Edge Detection Using Sobel Algorithm and YCbCr Colour Space Optimized on FPGA

Yang An*, Qianqian Yuan, Han Zhang
School of Science, Jiaozuo Normal College, Jiaozuo 454002, China
E-mail: jzszay@126.com, yuan-qianqian@vip.163.com, kxmytzxh@163.com
*Corresponding author

Keywords: edge detection, FPGA, YcbCr, sobel algorithm

Received: August 26, 2024

Edge detection plays a crucial role in image processing and computer vision, and is widely used in tasks such as object recognition and image segmentation. Traditional edge detection algorithms perform well in many applications, but there are still some shortcomings in terms of real-time performance and processing efficiency. To address this issue, a highly efficient image edge detection model combining Sobel algorithm and field programmable gate array technology was proposed. YCbCr color space conversion was performed on the image, then Sobel operator was utilized to calculate the image gradient, and adaptive thresholding method was applied to determine the edges. Finally, the model was implemented and optimized on a field programmable gate array. The experimental results showed that when the dataset size was 1000, the information retention rate of the proposed image preprocessing model was 0.89, and the structural information loss was 0.05. When the data volume was 100, the accuracy of the proposed image edge detection model was 0.90, and the root mean square error value was 0.16. The research results indicate that the proposed image edge detection model based on field programmable gate arrays has significant advantages in edge detection performance and processing efficiency. The model has high accuracy and speed in image edge recognition, which can provide certain guidance for research in the field of image edge detection.

Povzetek: Razvito je optimizirano zaznavanje robov slik s Sobelovim algoritmom v YCbCr barvnem prostoru na FPGA. Metoda izboljšujekvaliteto obdelave, kar omogoča hitrejšo in učinkovitejšo analizo slik.

1 Introduction

Image edge detection takes a critical part in the areas of computer vision and image processing, and is broadly applied in tasks such as pattern recognition [1]. The edge detection's main goal is to identify areas in an image with discontinuous grayscale or significant gradient changes, which typically correspond to important structural information in the image. Traditional edge detection algorithms perform well in many applications, but there are still some shortcomings in terms of real-time performance and processing efficiency. As a result of the accelerated advancement of artificial intelligence and computer vision technology, there has been a notable rise in the demand for image processing solutions in the field of image detection, especially in scenarios that require efficient and real-time processing, such as autonomous driving, intelligent monitoring, and medical image analysis. Traditional edge detection methods are inadequate [2]. In recent years, with the advancement of hardware technology, field programmable gate arrays (FPGAs) have gradually become an important direction in image processing research due to their high parallel processing capability and low power consumption characteristics. The advantage of FPGA lies in its flexible hardware programming ability and high parallel processing capability, which enables it to meet the needs of different application scenarios while maintaining high performance. Therefore, innovative research has proposed an efficient image edge recognition model that combines edge detection algorithms and FPGA technology. By performing YCbCr color space conversion on the image, the effect of edge detection is enhanced. Then, the Sobel operator is applied to calculate the image gradient, and the edge is determined through adaptive thresholding. Finally, the model is implemented and optimized on FPGA to improve processing efficiency. The research content contains four parts. Part 1 is a review of other scholars' relevant research topics. Part 2 is a simple introduction of the main methods utilized in this paper. Part 3 is the model findings acquired by using the methods to the research and analyzing the findings. Part 4 is a summary of the study and prospects for future research.

2 Related works

Image edge detection is a technique that is employed extensively in a multitude of tasks, including object recognition, image segmentation, and pattern recognition. To further optimize the performance of the large law threshold method, Yang et al. proposed a threshold deviation adjustment strategy based on the analysis of the relationship between pixel grayscale values and cumulative pixel changes. The experimental findings denoted that this strategy could better segment test images and obtain competitive misclassification errors [3]. Zheng et al. found that traditional particle swarm optimization (PSO) algorithm has a slow convergence speed in image

segmentation and is prone to getting stuck in local optima. Therefore, the research team proposed an improved particle swarm algorithm that combines the large law threshold method. The experiment finding indicated that the execution time of this algorithm was, on average, 30% faster than that of other algorithms, and its accuracy was superior to that of other algorithms [4]. Vite-Chávez et al. pointed out that significant challenges were encountered nal methods [5]. Chen et al. proposed an adaptive fractional order genetic classification method by combining PSO algorithm to improve the performance of the large law threshold method in image segmentation. The experimental outcomes indicated that the test results of this method in terms of regional contrast and peak signal-to-noise ratio (SNR) indicators, both qualitatively and quantitatively, were superior to the algorithm before improvement [6].

Gtifa et al. proposed a novel hardware architecture to improve the accuracy of brain tumor segmentation in 3D MRI images and optimize the diagnosis and treatment tion faced by convolutional neural networks when applied to edge devices. FPGAs demonstrated significant advantages in optimizing convolution operations due to their high flexibility and low power consumption. The research results indicated that FPGA is a promising energy-saving platform, providing important guidance for future artificial intelligence hardware research [8]. Narang et al. proposed an innovative hardware simulator to accelerate the design cycle of microelectromechanical system (MEMS) vibration gyroscopes. This simulator was easy to reconfigure and fully integrated on a FPGA board. The research results indicated that the simulator successfully simulated the Coriolis effect and various typical MEMS gyroscope error sources, and preliminary experiments demonstrated its effective performance in noise and nonlinear models [9].

in the classification, recognition, and detection of fruits and vegetables. Therefore, the research team proposed an image recognition method that enhanced binary segmentation by combining the idea of interest classification and the large law threshold method. The experimental outcomes denoted that regardless of the presence of additional noise, the recognition accuracy of this method was always superior to traditio

process of brain tumors. This architecture integrated intelligent algorithms, especially PSO and Darwin PSO techniques, and was implemented on the Xilinx Virtex6 FPGA platform. The outcomes indicated that this hardware architecture outperformed in segmenting brain tumors, with efficient and robust performance, providing clinical doctors with powerful diagnostic tools and potentially improving the speed and accuracy of diagnosis and treatment [7]. Hong et al. developed a convolutional neural network model design method using FPGAs as accelerators to overcome the challenges of large models, high computational complexity, and huge power consump In summary, in the last few years, many scholars have begun to explore image recognition and contour extraction, and have successively proposed various recognition methods. However, this field still faces challenges such as complex backgrounds and uneven lighting, which limits the effectiveness of traditional methods such as simple threshold segmentation. To address these challenges, a highly efficient image edge recognition model combining Sobel algorithm and FPGA technology has been proposed. It is expected to provide an efficient image edge recognition solution suitable for real-time image processing applications, showcase the potential application of FPGA in image processing for providing reference for research and application in related fields, and further explore the combination of FPGA and traditional image processing algorithms to promote the development of image processing technology.

Table 1: Related works

Authors	Method	Application	Key findings	Performance comparison	References
Yang P et al.	Threshold deviation adjustment strategy	Image segmentation	Improved segmentation results and reduced misclassification rate	Superior to Otsu's method	[3]
Zheng J et al.	Improved Particle Swarm Algorithm	Image segmentation	Enhanced convergence speed and accuracy; 30% faster runtime	Faster and more accurate than traditional PSO	[4]
Vite-Chávez O et al.	Interest classification + Otsu's method	Fruit and vegetable recognition	Recognition accuracy superior to traditional methods	Robust to noise	[5]
Chen L et al.	Adaptive fractional-order genetic classification	Image segmentation	Superior results in contrast and peak SNR	Better than pre-improved algorithms	[6]
Gtifa W et al.	Hardware architecture + PSO	Brain tumor segmentation	Enhanced accuracy for clinical diagnosis support	Efficient and robust	[7]
Hong H et al.	FPGA-accelerated convolutional neural network	Edge device application	Reduced power consumption and improved computational efficiency	Superior to traditional deep learning models	[8]
Narang S et al.	Hardware simulator	MEMS gyroscope design	Effectively simulated Coriolis effects and error sources	Good performance in noise and nonlinear models	[9]

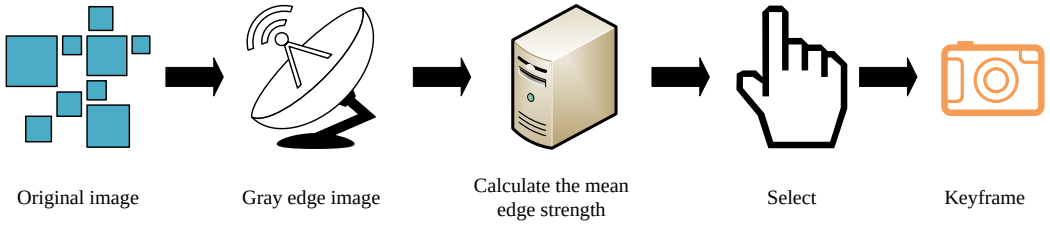


Figure 1: Keyframe image extraction process.

3 Methods

The first section addresses the issue of a large amount of noise in the dataset by using optical flow (OF) to process the image data, and then converting the RGB image into YCbCr image. In the second section, the Sobel algorithm is improved to address the issues with traditional algorithms and controlled using FPGA technology.

3.1 Image preprocessing method based on YCbCr

Different from the situation where the scale distribution of detection objects is relatively uniform in actual environments and conventional applications, it is necessary to preprocess the video data to improve the training performance of subsequent models. The study used OF for preprocessing video data [10-11]. To ensure that the images in the input algorithm model contain more information, it is necessary to select the keyframe with the highest information content from the video clip as the input of the model. The extraction flow is denoted in Figure 1.

In Figure 1, firstly, the edge grayscale image is extracted from the initial image, and then its average edge intensity (AEI) is calculated. The expression for the AEI is shown in equation (1) [12-13].

$$D_i = \frac{\sum_{x=1}^{W}\sum_{y=1}^{H} edge_i(x,y)}{W \times H} \tag{1}$$

In equation (1), D_i denotes the AEI of the i th frame image. H and W denote the height and width of the grayscale edge image. $edge_i(x,y)$ denotes the grayscale edge image The higher the AEI of an image, the more information it contains and the higher its clarity.

The image with the highest AEI is selected from each segment and it is considered as the key image frame, with its position shown in equation (2).

$$Index = \arg\max(D_i) \tag{2}$$

In equation (2), denotes the max edge intensity of the image. In some cases, both the camera and objects in the scene are in motion, and OF calculations can be used to estimate the joint motion of the camera and objects. The OF method requires the assumption of constant brightness in advance, that is, the grayscale values of pixels remain stable and unchanged during motion. The formula for brightness consistency is indicated in equation (3).

$$I(x,y,t) = I(x+dx, y+dy, t+dt) \tag{3}$$

In equation (3), $I(x,y,t)$ denotes the grayscale value of a pixel point with x on the x-axis and y on the y-axis at time t. y represents the variation of pixel points. By expanding equation (3) with a first-order Taylor formula, the basic OF constraint equation is obtained, as shown in equation (4).

$$I_x u + I_y v + I_t = 0 \tag{4}$$

In equation (4), I_x, I_y, and I_t represent the partial derivatives of $I(x,y,t)$ to x, y, and t, respectively. For image data, dense OF needs to be obtained, which requires calculating the motion velocity vector of each pixel in the image, rather than only calculating a selected portion of the pixels [14-15]. Therefore, total variation regularization with L1 norm minimization (TV-L1) is used to extract the OF of video frames in both directions, with the objective function denoted in equation (5).

$$\min_{u,v} E(u,v) = \iint [|T(x,y) - I(x+u, y+u)| + \lambda(|\nabla u| + |\nabla v|)]dxdy \tag{5}$$

In equation (5), $I(x,y)$ represents the current video frame, $T(x,y)$ represents the reference video frame, and these two video frames are continuous. λ represents the weight parameter. u represents the horizontal offset of a pixel, while v represents the vertical offset of a pixel. $|\nabla u|$ and $|\nabla v|$ respectively represent the length of the two-dimensional gradient. By solving equation (5) through TV-L1, the simplified objective function is obtained as shown in equation (6).

$$\min_{u,v} E(u,v) = \int \left[|\lambda| \rho(v) + \frac{1}{2\theta}(u-v)^2 + |\nabla u| \right] dx \tag{6}$$

In equation (6), θ represents a constant. The image processed by OF method is still an RGB image. To extract more features, the RGB image is converted to a YCbCr image. The conversion from RGB to YCbCr is actually based on the conversion of three basic colors in the color space of brightness and chromaticity.

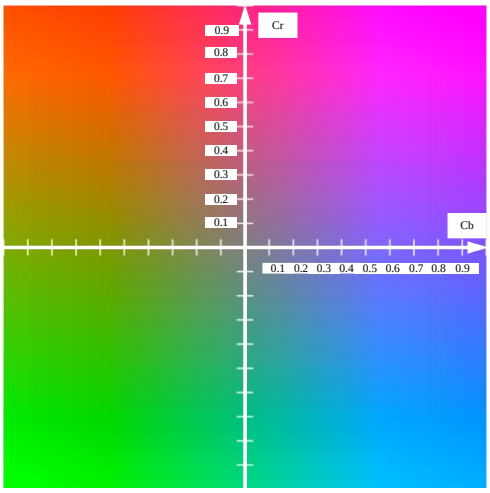


Figure 2: YCbCr color space.

image, while the chromaticity component denotes the color information. This separation makes it more flexible in handling brightness and chromaticity [16-17]. Secondly, the human eye is more sensitive to brightness information than chromaticity information, and different sampling rates can be used to handle the degree and chromaticity components when compressing images. Usually, the luminance component retains a higher resolution, while the chrominance component undergoes a higher degree of compression. The brightness component Y represents the brightness of the image, that is, the grayscale information. The brightness component usually occupies most of the information encoded in the image. The blue CDC Cb represents the color difference information of blue. The blue CDC denotes the difference between the blue color and brightness of a pixel. The red CDC Cr denotes the color difference information of red, and the red CDC represents the difference between the pixel's red color and brightness. The expression for converting the brightness component Y from the RGB color space is shown in equation (7).

$$Y = 0.299R + 0.587G + 0.114B \tag{7}$$

In equation (7), R refers to the value of the red CDC in RGB, G indicates the value of the green CDC in RGB, and B expresses the value of the blue CDC in RGB. The expressions for the blue CDC Cb and the red CDC Cr are shown in equation (8).

$$\begin{cases} Cb = -0.1687R - 0.3313G + 0.5B + 128 \\ Cr = 0.5R - 0.4187G - 0.0813B + 128 \end{cases} \tag{8}$$

YCbCr represents a pixel with three components, namely brightness component Y, blue color difference component (CDC) Cb, and red CDC Cr. There are two advantages to image processing using the YCbCr format. Firstly, the YCbCr format divides image colors into luminance and chrominance components. The brightness component represents the brightness information of the

In equation (8), B represents the value of the blue CDC in RGB. The color space of YCbCr is shown in Figure 2.

One of the main advantages of YCbCr is that it optimizes the perceptual characteristics of the human eye, allowing for more effective compression while maintaining image quality, fully utilizing the human eye's sensitivity to brightness changes and relatively low sensitivity to color changes.

3.2 Establishment of Image edge recognition model based on FPGA

After analyzing the original image data, the feature information in the data can be recognized. The research adopted Sobel algorithm combined with FPGA. Sobel operator is a classic edge detection method used in the fields of image processing and computer vision. This method recognizes edges in an image by calculating the gradient of image grayscale values [18-19]. The core feature of this method is directional detection ability, which can capture edge information in the X and Y directions through two convolution kernels, horizontal and vertical. Meanwhile, the Sobel operator can be extended to larger convolution kernels and more detection directions can be added to improve the accuracy of edge detection. In addition, the Sobel operator has a certain built-in smoothing effect, which gives it good noise suppression ability. Compared to simple differential operators, Sobel operator can effectively reduce small noise in images during edge extraction, ensuring its stability in low SNR images. Especially on hardware platforms such as FPGA, efficient parallel operations can be achieved, thereby improving processing speed. In terms of implementation, the Sobel operator algorithm has a simple principle and mainly uses addition and subtraction operations. It has low hardware and programming requirements and is easy to integrate into various image processing systems. Ultimately, the Sobel operator can generate clear edge responses in edge detection, resulting in distinct edge contours that are suitable for tasks such as image enhancement and feature extraction. The Sobel operator mainly uses convolution operations to calculate the gradient value of each pixel point, thereby detecting the edges of the image. The Sobel operator is based on a pair of 3×3 convolution kernels, which are used to calculate gradients in both directions. By convolving these two convolution kernels separately with the image, the gradient of the image in both directions can be calculated, as expressed in equation (9).

$$\begin{cases} G_x = \sum_{i=-1}^{1}\sum_{j=-1}^{1} K_x(i,j)\cdot I(x+i,y+i) \\ G_y = \sum_{i=-1}^{1}\sum_{j=-1}^{1} K_y(i,j)\cdot I(x+i,y+i) \end{cases} \quad (9)$$

In equation (9), K_x and K_y represent the convolution kernels in the horizontal and vertical directions, respectively. $I(x+i,y+i)$ represents the grayscale value of the image at the corresponding position. After calculating the gradient of each pixel in both directions, the gradient amplitude and direction of that point can be further calculated [20]. The gradient amplitude is indicative of the strength of the edge, while the gradient direction is representative of the edge's orientation. The expression is shown in equation (10).

$$\begin{cases} G = \sqrt{G_x^2 + G_y^2} \\ \theta = \arctan\left(\dfrac{G_y}{G_x}\right) \end{cases} \quad (10)$$

In equation (10), G represents the gradient amplitude and θ represents the gradient direction. However, an approximate value without a square is usually used instead to improve efficiency, so its expression can be rewritten as shown in equation (11).

$$|G| = |G_X| + |G_Y| \quad (11)$$

In equation (11), G_X and G_Y indicate the gradient amplitudes in the X and Y directions, respectively. An appropriate threshold is set based on the actual situation and accuracy requirements of image edge detection. When the gradient exceeds the threshold, it indicates that the pixel point is an edge point. Otherwise, the point is not an edge point. The judgment expression is shown in equation (12).

$$E = \begin{cases} 1 & \mathrm{G} \geq \mathrm{T} \\ 0 & else \end{cases} \quad (12)$$

In equation (12), E represents a Boolean quantity, T represents a threshold, and G represents the gradient amplitude. Although the Sobel operator has many advantages, its shortcomings are also quite obvious, as it is very sensitive to noise when detecting image edges.

The noise in the image can introduce high-frequency components, leading to an abnormal increase in gradient values and interfering with edge detection results. In this case, the edges detected using the Sobel operator may not be accurate and may even produce many false edges. The Sobel operator uses a fixed 3×3 convolution kernel, although this kernel size calculation is simple, it may not be flexible enough when processing images of different scales [21-22]. For images with rich details, a 3×3 kernel may not be sufficient to capture all the details, while for simple images, this kernel may introduce unnecessary computational complexity.

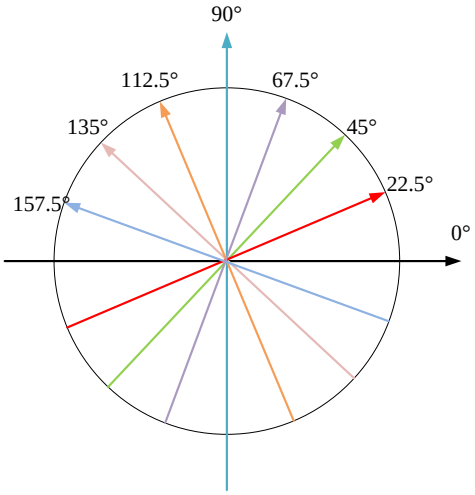


Figure 3: Direction detected by improved Sobel operator.

Therefore, it was improved by changing the size to 5×5 and eight directions based on the traditional 3×3 template, as shown in Figure 3.

The noise in the image can introduce high-frequency components, leading to an abnormal increase in gradient values and interfering with edge detection results. In this case, the edges detected using the Sobel operator may not be accurate and may even produce many false edges. The Sobel operator uses a fixed 3×3 convolution kernel, although this kernel size calculation is simple, it may not be flexible enough when processing images of different scales [21-22]. For images with rich details, a 3×3 kernel may not be sufficient to capture all the details, while for simple images, this kernel may introduce unnecessary computational complexity. Therefore, it was improved by changing the size to 5×5 and eight directions based on the traditional 3×3 template, as shown in Figure 3.

In Figure 3, in the improved Sobel operator, the eight angles are 0°, 22.5°, 45°, 67.5°, 90°, 112.5°, 135°, and 157.5°, respectively. By convolving the original image data with eight directional templates, the edge information of the image can be more complete, and each pixel can obtain eight different values. The improved Sobel module flowchart is shown in Figure 4. From Figure 4, it can be seen that firstly, a 5×5 filtering template is generated through two dual terminal RAM two-stage registers. Secondly, the derivatives in each direction are calculated separately. Then, the square root of the partial derivatives is calculated, and in contrast with the threshold, the value greater than the threshold is 1, and the opposite is 0. Finally, the row field signal and the enable signal are synchronously delayed. The study uses FPGA to control Sobel modules. FPGA is an integrated circuit device that can be programmed by users to achieve specific logic functions and digital circuits. FPGA has programmability and flexibility, and can be reconfigured and reprogrammed according to user needs. FPGA is composed of a large number of programmable logic units, programmable interconnect resources, and other supporting circuits. Programmable logic units allow users to define logical functions, while programmable interconnect resources are responsible for connecting

signal paths between logic units. Its structure is shown in Figure 5.

In Figure 5, configurable logic block (CLB) is the most basic programmable logic unit in FPGA, utilized to implement logic functions. CLB has programmable logic functions and connection resources, enabling it to implement various digital circuits. The arithmetic units in FPGA usually include adders, multipliers, and accumulators, which are used to implement tasks such as digital signal processing and algorithm operations. Block

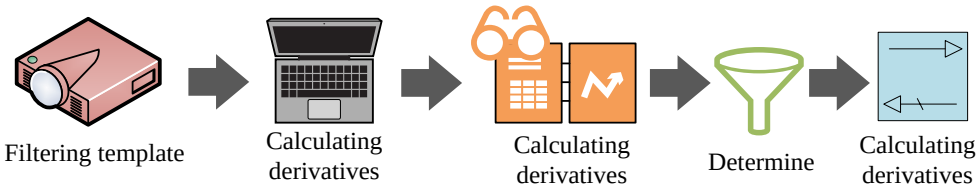


Figure 4: Improved Sobel module process.

RAM (BRAM) is a block storage unit used in FPGA to store data, with high-speed read and write characteristics and low power consumption. BRAM is typically used to store data, intermediate results, and algorithm parameters. Phase-locked loop (PLL) is a module in FPGA used to generate stable clock signals. It can generate multiple internal clock signals based on external clock signals and has the function of a PLL to ensure the stability and accuracy of clock signals. Input output block (IOB) refers to input/output resources, which are modules used in FPGA to connect external input/output signals and communicate with external devices. Network interconnection resources in FPGA are responsible for connecting various logic blocks, storage units, and input/output resources to form a complete digital circuit. The final model structure is shown in Figure 6.

In Figure 6, first is to generate a 5×5 matrix. Next is to calculate the mean of the matrix, and then calculate the gradients of the image in multiple directions, to enhance the edge detection effect. Then, the amplitude of the gradients in each direction is obtained by taking the square sum and square root of the gradients.

Subsequently, the adaptive threshold method is used to compare the gradient amplitude and determine whether it is an edge.

The final model is mainly divided into four modules. The first module is the filtering template generation module, which uses FPGA's dual terminal RAM to generate 5×5 Sobel templates for edge detection in different directions. The use of dual terminal RAM reduces access latency and provides sufficient band width to support parallel processing, making data flow smoother. Next is the gradient calculation module, which calculates gradients in the horizontal, vertical, and eight directions respectively. The gradient calculation in each direction is implemented through dedicated adders and multipliers, utilizing the parallel computing characteristics of FPGA to reduce processing time. In the sum of squares module, the gradient values in each direction are subjected to a sum of squares operation, followed by square root calculation to obtain the gradient amplitude of each pixel point. This module adopts the FPGA pipeline processing architecture, where each operation is executed within an independent clock cycle, ensuring the continuity of the data stream and improving computational efficiency. Finally, there is the threshold judgment module. After the gradient amplitude calculation is completed, the threshold judgment module performs adaptive threshold comparison on the amplitude to determine whether it is an edge. The Otsu method was used as the adaptive threshold method in the study. Firstly, the histogram of the image was calculated and the intra-class and inter-class variances were initialized. Then, by gradually traversing the gray levels, the optimal segmentation point is found, where the inter-class variance is maximized. Finally, the selected threshold is applied to edge detection.

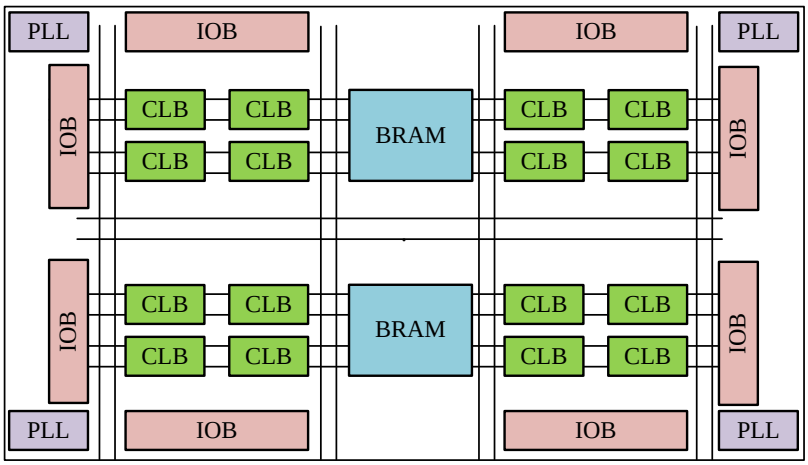


Figure 5: FPGA structure analysis.

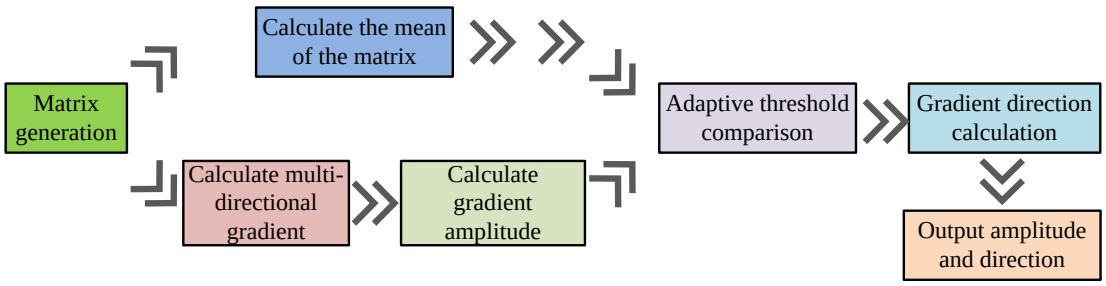


Figure 6: Flow of improved Sobel operator on FPGA.

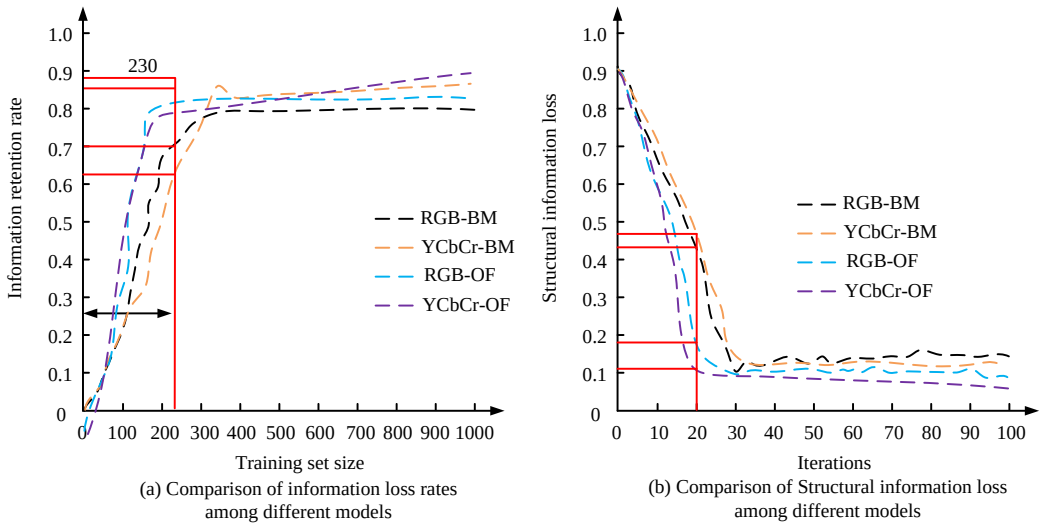


Figure 7: Comparison of information retention rate and structural information loss among various models.

4 Results

The first section of the study analyzes the performance of the raised data preprocessing method, and the second section analyzes and simulates the performance of the image edge detection model.

4.1 Image preprocessing method based on YCbCr

The server CPU used in the study was Inter (R) Core (TM) i5-10210U, with 16GB of RAM, NVIDIA Geforce GTX2080Ti GPU, and 8GB of video memory. The operating system was Windows 10. The dataset was the publicly available Kaggle dataset, named Kaggle Surveillance Video Dataset, which contains 6000 images extracted from surveillance videos at different time periods. It was mainly used for object detection and behavior recognition tasks. In terms of image properties, the dataset included color images or grayscale images in PNG format. The resolutions were 640x480 (VGA), 1280x720 (720p), and 1920x1080 (1080p). The diversity of image content was reflected in the pedestrians, vehicles of different types and colors, and various background scenes that may appear in the monitoring scene. The monitoring images also included different lighting conditions, covering both daytime and nighttime scenes, ensuring the diversity and complexity of the dataset. A dataset consisting of 6000 images was divided into a training set and a validation set in a 5:1 ratio. The introduction of Block Matching (BM) method was compared and analyzed with the method proposed in the study, and the results are shown in Figure 7.

Figure 7 (a) showcases the comparison of information retention rates of various models under different dataset sizes, and Figure 7 (b) showcases the comparison of structural information loss of various models under different iteration times. In Figure 7 (a), with the increase of the dataset, the information retention rate of the images processed by the four models also increased. When the dataset size was around 230, the performance of RGB-OF model and YCbCr-OF model basically reached the best. When the dataset size was 1000, the information retention rates of RGB-BM model, YCbCr-BM model, RGB-OF model, and YCbCr-OF model were 0.81, 0.83, 0.86, and 0.89, respectively. In Figure 7 (b), with the increase of iteration times, the loss of structural information in the images processed by the four models decreased. When the iteration times were around 20, the performance of RGB-OF model and YCbCr-OF model basically reached the best. When the amount of iterations was 100, the structural information loss of RGB-BM model, YCbCr-BM model, RGB-OF model, and YCbCr-OF model was 0.15, 0.13, 0.09, and 0.05, respectively. The experiment outcomes

illustrated that the proposed YCbCr-OF-based image preprocessing model had a high information retention rate and low structural information loss. The dataset was divided into dataset A and dataset B according to size, and the processing time of each model was compared. The findings are illustrated in Figure 8.

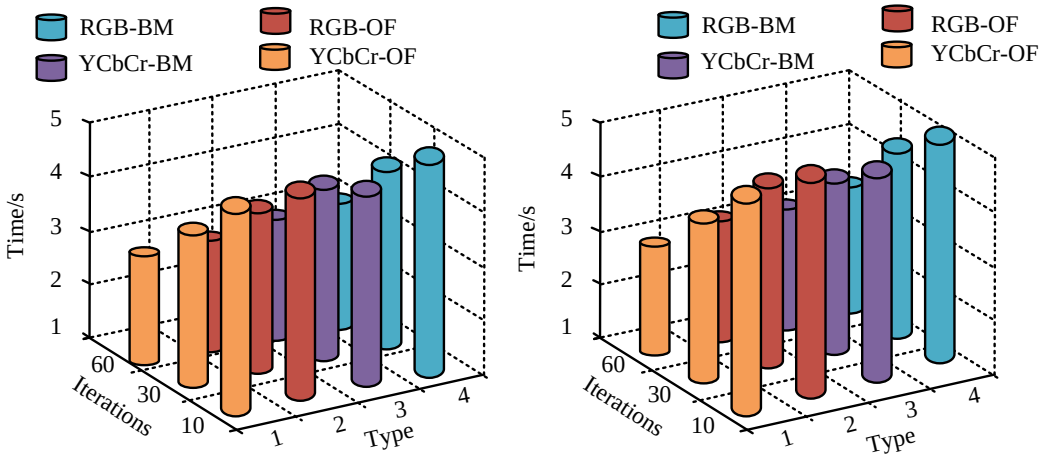
Figures 8 (a) and (b) show the recognition time of different algorithms in dataset A and dataset B, respectively. According to Figure 8 (a), the effectiveness of the model did not reach its optimal level when the number of iterations was low, resulting in longer recognition time. When the number of iterations was 60, the processing times for YCbCr-OF model, RGB-OF model, YCbCr-BM model, and RGB-BM model were 2.3, 2.6, 2.7, and 2.9 seconds, respectively. In Figure 8 (b), each algorithm model took slightly more time on dataset B than on dataset A. When the number of iterations was 60, the scheduling times for YCbCr-OF model, RGB-OF model, YCbCr-BM model, and RGB-BM model were 2.6s, 2.8s, 3.0s, and 3.1s, respectively. The findings illustrated that the developed algorithm model performed well in two different datasets, and the YCbCr-OF model took less processing time than other algorithm models. The comprehensive performance of the four models was analyzed, and the outcomes are indicated in Table 2.

According to Table 2, in dataset A, the loss function value (Loss), intersection over union (IoU), F1 value, and Micro F1 of the YCbCr-OF algorithm model were 0.189, 0.883, 0.612, and 0.957, respectively. In dataset B, the Loss, IoU, F1 value, and Micro F1 of the YCbCr-OF algorithm model were 0.177, 0.899, 0.618, and 0.939, respectively. The YCbCr-OF algorithm model proposed in the four methods has excellent performance in various aspects.

4.2 Analysis of image edge recognition model Based on FPGA

In the Roberts operator, the learning rate was set to 0.001, the batch size was 32, the number of iterations was 1000, the initial adaptive threshold was set to 20-40, and the step size was 5. In terms of FPGA configuration, the clock speed was 100 MHz, the data width was 16 bit fixed-point, the on-chip memory used 50% BRAM, and the power consumption was controlled within 10 W. To verify the performance advantages of FPGA, it was compared with CPU and GPU, and the results are shown in Table 3.

According to Table 3, the processing time of FPGA was 15 milliseconds, which is significantly faster than GPU's 28 milliseconds and CPU's 45 milliseconds. FPGA utilized its high parallel computing architecture to efficiently perform convolution operations and gradient calculations, thereby reducing computational latency and meeting real-time edge detection requirements. In terms of power consumption, the average power consumption of



(a) Comparison of time in dataset A

(b) Comparison of time in dataset B

Figure 8: Processing time of different models in different datasets.

Table 2: Comparison of comprehensive performance of models.

Index	Dateset	RGB-BM	YCbCr-BM	RGB-OF	YCbCr-OF
Loss	A	0.322	0.279	0.252	0.189
	B	0.308	0.267	0.238	0.177
IoU	A	0.751	0.793	0.82	0.883
	B	0.764	0.809	0.834	0.899
F1	A	0.472	0.515	0.542	0.612
	B	0.513	0.528	0.56	0.618
Micro F1	A	0.823	0.868	0.893	0.957
	B	0.837	0.879	0.907	0.939

Table 3: Performance Comparison of FPGA, GPU, and CPU.

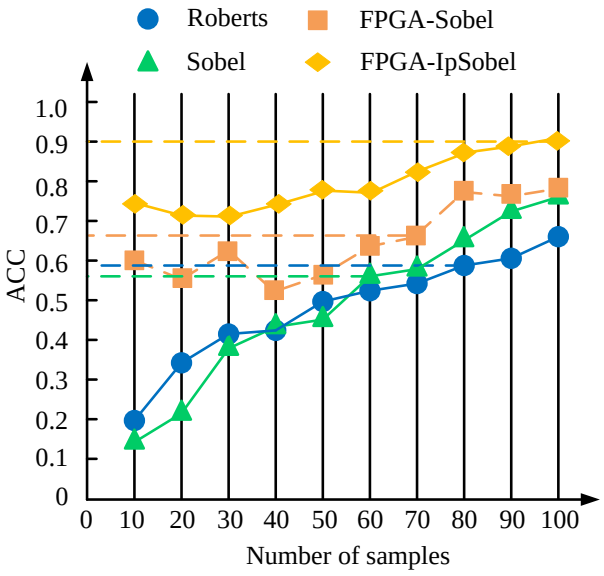
Implementation model	Hardware model	Average processing time (ms)	Average power consumption (W)
FPGA	Xilinx Zynq UltraScale+ MPSoC	15	4
GPU	NVIDIA GeForce RTX 2080	28	75
CPU	Intel Core i7-9700K	45	65

FPGA was only 4W, significantly lower than GPU and CPU. This indicates that the low-power characteristics of FPGA make it more suitable for deployment in power sensitive applications such as embedded systems and IoT devices.

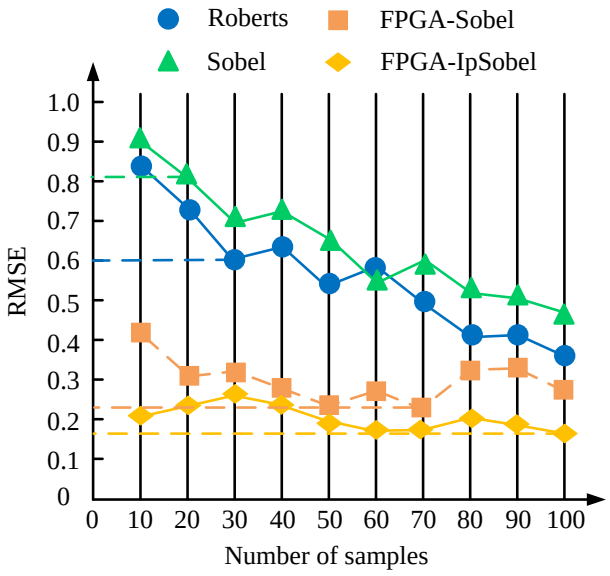
After verifying the effectiveness of the data preprocessing method, a performance analysis was conducted on the FPGA-based image edge detection model, and the Roberts operator model was introduced for comparison. The findings are indicated in Figure 9.

Figure 9 (a) represents the comparison of model accuracy (ACC) in case of different sample sizes and Figure 9 (b) represents the comparison of root mean square error (RMSE) in case of different sample sizes. From Figure 9(a), the ACC of each model increased as the sample size increased. The ACC of Roberts model, Sobel model, FPGA-Sobel model, FPGA-IpSobel model was 0.67, 0.77, 0.79, and 0.90 respectively for a data size of 100. From Figure 9(b), the RMSE values of the models were decreasing as the sample size increased. The RMSE values of Roberts' model, Sobel's model, FPGA-Sobel's model, and FPGA-IpSobel's model were 0.45, 0.37, 0.28, and 0.16, respectively, when the data size was 100. The experimental results show that the proposed FPGA-IpSobel model has a more excellent performance. Classic images in the image field were selected for analysis, and the outcomes are indicated in Figure 10.

Figure 10 (a) shows the image without edge recognition, and Figures 10 (b), (c), and (d) show the image processed by the FPGA-IpSobel model, FPGA-Sobel model, and Sobel model, respectively. From Figure 10, among the three models, the FPGA-IpSobel model was more satisfactory in image processing, retaining a large amount of details and having clearer edges.

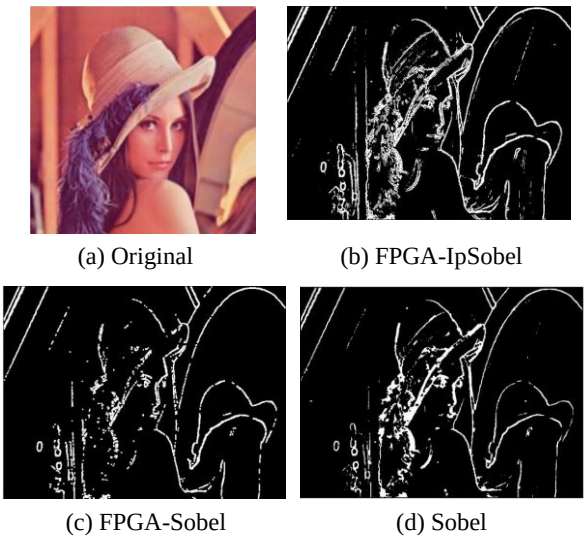


(a) Comparison of ACC among various models



(b) Comparison of RMSE among various models

Figure 9: Comparison of ACC and RMSE values for various models.



(a) Original (b) FPGA-IpSobel

(c) FPGA-Sobel (d) Sobel

Figure 10: Performance analysis of edge recognition model.

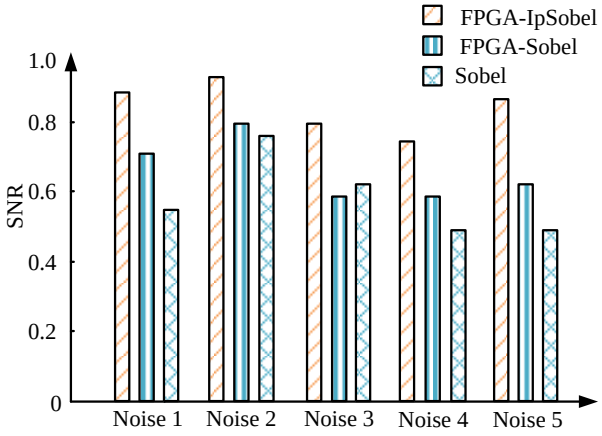


Figure 11: Comparison of SNR ratios for different noise images processed by various models.

Table 4: Actual performance analysis of various models.

Model	P	R	IoU	F1	mAP	Loss
Roberts	0.83	0.78	0.86	0.82	0.83	0.097
Sobel	0.87	0.81	0.92	0.87	0.84	0.082
FPGA-Sobel	0.90	0.86	0.96	0.91	0.88	0.076
FPGA-IpSobel	0.92	0.93	0.98	0.97	0.92	0.064

However, the FPGA-Sobel model and Sobel model had significantly poorer performance in image processing, with varying degrees of noise. The performance of the model was analyzed by selecting different types of noise, and the results are shown in Figure 11. Figure 11 shows the comparison of SNRs of different models for processing noisy images. From Figure 11, the proposed FPGA IpSobe model exhibited excellent performance in five different noisy images. Among the noisy images, the performance of each model was the best for image type 2 noise. The comprehensive effectiveness of each model was evaluated, and the findings are illustrated in Table 4.

According to Table 4, when comparing the performance of various algorithms, the difference between the Roberts method and the Sobel method was not significant, while the FPGA-IpSobel method performed well compared to the other three methods. The P-value, R-value, IoU value, F1 value, mAP, and Loss value corresponding to the FPGA-IpSobel method were 0.92, 0.93, 0.98, 0.97, 0.92, and 0.064, respectively. The experiment findings illustrated that the designed FPGA-IpSobel method exhibited excellent effectiveness.

5 Discussion

In the study, the proposed FPGA-IpSobel model demonstrated superior performance, with significant advantages in ACC and RMSE compared to other models. The experimental results showed that the FPGA-IpSobel model performed well in multiple performance indicators, such as an ACC of 0.90 under different sample sizes, which was significantly higher than the Roberts model and

the traditional Sobel model. This ACC difference was mainly due to the improved Sobel algorithm, which used a 5×5 convolution kernel and eight directional edge detection to enhance the ability to capture edge information. Meanwhile, the parallel processing capability of FPGA enabled multiple edge detection operations to be performed simultaneously, greatly improving processing speed. On the other hand, in terms of processing time, the RMSE of the FPGA-IpSobel model at a data volume of 100 was 0.16, significantly lower than that of the Roberts model and Sobel model. This efficiency improvement is mainly due to the hardware acceleration of FPGA. The hardware architecture of FPGA can efficiently configure computing resources, optimize data flow, and achieve fast real-time processing. In contrast, traditional models rely on CPUs or GPUs, which are limited by processing power and resource allocation, and can easily cause delays. In addition, the adaptive threshold processing method used in the FPGA model effectively filtered out edge information, reduces unnecessary calculations, and further improved processing efficiency. The difference between accuracy and efficiency was also reflected in the flexibility of model design. FPGA allows for the reconfiguration of logic units according to specific application requirements, providing broad space for algorithm optimization. However, traditional algorithms are often fixed and difficult to optimize specifically. In addition, models implemented on FPGA can achieve algorithm level optimization through specific hardware resources, fully utilizing the parallel computing characteristics of FPGA to improve overall efficiency.

6 Conclusion

A high-efficiency image edge recognition model combining Sobel algorithm and FPGA technology was proposed to address the real-time and processing efficiency issues of traditional edge detection algorithms. The model first performed YCbCr color space conversion on the image, then used Sobel operator to calculate the image gradient, and used adaptive thresholding to determine the edges. Finally, the model was implemented and optimized on FPGA. The experiment findings indicated that when the dataset size was 1000, the information retention rates of RGB-BM model, YCbCr-BM model, RGB-OF model, and YCbCr-OF model were 0.81, 0.83, 0.86, and 0.89, respectively, and the structural information loss was 0.15, 0.13, 0.09, and 0.05, respectively. When the data volume was 100, the ACC of Roberts model, Sobel model, FPGA-Sobel model, and FPGA-IpSobel model were 0.67, 0.77, 0.79, and 0.90, respectively, with RMSE values of 0.45, 0.37, 0.28, and 0.16, respectively. Among the three models, the FPGA-IpSobel model was more satisfactory in image processing, retaining a large amount of details and having clear edges, while the FPGA-Sobel model and Sobel model had significantly poorer performance in image processing, with varying degrees of noise. The research results indicated that the proposed FPGA-based image edge detection model improved the ACC and speed of image edge detection. However, there are also some shortcomings in the research, as the performance of this model is poor in scenarios such as high-frequency textures, complex lighting, or overlapping multiple objects. For example, in complex textured scenes such as forests, grasslands, and architectural complexes, due to the multitude of texture details, the adaptive thresholding method of the model may misidentify many small irrelevant textures as edges, leading to an increase in false edges and affecting the overall recognition accuracy. Future work can be improved in the following aspects. Firstly, a more robust edge detection algorithm is introduced and fused with Sobel algorithm to enhance the model's detection accuracy for texture backgrounds. In addition, based on the adaptive thresholding method, a scene-based threshold self-learning algorithm can be introduced to dynamically adjust edge detection parameters according to different scene features, thereby reducing false edges.

References

[1] Ádria Barros de Oliveira, and Fernanda Lima Kastensmidt. Evaluating fault-tolerant techniques on COTS RISC-V NOEL-V processor in Zynq UltraScale+ FPGA under proton testing. IEEE Transactions on Nuclear Science, 70(8):1708-1715, 2023. https://doi.org/10.1109/TNS.2023.3281396

[2] Wafa Gtifa, and Anis Sakly. Integrating Xilinx FPGA and intelligent techniques for improved precision in 3D brain tumor segmentation in medical imaging. Journal of Real-Time Image Processing, 20(6):115.1-115.15, 2023. https://doi.org/10.1007/s11554-023-01372-x

[3] Pei Yang, Wei Song, Xiaobing Zhao, Rui Zheng, and Letu Qingge. An improved Otsu threshold segmentation algorithm. International Journal of Computational Science and Engineering, 22(1):146-153, 2020. https://doi.org/10.1504/ijcse.2020.107266

[4] Zheng J, Gao Y, Zhang H, Lei Y, Zhang J. OTSU multi-threshold image segmentation based on improved particle swarm algorithm. Applied Sciences, 12(22):11514-11515, 2022. https://doi.org/10.1109/ICICSP48821.2019.8958573

[5] O. Vite-Chavez, J. Flores-Troncoso, Reynel Olivera-Reyna, and Jorge Ulises Munoz. Improvement procedure for image segmentation of fruits and vegetables based on the otsu method. Image Analysis and Stereology, 42(3):185-196, 2023. https://doi.org/10.5566/ias.2939

[6] Liping Chen, Jinhui Gao, António M. Lopes, Zhiqiang Zhang, Zhaobi Chu, and Ranchao Wu. Adaptive fractional-order genetic-particle swarm optimization Otsu algorithm for image segmentation. Applied Intelligence, 53(22):26949-26966, 2023. https://doi.org/10.1007/s10489-023-04969-8

[7] Wafa Gtifa, and Anis Sakly. Integrating Xilinx FPGA and intelligent techniques for improved precision in 3D brain tumor segmentation in medical imaging. Journal of Real-Time Image Processing,

20(6):115.1-115.15, 2023. https://doi.org/10.1007/s11554-023-01372-x

[8] Hyeonseok Hong, Dahun Choi, Namjoon Kim, Haein Lee, Beomjin Kang, Huibeom Kang, and Hyun Kim. Survey of convolutional neural network accelerators on field-programmable gate array platforms: architectures and optimization techniques. Journal of Real-Time Image Processing, 21(3):156-167, 2024. https://doi.org/10.1007/s11554-024-01442-8

[9] Sanjoli Narang, and Siddharth Tallur. Field-programmable gate array (FPGA) based programmable digital emulator of vibratory microelectromechanical systems (MEMS) gyroscopes. The Review of Scientific Instruments, 93(3):353-359, 2022. https://doi.org/10.1063/5.0065642

[10] Raúl Lora-Rivera, Óscar Oballe-Peinado, and Fernando Vidal-Verdú. Texture detection with feature extraction on embedded FPGA. IEEE Sensors Journal, 23(11):12093-12104, 2023. https://doi.org/10.1109/JSEN.2023.3268794

[11] Mara Pistellato, Filippo Bergamasco, Gianluca Bigaglia, Andrea Gasparetto, Andrea Albarelli, Marco Boschetti, and Roberto Passerone. Quantization-aware NN layers with high-throughput FPGA implementation for edge AI. Sensors, 23(10):141-149, 2023. https://doi.org/10.3390/s23104667

[12] Srinivasan Kalaiarasu, and Sudhakar Natarajan. Conducted electromagnetic interference mitigation on two-stage cascaded boost (TSCB) DC-DC converter using FPGA based DCPWM technique for EV Applications. Journal of Electrical Engineering & Technology, 18(3):2003-2013, 2022. https://doi.org/10.1007/s42835-022-01264-3

[13] Yuchen Yang, Zhongtao Shen, Xing Zhu, and Ziqi Wang. FPGA-based electronic system for the control and readout of superconducting quantum processors. The Review of Scientific Instruments, 93(7):74701.1-74701.14, 2022. https://doi.org/10.1063/5.0085467

[14] Christian Pilato, Zhenman Fang, Yuko Hara-Azumi, and Jim Hwang. Introduction to the special section on high-level synthesis for FPGA: next-generation technologies and applications. ACM Transactions on Design Automation of Electronic Systems, 27(4):29-31, 2022. https://doi.org/10.1145/3519279

[15] Yongli Gao, and Zijie Zhou. Automatic recognition and repair system of mural image cracks based on cloud edge computing and digitization. Mobile Information Systems, 22(31):1534596.1-1534596.12, 2022. https://doi.org/10.1155/2022/1534596

[16] K. P. Krishna Kumar, and Varghese Paul. Complementary spatial transformer network for real-time 3D object recognition a tiny deep learning model in target space. Journal of Real-Time Image Processing, 20(5):88.1-88.12, 2023. https://doi.org/10.1007/s11554-023-01340-5

[17] Yuzhuo Li, Jiang Lei, Xinrong Li, and Wenqian Feng. Non-contact clothing anthropometry based on two-dimensional image contour detection and feature point recognition. Industria Textila, 74(1):67-73, 2023. https://doi.org/10.35530/IT.074.01.202279

[18] Florian A. Mann, Phillip Galonska, Niklas Herrmann, and Sebastian Kruss. Quantum defects as versatile anchors for carbon nanotube functionalization. Nature Protocols, 17(3):727-747, 2022. https://doi.org/10.1038/s41596-021-00663-6

[19] Yanmin Guo, Yu Wang, Kai Meng, and Zongna Zhu. Otsu multi-threshold image segmentation based on adaptive double-mutation differential evolution. Biomimetics, 8(5):418-421, 2023. https://doi.org/10.3390/biomimetics8050418

[20] Bo Xu, Songting Zou, Libing Bai, Kai Chen, and Jia Zhao. A general discrete memristor emulator based on Taylor expansion for the reconfigurable FPGA implementation and its application. Nonlinear Dynamics, 112(2):1395-1414, 2024. https://doi.org/10.1007/s11071-023-09092-4

[21] Padmaprabha Preethi, and Hosahalli Ramappa Mamatha. Region-based convolutional neural network for segmenting text in epigraphical images. Artificial Intelligence and Applications, 1(2):119-127, 2023. https://doi.org/10.47852/bonviewAIA2202293

[22] Oussama Azzouzi, Mohamed Anane, Mouloud Koudil, Mohamed Issad, and Yassine Himeur. Novel area-efficient and flexible architectures for optimal Ate pairing on FPGA. Journal of supercomputing, 80(2):2633-2659, 2024. https://doi.org/10.1007/s11227-023-05578-5