

Design of mean filter using field programmable gate arrays for digital images

Duong Huu Ai¹, Van Loi Nguyen², Khanh Ty Luong², Viet Truong Le²

¹Faculty of Electronics and Computer Engineering, The University of Danang, Vietnam-Korea University of Information and Communication Technology, Danang City, Vietnam

²Faculty of Computer Science, The University of Danang, Vietnam-Korea University of Information and Communication Technology, Danang City, Vietnam

Article Info

Article history:

Received Mar 5, 2024

Revised Aug 7, 2024

Accepted Aug 31, 2024

Keywords:

Algorithm for image processing

Design filter

Field-programmable gate array

Image edge detection

Image processing

ABSTRACT

In this paper, we design and analysis of mean filter using field programmable gate arrays (FPGAs) for digital images, FPGAs are integrated circuits consisting of interconnections that connect programmable internal hardware blocks allows users to customize operations for a specific application. FPGA is an ideal choice for real-time image processing, these FPGA devices are controlled in Verilog or VHDL languages, allowing to design at different levels and adapt to design changes or even support new applications throughout the life of the component. Digital image filtering is the most important task in image processing and with the help of computers, image recognition involves identifying and classifying objects in an image. This paper design of mean filter for digital image processing, implementation and analysis of image processing algorithms on FPGAs. The results obtained on the FPGA are compared and analyzed with the results by MATLAB software.

This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



Corresponding Author:

Duong Huu Ai

Faculty of Electronics and Computer Engineering, The University of Danang

Vietnam-Korea University of Information and Communication Technology

Ngu Hanh Son District, Danang City, Vietnam

Email: dhai@vku.udn.vn

1. INTRODUCTION

Digital image processing is a field that has been and is developing very quickly, related to many different fields of study and research directions. As the computing power of computers becomes more and more powerful, computers can now process large data sets, such as videos, images, concepts and techniques. Digital image processing is increasingly being discussed and researched. Digital image processing is considered as one of the fastest growing areas of technology today used in many industries. It is a complex of applications that ranges from image enhancement to automatic image recognition, to computer vision or robotics processing tasks [1]-[8].

Image processing tasks involving various processes such as image enhancement and object detection can be performed easily on computer systems, but are time-limited because of additional constraints on memory and peripherals. Executing specific applications on hardware often results in higher processing speeds than on software. With advancements in VLSI technology, hardware-based implementations are slowly taking over the market. Complex computational tasks on hardware are solved by parallel and overlapping algorithms; this will reduce most of the processing time [9]-[17].

Field programmable gate arrays (FPGAs) is an ideal choice for real-time image processing. These FPGA devices are controlled in Verilog or VHDL languages, allowing engineers to design at different levels.

Specifically, these hardware description languages allow engineers not only to describe the image processing circuitry at a logical level, but also to simulate and evaluate processing performance [18]-[26].

This paper design of mean filter using FPGAs for digital images, and is organized as follows. Section 2 present the digital image filter methods. Section 3, the system analysis and design are described. Section 4, the results and discussions are presented, and the paper is concluded in section 5.

2. DIGITAL IMAGE FILTER METHODS

2.1. Digital image processing algorithms

Windowing operator: in the implementation of basic digital image processing algorithms, a special operator called the window operator is often used. The window operator is a set of definite shape, consisting of pixels associated with a central pixel, which is the pixel being processed. Operations on these pixels will have an effect on the central pixels that are also pixels being processed in an image processing algorithm. The window operator has many shapes, depending on the implementation algorithm. However, the most commonly used operators are square operators with odd sides, for example: 3×3, 5×5, 7×7.

Convolution: is not an image processing algorithm, but a common operation in image processing algorithms that use the window operator. Convolution is used in edge detection and linear filtering problems. Examble of convolution is shown in Figure 1. Convolution computes the new value of the center pixel of the window operator, by performing the calculation with the neighboring pixels and the center pixel itself.

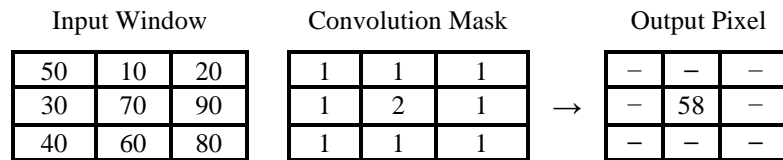


Figure 1. Example of convolution

Filters: in many technical fields, noise plays a major role in causing difficulties when we need to analyze a certain signal, nor does it exclude the image signal. There is a big difference between a real image and the resulting digitized image because there are many interfering processes. It is caused by electronic noise of the receiver or poor quality of the digitizer. Let's see how noise is represented in the image.

Assume the image is a region of uniform gray level. Thus, the elements of the matrix representing the image after the digitization process must have the same value. But actually observed, we see: near the average value of the gray level, there are elements that dominate quite a lot. That is the noise phenomenon. Thus, the noise in the digital image is considered as a fast displacement of the received signal (image signal I[m,n]) over a short distance. Considered equivalently in frequency space, the noise corresponds to the high frequency components in the image.

2.2. Analysis of mean filter

Mean filter is a linear filtering technique [10]. The averaging filter algorithm is to use a filter window (3×3), which scans each input pixel in turn. At the position of each pixel, the value of the corresponding pixels in the 3×3 region of the original image is "filled" into the filter matrix. In output image, the pixel value is the average value of all pixels in the filter window.

With mean filtering, each pixel (Pixel) is replaced by the weighted average of the points in the neighboring. Assume that there is a filter matrix (Kernel) (3×3) that scans each pixel of the input image Isrc. At the position of each pixel, the value of the corresponding pixels in the region (3×3) of the original image is put into the filter matrix (Kernel). The pixel value is the average of all the pixels in the image in the filter matrix.

To smooth an image, a moving window of N×N pixels of the image can be performed. For example, with the 3×3 filter below, set the center pixel of the window on the given pixel; multiply the image's pixels by the pixels in the window, sum the result, and copy as the value of the output pixel. Then move the window from one place to the right or down and repeat; The operation is called convolution. The filter mask is shown in the equation.

$$H(i, j) = \begin{bmatrix} 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \end{bmatrix} = 1/9 \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

By blurring the image, the median filter removes noise from the image, this process is done through the following steps:

- Step 1: calculate the sum of the components in the filter matrix (Kernel).
- Step 2: divide the average of the sum of the components in the matrix calculated above with the number of elements of the filter window to get 1 value $I_{tb}(x, y)$.
- Step 3: correction:

$$\begin{aligned} \text{If } I(x, y) - I_{tb}(x, y) > \theta \text{ then } I(x, y) &= I_{tb}(x, y) \\ \text{If } I(x, y) - I_{tb}(x, y) \leq \theta \text{ then } I(x, y) &= I(x, y) \end{aligned}$$

where $I(x, y)$ is the pixel value at a point (x, y) and a threshold θ , $I_{tb}(x, y)$ is the average of all pixels in the filter window. The averaging filter algorithm to compute its output, therefore, the design of the algorithm in VHDL is difficult. The VHDL synthesis engine provides efficient mapping to hardware mathematical designs for it. The hardware divider on FPGA is quite big and slow, we use the bitwise division by shift method. Since this can be done with powers of two, division by 8 is performed instead of division by 9, as intended in the design of the algorithm.

2.3. Field programmable gate arrays kit

The FPGAs kit used is Virtex-7 VC707, that is a high flexible, full featured, and high speed serial base platform. The Virtex-7 XC7VX485T-2FFG1761C includes basic components of design tools, hardware, IP, serial connectivity and advanced memory interface, and pre-verified reference design for system that demand of high performance. The schematic diagram of the Board Virtex-7 VC707 is shown in Figure 2.

The Virtex-7 FPGA VC707 kit has basic features such as; pre-verified reference designs, designs tools, hardware, and internet protocol; the Virtex-7 VX485T FPGA is using for 40 GB/s connectivity platform for high bandwidth applications, and high performance; embedded processing support with RISC 32 bit, Micro Blaze. And it can develop network applications with 1000 Mbps, 100 Mbps, 10 Mbps ethernet. It allows serial connections to SFP+ and IIC pairs, PCIe Gen2x8, SMA, UART; memory interface with 1 GB DDR3 SODIMM Memory up to 800 MHz/1600 Mbps; expand I/O with FPGA Mezzanine card interfaces.

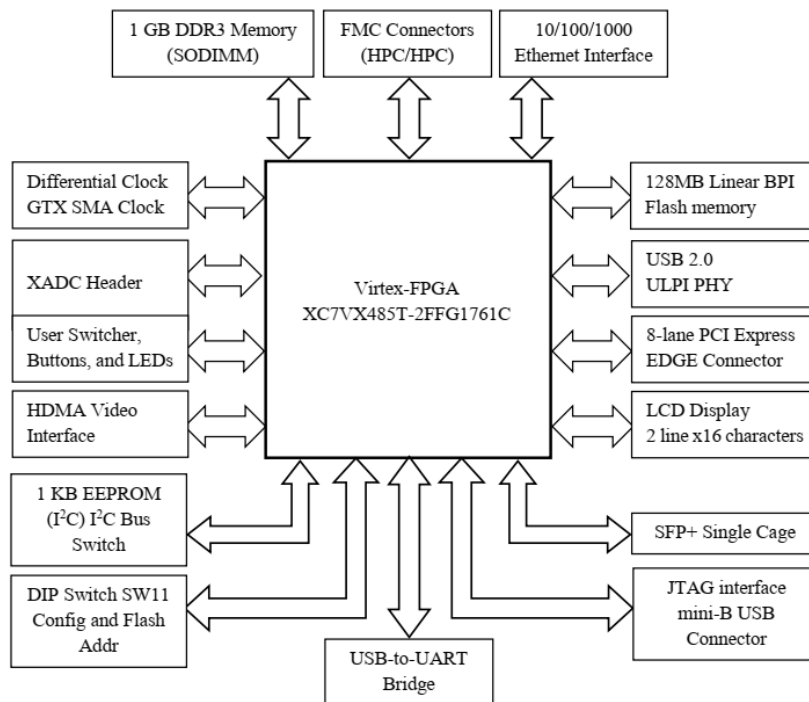


Figure 2. Diagram of board virtex-7 VC707

3. SYSTEM ANALYSIS AND DESIGN

3.1. Implementation of mean filter on FPGA

The mean filter is used to calculate the pixel value of a certain pixel of the final image. The values of column and row are given as 5 bits with the address of the memory location of the output component of BRAM, respective of 9 Pixel values are retrieved from the input, address of the pixel values in input BRAM is shown in Figure 3. Overview diagram of the averaging filter block is shown in Figure 4.

$$\text{Address of the pixel values in input BRAM} = (\text{column} + \text{col_offset}) \times \text{size of final image} + (\text{row} + \text{row_offset})$$

Figure 3. Address of the pixel values in input BRAM

With that, the variables row_offset and col_offset is used for the purpose of implementing the for loop. And both are initialized to -1 and it loops until the value of row_offset reaches 1. The address value is used to get the pixel value and added to the global variable sum. The final value for the output image is obtained by dividing it by 9 Pixel, which is the kernel size. It is then sent to the output with the address of the output Block RAM.

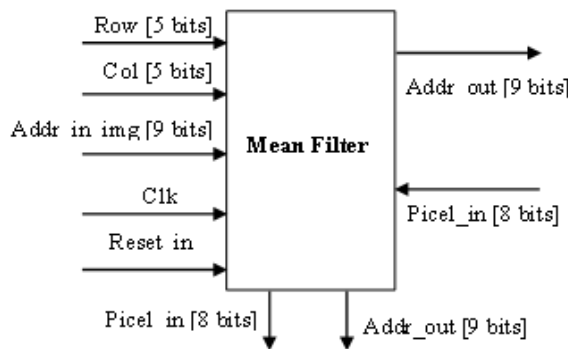


Figure 4. Overview diagram of the averaging filter block

The in/out blocks with high drive of clock buffers, it is located around the chip. The buffers are connected to the clock input pins and drive the clock signals into the global clock lines. The clock lines are designed so that the time offset is minimal and the propagation time is fast. Synchronous design is required for FPGAs since absolute offsets and hysteresis are not guaranteed. Only, when using clock signals from the buffers, relative delay and offset times are guaranteed.

3.2. Systems design in VHDL

The diagram of the high-order architecture of the mean filter in FPGA is shown in Figure 5, it is basic block functions are listed as:

- Scheduler: the image convolution process is managed by the scheduler, when user input is given this process is initiated by the scheduler. And then, the values of pixel of column and row that the average value should be obtained are fed to the mean filter and the init_single_op flag be activated. All the column and row values of the pixels are fed to the mean filter.
- Block RAM: BRAM is used to store large amounts of data inside FPGAs. They are one of four components commonly identified on the FPGAs datasheet. BRAM is created using the block memory generator IP core and it is used to store the input image and the calculated output image.
- UART handler and UART line: the UART processor receives the input image from the Rx line and loads it into the first BRAM. After completing the whole process, the BRAM sends the resulting image to the Tx line. UART Lite communication between the computer and FPGA module and it has four registers as follows. UART communication is achieved by performing read and write operations on the above four registers. The read and write operations are performed using the AXI protocol, which is the protocol used for communication in network-on-chip systems.

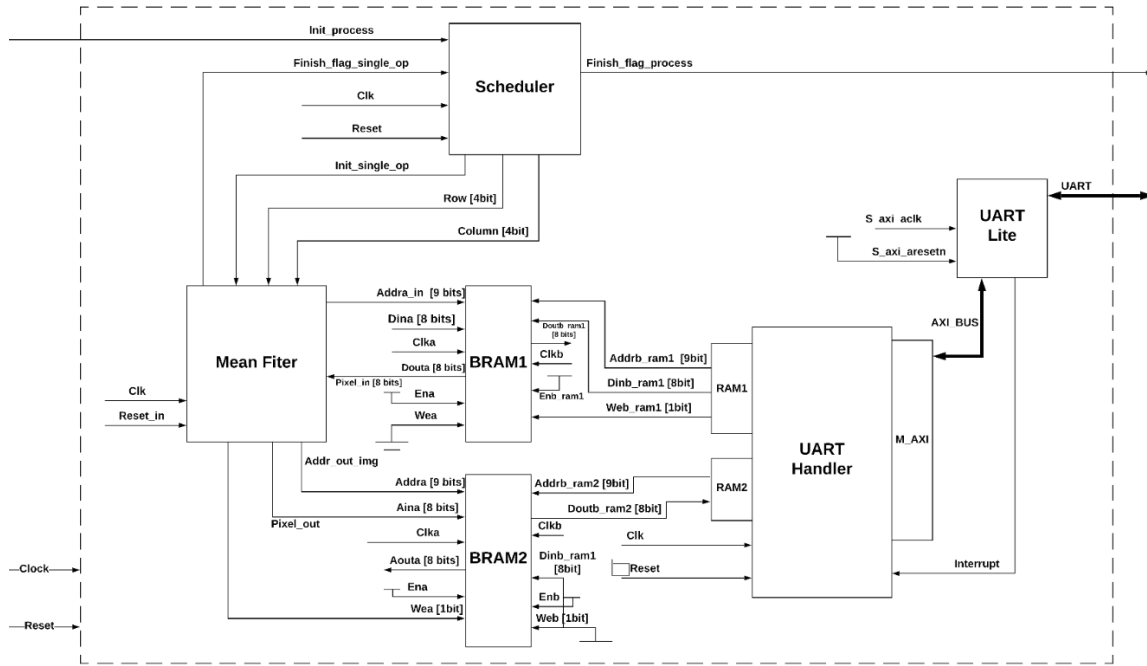


Figure 5. Diagram of the high-order architecture of the mean filter in FPGA

4. RESULTS AND DISCUSSION

After analyzing and designing the mean filter system, we proceed to initialize and build it on Vivado 2020.2 with VHDL language. Using board of Virtex-7 FPGA VC707 and perform edge detection digital image with a grayscale image with width of 512 Pixel and height of 512 Pixel, the pixel count of 512×512 equals 262,144 pixels. Then in these 262,144 pixels, there will be pixels with the same gray intensity. Assuming that, the bit range of image is from 0 to 255, and bit 1 has 150 identical pixels, etc. Continue doing this up to bit 255 which is N identical pixels.

The matrix considered is of size 3×3. So, to count 9 Pixel elements, that need 4 bits (0000 to 1001). The “pixel count” consists of 4 bits used to keep track of the number of pixel elements read before the convolution process. The filtered image is not currently in the usual format to be displayed. Therefore, to be able to compare the results of the image processing algorithm in VHDL with the algorithm in MATLAB, we need to convert the processed image into a normal image format.

Figure 6, illustrates the histogram of the different between VHDL and MATLAB, using Virtex-7 FPGA VC707. This will allow the designer based on those comparison results to be able to make appropriate changes, in order to come up with the most profitable options when designing an image processing system. Looking at the histogram, based on the value columns, it is easy to see that: most of the pixels have values in the range [150, 250]. Compare the results performed on hardware and simulation results on software are similar. This shows that the algorithm built on FPGA using Vivado engine has ensured the correctness of the design.

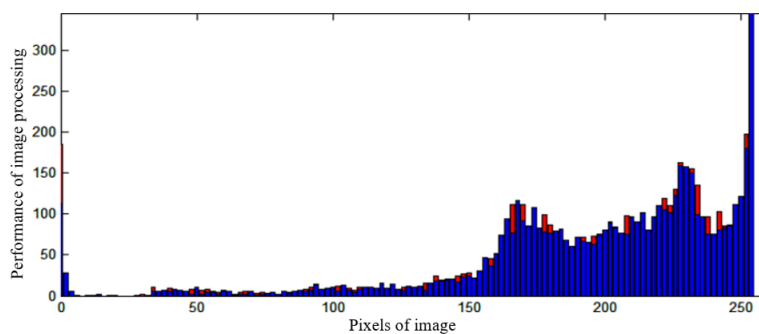


Figure 6. Histogram of the different between VHDL and MATLAB

5. CONCLUSION

This paper design of mean filter using FPGAs for digital images, implementation and analysis of image processing algorithms on FPGAs. The proposed FPGAs implementation takes 0.721 ms, including the write time, SRAM read and the computation time to detect edges of 512×512 images when clocked at 50 MHz. Among hardware solutions for building digital image processing systems, perhaps FPGA is one of the most suitable solutions. Not only the processing results are equivalent to software solutions, but it also meets the simplicity and compactness in terms of hardware. With FPGA, we can design a block-by-block system that works in parallel, allowing processing speed to be increased many times over sequential processing. This is very important for systems that require fast processing speed, such as robotic arms that pick up moving objects.




REFERENCES

- [1] J. Stowers, A. Bainbridge-Smith and M. Hayes, "Beyond optical flow biomimetic UAV altitude control using horizontal edge information," in *Proceedings of the 5th International Conference on Automation, Robotics and Applications*, 2011, pp. 372–377, doi: 10.1109/ICARA.2011.6144912.
- [2] M. Juneja and P. S. Sandhu, "Performance evaluation of edge detection techniques for images in spatial domain," *International Journal of Computer Theory and Engineering*, vol. 1, no. 5, pp. 614–621, 2009, doi: 10.7763/IJCTE. 2009. V1.100.
- [3] L. Guo and S. Wu, "FPGA Implementation of a real-time edge detection system based on an improved canny algorithm," *New Intrusion Detection Technology Driven by Artificial Intelligence*, vol. 13, no. 2, 2023, <https://doi.org/10.3390/app13020870>.
- [4] D. Alghurair and S. S. Al-Rawis, "Design of sobel operator using field programmable gate arrays," *2013 The International Conference on Technological Advances in Electrical, Electronics and Computer Engineering (TAECE)*, May. 2013, pp. 589–594, doi: 10.1109/TAECE.2013.6557341.
- [5] Altera Corporation, *Logic Elements and Logic Array Blocks in Cyclone IV Devices* in Cyclone IV Device Handbook, vol. 1, 2009.
- [6] Altera Corporation, *Cyclone IV FPGA Device Family Overview* in Cyclone IV Device Handbook, vol. 1, 2013.
- [7] N. Gaikwad; V. N. Patil, "Low complexity illumination-invariant motion vector detection based on logarithmic edge detection and edge difference," *IEEE International Symposium on Circuits and Systems (ISCAS)*, Oct. 2020, doi: 10.1109/ISCAS45731.2020.9180935.
- [8] B. Du, Z. Hao and X. Wei, "Roundness detection of end face for shaft workpiece based on canny-zernike sub pixel edge detection and improved hough transform," *IEEE 11th International Conference on Electronics Information and Emergency Communication (ICEIEC)*, 2021, pp. 1–4, doi: 10.1109/ICEIEC51955.2021.9463822.
- [9] C. Hui, B. Xingcan and L. Mingqi, "Research on image edge detection method based on multi-sensor data fusion," *IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA)*, Jun. 2020, pp. 789–792, doi: 10.1109/ICAICA50127.2020.9182548.
- [10] P. Kanchanatripop and D. Zhang, "Adaptive image edge extraction based on discrete algorithm and classical canny operator," *Mathematical Modeling and Computational Methods in Science and Engineering II*, vol. 12, no. 11, pp. 1–15, Oct. 2020, <https://doi.org/10.3390/sym12111749>.
- [11] R. Ramya and P. Babu, "Automatic tuberculosis screening using canny Edge detection method," *2015 2nd International Conference on Electronics and Communication Systems (ICECS)*, Feb. 2015, pp. 282–285, DOI:10.1109/ECS.2015.7124909.
- [12] S. Zheng, J. Guo, H. Yue and X. Liu, "Cross domain edge detection based label decoupling salient object detection," *2021 IEEE 21st International Conference on Communication Technology (ICCT)*, Oct. 2021, pp. 1143–1147, doi: 10.1109/ICCT52962.2021.9657871.
- [13] P. Li, D. J. Lilja, W. Qian, K. Bazargan, and M. D. Riedel, "Computation on stochastic bit streams digital image processing case studies," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 22, no. 3, pp. 449–462, Apr. 2013, doi: 10.1109/TVLSI.2013.2247429.
- [14] Q. Xu, S. Varadarajan, C. Chakrabarti, "A distributed canny edge detector: algorithm and FPGA implementation," *IEEE Trans. Image Process*, vol. 23, no. 7, pp. 2944–2960, Mar. 2014, DOI: 10.1109/TIP.2014.2311656.
- [15] J. D. Owens *et al.*, "A survey of general-purpose computation on graphics hardware," *Computer Graphics Forum*, vol. 26, no. 1, pp. 80–113, 2007, doi: 10.1111/j.1467-8659.2007.01012.x.
- [16] D. Sangeetha and P. Deepa, "FPGA implementation of cost-effective robust Canny edge detection algorithm," *Journal of Real-Time Image Processing*, vol. 16, pp. 957–970, 2019, doi: 10.1007/s11554-016-0582-2.
- [17] I. K. Park, N. Singhal, M. H. Lee, S. Cho, and C. Kim, "Design and performance evaluation of image processing algorithms on GPUs," *IEEE Transactions on Parallel and Distributed Systems*, vol. 22, no. 1, pp. 91–104, 2011, doi: 10.1109/TPDS.2010.115.
- [18] L. H. A. Lourenço, D. Weingaertner and E. Todt, "Efficient implementation of canny edge detection filter for ITK using CUDA," *2012 13th Symposium on Computer Systems*, Petropolis, Brazil, 2012, pp. 33–40, doi: 10.1109/WSCAD-SSC.2012.21.
- [19] J. G. Pandey, A. Karmakar, C. Shekhar and S. Gurunayanan, "A novel architecture for FPGA implementation of Otsu's global automatic image thresholding algorithm," *2014 27th International Conference on VLSI Design and 2014 13th International Conference on Embedded Systems*, Mumbai, India, 2014, pp. 300–305, doi: 10.1109/VLSID.2014.58.
- [20] A. F. Torres-Monsalve and J. Velasco-Medina, "Hardware implementation of ISODATA and Otsu thresholding algorithms," *2016 XXI Symposium on Signal Processing, Images and Artificial Vision (STSIVA)*, Bucaramanga, Colombia, 2016, pp. 1–5, doi: 10.1109/STSIVA.2016.7743329.
- [21] N. Tariq, R. A. Hamzah, T. F. Ng, S. L. Wang, and H. Ibrahim, "Quality assessment methods to evaluate the performance of edge detection algorithms for digital image: a systematic literature review," *IEEE Access*, vol. 9, pp. 87763–87776, Jun. 2021, doi: 10.1109/ACCESS.2021.3089210.
- [22] S. Jingcheng, W. Zhengyan and L. Zenggang, "Implementation of Sobel edge detection algorithm and VGA display based on FPGA," *2019 IEEE 4th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*, Chengdu, China, 2019, pp. 2305–2310, doi: 10.1109/IAEAC47372.2019.8997533.
- [23] H. T. Ngo and V. K. Asari, "A pipelined architecture for real-time correction of barrel distortion in wide-angle camera images," in *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 15, no. 3, pp. 436–444, 2005, doi: 10.1109/TCSVT.2004.842609.




- [24] H. Kim, B.-G. Nam, J.-H. Sohn, J.-H. Woo and H.-J. Yoo, "A 231-MHz, 2.18-mW 32-bit logarithmic arithmetic unit for fixed-point 3-D graphics system," in *IEEE Journal of Solid-State Circuits*, vol. 41, no. 11, pp. 2373-2381, 2006, doi: 10.1109/JSSC.2006.882887.
- [25] D. H. Ai, C. D. Vuong, K. T. Luong and V. T. Le, "Field programmable gate array implementation of edge detection system based on an improved sobel edge detector," *Indonesian Journal of Electrical Engineering and Computer Science (IJECS)*, vol. 32, no. 3, pp. 1378-1383, Dec. 2023, doi: 10.11591/ijeecs.v32.i3.pp1378-1383.
- [26] K. H. Abed and R. E. Siferd, "CMOS VLSI implementation of 16-bit logarithm and anti-logarithm converters," *Proceedings of the 43rd IEEE Midwest Symposium on Circuits and Systems (Cat.No.CH37144)*, Lansing, MI, USA, 2000, pp. 776-779 vol.2, doi: 10.1109/MWSCAS.2000.952871.

BIOGRAPHIES OF AUTHORS






Duong Huu Ai    received the Master of Electronic Engineering from Danang University of Technology, Vietnam, in 2011, and the Ph.D. degree in Electronics and Telecommunications from Hanoi University of Technology, Vietnam, in 2018. Currently, he is a lecturer at The University of Danang, Vietnam-Korea University of Information and Communication Technology, Danang City, Vietnam. His research interests include optical wireless communications, optical and quantum electronics, 5G wireless communications and broadband networks, and IoT. He can be contacted at email: dhai@vku.udn.vn.






Van Loi Nguyen    received his Master of Engineering in Computer Science from the University of Danang, Vietnam in 2010, a Ph.D. degree from Soongsil University in 2017. Currently, he is a lecturer at The University of Danang, Vietnam-Korea University of Information and Communication Technology, Danang City, Vietnam. His research interests include multimedia, information retrieval, artificial intelligence, database, and IoT. He can be contacted at email: nvloi@vku.udn.vn.



Khanh Ty Luong    received his Master of Engineering in Computer Science from the University of Danang, Vietnam in 2012. Currently, he is a lecturer at The University of Danang, Vietnam-Korea University of Information and Communication Technology, Danang City, Vietnam. His research interests include database, artificial intelligence, IoT, and optical wireless communications. He can be contacted at email: lkty@vku.udn.vn.



Viet Truong Le    received his Master of Science in Informatics from Hue University, Vietnam in 2005. Currently, he is a lecturer at The University of Danang, Vietnam-Korea University of Information and Communication Technology, Danang City, Vietnam. His research interests include database, data warehouse, data mining, system analysis, and design. He can be contacted at email: lvtruong@vku.udn.vn.