S. Generalized intersection over union: A metric and a loss for bounding box regression. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–20 June 2019; pp. 658–666.
25. Cho, J.; Mirzaei, S.; Oberg, J.; Kastner, R. Fpga-based face detection system using haar classifiers. In Proceedings of the ACM/SIGDA International Symposium on Field Programmable Gate Arrays, Monterey, CA, USA, 22–24 February 2009; pp. 103–112.