FPGA Implementation of Archery Target Detection using Color Sequence Recognition Algorithm

Dino Dominic Ligutan, Alexander C. Abad, Melvin Cabatuan, Cesar Llorente, Elmer P. Dadios

Abstract*: In this paper, an implementation of image processing methods to extract and recognize a standard tri-colored archery target to a field-programmable gate array is demonstrated. Detection and recognition of the archery target was never been done on an FPGA platform. The platform used to realize the design was the ZedBoard™ Development Kit equipped with Xilinx Zynq®-7000 All Programmable system on chip. The algorithms used to extract the central region is based on color classification in HSV color space. Once each image pixels are classified, the color sequence recognition algorithm attempts to look for the target and extract the central region of the archery target if present. Image filtering techniques and analysis such as morphological filtering and contour feature analysis are used to properly identify the shape and location of the extracted pixels. Discussed next is the implementation of the algorithm both in the software and hardware aspects and a comparison between their response time and accuracy is demonstrated. There was about two-fold decrease in processing time when FPGA implementation was deployed. The accuracy of the system was also tested and able to reach an accuracy of 96.67% for near target distance. For far target distance, the accuracy degraded to 88.33% but the system has managed to maintain its specificity value despite the noise becoming dominant for smaller region occupied by the target.

Keywords: Archery target, Color classification, Color sequence recognition, Field-programmable gate array, Image processing.

1. INTRODUCTION

A growing interest in the field of robotics is to mimic the human abilities and ultimately outperform them in different skills such as shooting an arrow accurately to a target. In such a robot, an ability to detect and recognize the central golden region of the archery target is of utmost importance to meet these human feats. A study in [1] shows a humanoid robot capable of learning to shoot more accurately as it is exposed to archery shooting tasks. Demonstration of this capability of robots was also demonstrated in [2]. The objective if these studies can only be achieved if appropriate image processing of the target and the arrow were available and reliable. As such, appropriate image processing methods are needed to realize this objective. There are a handful of studies on image processing methods applied to the archery target. Most of these studies pertains to automatic scoring system; that is, to provide an automated solution to determine the score of a shot arrow on a target. The study in [3] describes an algorithm to rectify the archery target for different viewing angles as well as recognize the shot arrows embedded within and by approximating their positions relative to the target center as a basis for scoring. Detecting the point at which the arrow struck the target is a non-trivial task for image processing and is very much affected by many factors such as lighting condition and viewing angle. Another automatic scoring system by [4] provides a very different approach in scoring arrows that is dependent on background subtraction and morphological operations to detect the target and then the arrow through the process of arrow head nomination. However, the study used a much higher frame resolution and achieved a higher accuracy rate than the former. This study was taken further by proposing a new algorithm for real-time arrow detection and scoring using the concept of score region images. [5] All of these studies however are limited to stationary archery targets only. Furthermore, the systems developed are implemented on a conventional computing device that operates in full dependence of software algorithms.

On the other hand, there are numerous image processing algorithms were already implemented on FPGA to take advantage of parallel processing that it offers. The study on [6] has implemented several image processing modules on an FPGA that can be used for a variety of settings. The system was reconfigurable and specific image processing can be chosen easily depending on the application at hand. Similar purpose was also achieved in [7] is optimized through implementation of different architectures at the block level. Besides the need for faster color recognition, FPGA was utilized not only to recognize colors but to track motion as well [8] to serve as an input for robot formation control to guide their motion in real-time. Another study [9] also demonstrated the possibility of an efficient color conversion from YCbCr to RGB so that it becomes easier for standard algorithms to identify and calibrate color detection for different display devices. Color recognition and classification has diverse applications that several studies were conducted [10]–[13] due to the necessity of a far more faster processing of continuous stream of images on real-time applications.

Revised Version Manuscript Received on August 19, 2019.

Dino Dominic Ligutan, ECE Department, De La Salle University, Manila, Philippines.
Alexander C. Abad, ECE Department, De La Salle University, Manila, Philippines.
Melvin Cabatuan, ECE Department, De La Salle University, Manila, Philippines.
Cesar Llorente, ECE Department, De La Salle University, Manila, Philippines.
Elmer P. Dadios, MEM, Department, De La Salle University, Manila, Philippines.

This study focuses on the detection and recognition of the archery target central area as well as its implementation on an FPGA. FPGA has proven to be capable of implementing software algorithms way more faster [14]–[19] and as such has found numerous applications in many fields of study as demonstrated by the studies [20]–[24]. Application of image processing methods for archery target detection and recognition has never been implemented on an FPGA until now. Implementation on FPGA will take advantage of its parallel architecture and will significantly speed up the algorithm implementation as against to full software implementation. The challenge met with this setup is that an algorithm that must be devised should only take only a few clock cycles of the FPGA in order to achieve real-time application. This is further complicated by the limited amount of resources that the FPGA has to offer and using any floating-point operations must be avoided to meet real-time processing as much as possible. The next chapter shall discuss the setup of the system, followed by the description of the algorithm and implementation on the FPGA. Finally, the results, conclusions and recommendations are laid out to demonstrate the effectiveness of the algorithm as well as to enumerate the areas of improvement for the study.

II. THE PROPOSED METHOD

A.Hardware Setup

The hardware that will serve as a platform for this study is the ZedBoard™ Development Kit [25] board using the Xilinx Zynq®-7000 All Programmable system on chip (SoC) as its main processing device. The board offers built-in peripheral connections such as USB On-The-Go (OTG) and UART ports, HDMI and VGA display ports, audio processing jack ports, SD Card slot as well as general purpose I/O pins that can be used for expansion. The main processing device contains dual-core ARM Cortex-A9 processors (PS) integrated with 28nm Artix-7 or Kintex®-7 based programmable logic (PL). [26] The combination allows for the design of accelerated software algorithms with the aid of hardware-based accelerator system.

The main sensory input is provided by a generic web camera connected to a USB Hub. The USB Hub, connected to the USB OTG port, serves as an expansion for USB web camera, mouse and keyboard multiplexing. The web camera supports 640 by 480 resolution at a frame rate of 30 frames per second. The PS communicates with the web camera through the Multiplexed I/O interface and acquires each frame suitable for processing. The VGA display adapter is directly connected to the PL and as such, the PS must interact with the PL for proper display. The system setup is shown in Fig.1.

The ZedBoard is also capable of functioning as a stand-alone graphical desktop computer running an operating system such as Linux. The SD Card inserted to the board provides the bootable media as well as the storage medium for the operating system. The SD Card is installed with a pre-configured Linux OS named Xillinux by Xillybus Ltd. [27]Xillinux comes with a pre-configured device driver files that represents communication interface with the FPGA. Thus, sending and receiving data to and from the FPGA is just a matter of writing and reading to device files on the software side.

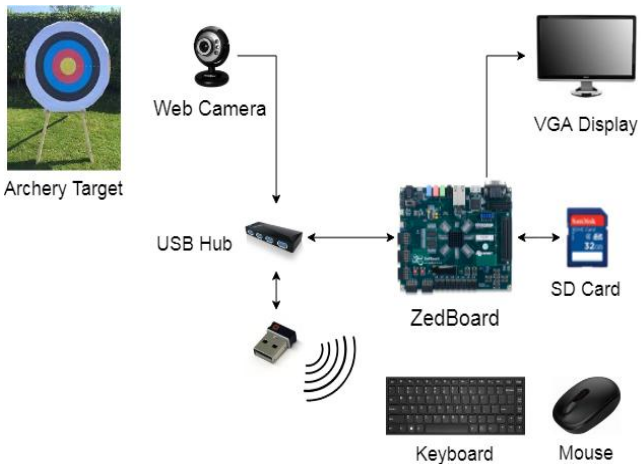


Fig.1. Zedboard setup and connections of external devices

B. Color Classification and Sequence Recognition

The archery target is predominantly color-coded to facilitate the scoring of shot arrow. As such, the colors present on the target will be exploited to aid in a more robust recognition algorithm. The first process involves transforming each pixel values from RGB to HSV color-space. The transformation to HSV color-space allows a more convenient way to filter the image by setting proper hue range values. For the archery target, three different colors can be effectively filtered: blue, red and yellow. The appropriate hue values are determined through test and experiments and by subjecting the archery target to different levels of illumination. Table 1 summarizes the values of HSV ranges for each of three colors.

Table 1. Specific HSV range values for three different archery target colors.

Color	Blue	Red	Yellow
Hue	192° - 252°	0° - 28°; 318° - 360°	40° - 104°
Saturation	90 - 255	74 - 255	60 - 255
Values	0 - 255	0 - 255	45 - 255

In much the same way that each of the colors corresponds to a score when hit by the arrow, each of the pixels are re-assigned a single value corresponding to which color it is classified. For instance, a pixel classified as blue is assigned a value of 1, red assigned with 2, yellow assigned with 3 and a 0 if it does not belong to any of the colors aforementioned.

The next process involves scanning the whole image pixel by pixel to look for specific sequence of colors. Since the sequence of colors for the archery target is well defined, the central target can be recognized by recognizing the proper sequence of the colors as each frame is scanned pixel by pixel. No matter which direction the scan is made, the resulting sequence will be invariant by virtue of the archery target being composed of concentric circles of colors; blue

followed by red then followed by yellow. The process of recognizing a color sequence is detailed in Algorithm 1.

Algorithm 1. Color sequence recognition algorithm

Input: Input image I of a video frame, minimum value m of accumulated correct color pixels and maximum value M of accumulated incorrect color pixels

Output: Output image whose pixels are re-assigned a value proportional to the probability that the target's central region is in that pixel location

Precondition: Each pixel p_i of image I is already assigned a color accordingly by color classification procedure

Let color Blue has Index 1, color Red has Index 2, color Yellow has Index 3 and other colors has Index 0

Set Color to search as Blue

Set Accumulated number of pixels with the same color as currently searched color to Zero

Set Accumulated number of pixels with different color from the currently searched color to Zero

For each rows of I **do**

For each columns of I **do**

If color of pixel p_i is the same as the color to search **then**

Set pixel p_i value to the index value of that color

Increment accumulated number of pixels with same color

Reset accumulated number of pixels with different color to Zero

Else If color of pixel p_i is the same as the next color to search **then**

If accumulated number of pixels with same color $\geq m$ **then**

Set color to search to next color to search

Reset accumulated number of pixels with same color to Zero

Reset accumulated number of pixels with different color to Zero

Set pixel p_i value to the index of the next color

End If

Else

Increment accumulated number of pixels with different color

End If

If accumulated number of pixels with different color $\geq M$ **then**

Set color to search back to Blue

Reset accumulated number of pixels with same color to Zero

Reset accumulated number of pixels with different color to Zero

End If

End For

End For

The algorithm attempts to look for regions of the image where the sequence of colors Blue, Red and Yellow holds if it is scanned from left to right. If a correct sequence is found, then the subsequent pixels are assigned a higher index value up until it encounters a yellow colored pixel given an index value of 3. It is worth mentioning that if the scan is made right to left, then the same color sequence must be expected. The same procedure is done to scan the pixels top to bottom and bottom to top as well. In the process, each scanned pixel accumulates a value assigned by the four scanning process and the group of pixels with the highest index values must be the region where the central yellow region of the archery target can be found in the image. The next process involves thresholding where only pixels with higher index are passed through to obtain an outline of the central region of archery target. The result of such process is illustrated in Fig. 2.

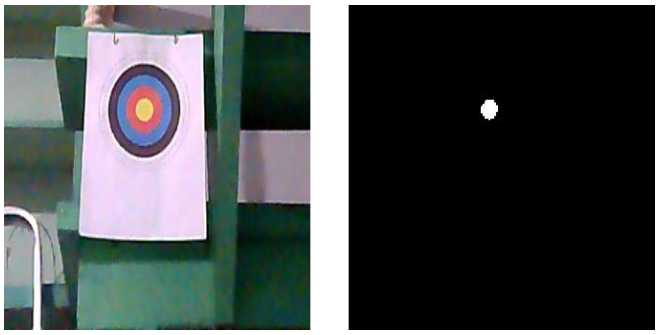


Fig. 2. Results of color classification and sequence recognition: Original image on the left and extracted central region image on the right.

Once the central yellow region of the archery target is extracted, morphological closing operation is performed to remove any holes within the extracted region as well as eliminating any pixels that are considered as noise. Once the image is conditioned, contours are extracted and its features are analyzed. By using the OpenCV functions for contour feature analysis [28], the shape of the extracted region can be determined and it ensures that only a circular or elliptical shaped region is passed through as the actual archery target position.

III. METHOD

In order to facilitate detection and recognition of the archery target, a portion of the image processing is to be executed by the FPGA to speed up the process. Programming the FPGA allows an inherently parallel processing feature to speed up the process of extracting the central yellow region. However, programming the FPGA requires an extensive knowledge on logic circuits designs as well as proficiency in Hardware Description Language (HDL) or Register-Transfer Level (RTL). Moreover, the algorithm is conceptualized with the C programming language in mind. Fortunately, Xilinx provides a tool to easily convert C code into HDL using High Level Sythesis Tools. One such tool is the Vivado HLS [29] which allows the designer to write a C code then does the translation to HDL/RTL. Moreover, the tool also provides directives that allows the user to pipeline sequential operations in the C language for faster processing, thus providing flexibility beyond the capabilities of the C

language itself. Once the HDL is ported, the code is then used to program a block in non-HLS Vivado provided by the Xillybus demo bundle. The non-HLS Vivado's task is to generate a bitstream file to be uploaded to the FPGA to configure its logic fabric.

The distribution and sequence of operations being performed from the retrieval of the frame form the web camera to the extraction of the coordinates of the archery target is shown in Fig. 3. Color classification as well as color sequence recognition are done by the FPGA because these processes can be easily pipelined in a pixel-by-pixel basis. On the other hand, OpenCV functions, such as morphological filtering and contour feature analysis are done by the processing system once the extracted image is sent back by the FPGA.

The first step transmits the HSV color classification parameters to the FPGA to allow convenient change of parameters without reprogramming the FPGA. Afterwards, a frame from web camera is retrieved and stored temporarily in an array already in the BGR format specification of OpenCV Matrix data type. This array is immediately sent to the FPGA blocks and is processed by the FPGA pixel-by-pixel.

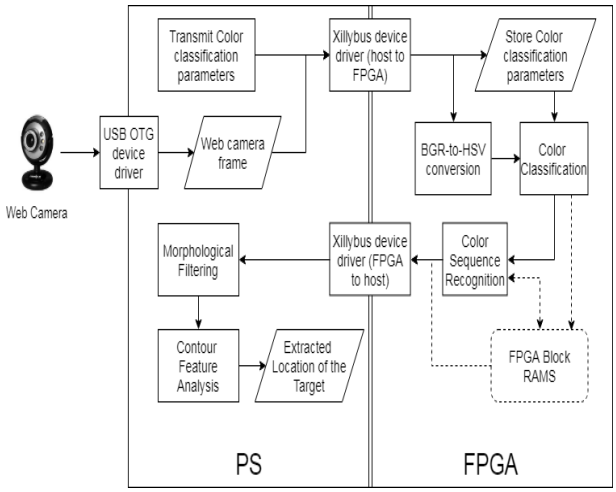


Fig. 3. Flow chart and distribution of process for archery target detection

The FPGA is configured as a state machine that expects to process a total of 640x480 = 307,200 pixels for each frame. The result of color classification and sequence recognition for each pixels is stored internally in FPGA block RAMs. A received pixel are group into threes representing the blue, green and red components and is fed to the color classification block. The block gives out a value according to the classification of the pixel's color. As to the implementation of the color classification block, the standard conversion formulas were avoided to minimize processing delay. As such, integer based conversion as presented in [30] was utilized to avoid floating point operations that may cause processing delays.

Since four passes of scanning are required for each frame (left-to-right, right-to-left, top-to-bottom and bottom-to-top), there are four internal arrays that stores the classified color frames. The logic in FPGA multiplexes between these four arrays to store the immediate results of color classification. The result of color classification goes to color sequence recognition (CSR) block and is first scanned left to right then stored in the first array. On the next frame, the results of color classification is stored on the second array, and so on until the cycle repeats. As the frame is stored on the second array the first array is simultaneously scanned top to bottom. Upon processing of third frame, the first array is now being scanned right to left. Finally, on the fourth frame, the first array is being scanned bottom to top. Once the cycle repeats, the first array now contains the required extracted image and then sent back to host before a new frame is written to it. This sequence is done to all the four arrays simultaneously, thus pipelining the process and ensuring that the flow of data is unnecessarily delayed due to the scanning processes if all are done sequentially for each frame received.

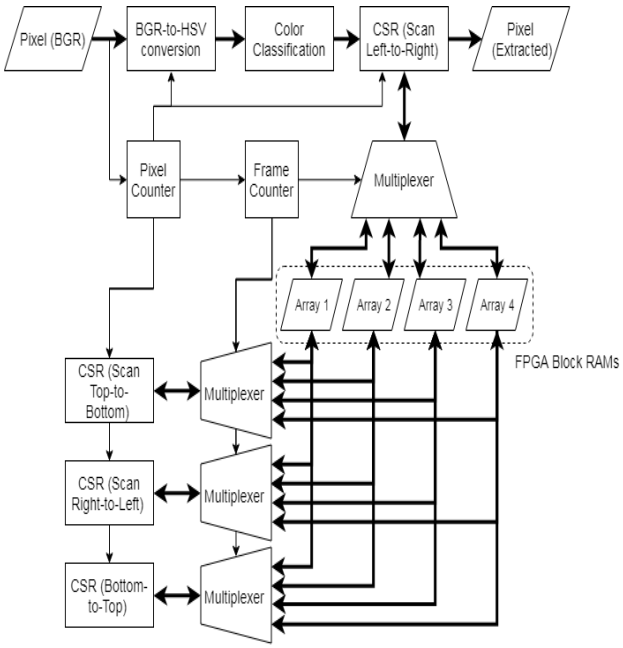


Fig. 4. Simplified FPGA logic diagram for storing and processing image frames

The processing can still be simplified and made faster if only two scanning process were to be made; left-to-right and top-to-bottom. However, this sacrifices the accuracy of the algorithm as it may falsely detect a figure with the same color sequence left-to-right but of different sequence right-to-left.

Once the extracted image is sent by the FPGA to host, the software fetches the data back into an OpenCV Matrix data type. The software does not wait for the extracted image to come out, rather the software program must be multithreaded to prevent delays due to waiting for the FPGA to send some data back. The main thread takes care of sending the retrieved web camera images and a child thread polls the device file for incoming data. If an image is received, the morphological closing operation is applied to eliminate noises and at the same time close any holes that may be present on the extracted central region. The next process extracts the contours present in the image and determines the number of points as well as the area of each detected contour. By specifying a lower bound to the areas of the contour, any small speckles of the image due to noise may be eliminated.

Additionally, if a contour is truly circular in shape, then the following equation must be close to zero as much as possible:

$$P^2 - 4\pi A \approx 0 \quad (1)$$

where P is the perimeter of the contour and A is the area of the contour. Once the above requirement is met, a minimum enclosing circle is calculated, thus giving us the center coordinates as well as the radius of the extracted central region of the archery target.

IV. RESULTS AND DISCUSSION

A software only implementation of the algorithm will serve as a basis to test if co-processing with FPGA has indeed improved the processing time from retrieval of a frame to getting the coordinates of the central region of the archery target. The presence of target is tested if it affects the overall processing time. Additionally, the accuracy of recognition is tested, and a confusion matrix is obtained to characterize its robustness.

At least 30 trials of time measurement were made both for the software only implementation and with co-processing FPGA. The results indicate that whether the target is present or not, the software only implementation took about twice the time than that with co-processing implementation. This means that there is about twice speed up improvement with FPGA. Table 2 summarizes the results of time measurement taken between the times to load the image frame up to extraction of the central region of archery target. It can also be noted that with the presence of target, the processing with FPGA was a bit slower compared to the one without the target but still is way faster than with software only implementation.

Table 2. Comparison of processing time results between two implementations

Implementation	Software Only		With Co-processing	
	No Target	With Target	No Target	With Target
Average Time	0.3342 s	0.3359 s	0.1139 s	0.0763 s
Std. Deviation	0.0156	0.0092	0.1085	0.0601

Based from the table shown, the standard deviation of processing time with co-processing is noticeably higher than compared with software only implementation. This is an indication that there is a significant variation on time measurements. To conclusively show that the co-processing implementation is indeed faster, images were fed to the FPGA manually upon press of the button to measure the time of a single frame for at least another 30 trials.

Table **3** shows that a non-continuous feed of data results in less variation of measurements if no target is present. The variation for continuous mode was due to the task scheduling being controlled by the context switcher of the OS. The case is different if a target is present presumably due to different processing times required for detecting and recognizing archery target at different locations.

Table 3. Time measurements for continuous and non-continuous co-processing modes

Implementation	Without Target		With Target	
	Continuous	Manual	Continuous	Manual
Average Time	0.1139 s	0.3359 s	0.0763 s	0.0763 s
Std. Deviation	0.1085	0.0092	0.0601	0.0601

The accuracy of the algorithm with co-processing implemented is also tested by capturing at least 30 images each for with or without the target present. A confusion matrix is obtained based from these images whether the system classified that a target is present or not. The actual target is about a meter away from the camera thus making it easier to detect.

Table 4. Confusion matrix for a close archery target detection

	Predicted Absent	Predicted Present
Actual Target Absent	28	2
Actual Target Present	0	30

Based from Table 4, we can calculate different parameters for the performance of the system. The accuracy is about 96.67%, precision is about 93.75%, sensitivity is 100 % and specificity is about 93.33%. To test the accuracy of the system for targets farther away from the camera, additional 30 images were taken with the target about 3 meters away. Table 5 summarizes the results for far archery target detection.

Table 5. Confusion matrix for a far archery target detection

	Predicted Absent	Predicted Present
Actual Target Absent	28	2
Actual Target Present	5	25

In this case, the accuracy is about 88.33%, precision is about 92.59%, sensitivity is about 83.33% and specificity is about 93.33%. As expected, the performance of the system degraded as the distance was increased because the target itself occupies a small region in the image and its color may easily be affected by noise. As the camera is a consumer grade low resolution camera, the presence of strong noise is difficult to avoid and may cause false positive results.

V. CONCLUSION

This study has successfully implemented a color classification and sequence recognition algorithm on an FPGA. Results show that there is a twofold increase in

performance with the use of FPGA as against to software only implementation. Despite the measured variations on time measurements for co-processing implementation, the overall time is still way faster than without FPGA co-processing. The accuracy of the system was also tested and able to reach an accuracy of 96.67% for near target distance. For far target distance, the accuracy degraded to 88.33% but the system has managed to maintain its specificity value despite the noise becoming dominant for smaller region occupied by the target.

For this implementation, only two scan directions were utilized due to limitations on FPGA resources. However, it is possible to implement the system in its full functionality by resizing the retrieved image from the webcam into a smaller resolution before being fed to the FPGA to reduce memory footprint and to manually route the components in the FPGA itself without relying on Vivado HLS. When generating a bit stream for the FPGA, not all timing constraints were met, despite reducing the word width of each pixel from 8 bits down to 2 bits.

Regarding the accuracy of detection, an adaptive algorithm to detect the brightness of the image is needed so that colors are adaptively filtered. A system that can store the last known coordinate of the system as well as its predicted trajectory will solve the problem of archery target getting lost upon the glare of reflections or strong light sources. Finally, implementing a direct interfacing FPGA-based hardware to communicate with camera will drastically improve the response time of the system.

VI. ACKNOWLEDGMENT

The authors would like to acknowledge De La Salle University – Manila (DLSU) and Department of Science and Technology – Engineering Research and Development for Technology (DOST-ERDT) for funding and helping us to finish this study.

REFERENCES

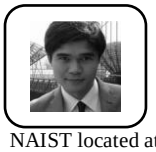
1. P. Kormushev, S. Calinon, R. Saegusa, and G. Metta, "Learning the skill of archery by a humanoid robot iCub," in *2010 10th IEEE-RAS International Conference on Humanoid Robots*, Nashville, TN, USA, 2010, pp. 417–423.
2. "i-SOBOT." [Online]. Available: http://www.isobotrobot.com/eng/about/whats/index.html. [Accessed: 22-Jul-2018].
3. C. Nguyen and I. Lin, "Arrowsmith: Automatic Archery Scorer."
4. T. Zin, I. Oka, T. Sasayama, S. Ata, H. Watanabe, and H. Sasano, "Image Processing Approach to Automatic Scoring System for Archery Targets," in *2013 Ninth International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, Beijing, China, 2013, pp. 259–262.
5. K. Shiiya, T. T. Zin, M. Jomoto, and H. Watanabe, "A study on automatic display system of the archery score for the visually impaired," in *2017 IEEE 6th Global Conference on Consumer Electronics (GCCE)*, Nagoya, 2017, pp. 1–2.
6. J. G. Mertes, N. Marranghello, and A. S. Pereira, "Real-time Module for Digital Image Processing Developed on a FPGA," *IFAC Proc. Vol.*, vol. 46, no. 28, pp. 405–410, 2013.
7. A. F. Dias, C. Lavarenne, M. Akil, and Y. Sorel, "Optimized implementation of real-time image processing algorithms on field programmable gate arrays," in *ICSP '98. 1998 Fourth International Conference on Signal Processing (Cat. No.98TH8344)*, Beijing, China, 1998, vol. 2, pp. 1080–1083.
8. Y.-H. Yu, N. M. Kwok, and Q. P. Ha, "FPGA-Based Real-Time Color Tracking for Robotic Formation Control," presented at the 26th International Symposium on Automation and Robotics in Construction, Austin, TX, USA, 2009.
9. G. Sun, Z. Weng, P. Zhao, D. Guo, Y. Tian, and L. Xiao, "Design of Color Recognition System Based on FPGA," in *Proceedings of the 2016 International Conference on Electrical, Mechanical and Industrial Engineering*, Phuket, Thailand, 2016.
10. C.-L. Yang and B. Wang, "Fuzzy algorithm for color recognition and realization on FPGA," in *2015 IEEE 10th Conference on Industrial Electronics and Applications (ICIEA)*, Auckland, New Zealand, 2015, pp. 1529–1533.
11. H. Dong, Y. Zhang, M. Chen, and W. Jin, "Design of the Image Acquisition and Processing System for Color Sorter Based on FPGA," in *2017 2nd International Conference on Cybernetics, Robotics and Control (CRC)*, Chengdu, 2017, pp. 184–188.
12. M. Nasir Bin Mohamed Shukor, L. Hai Hiung, and P. Sebastian, "Implementation of Color Filtering on FPGA," in *2007 International Conference on Intelligent and Advanced Systems*, Kuala Lumpur, 2007, pp. 803–805.
13. W. Farhat, H. Faiedh, C. Souani, and K. Besbes, "Real-time hardware/software co-design of a traffic sign recognition system using Zynq FPGA," in *2016 11th International Design & Test Symposium (IDT)*, Hammamet, Tunisia, 2016, pp. 302–307.
14. D. Jinghong, D. Yaling, and L. Kun, "Development of Image Processing System Based on DSP and FPGA," in *2007 8th International Conference on Electronic Measurement and Instruments*, Xian, China, 2007, pp. 2-791-2–794.
15. M. Birla, "FPGA Based Reconfigurable Platform for Complex Image Processing," in *2006 IEEE International Conference on Electro/Information Technology*, East Lansing, MI, USA, 2006, pp. 204–209.
16. A. Sergiyenko, P. Serhiienko, and J. Zorin, "High Dynamic Range Video Camera with Elements of the Pattern Recognition," in *2018 IEEE 38th International Conference on Electronics and Nanotechnology (ELNANO)*, Kiev, 2018, pp. 435–438.
17. W. Zhou, C. Lyu, X. Jiang, P. Li, H. Chen, and Y.-H. Liu, "Real-time implementation of vision-based unmarked static hand gesture recognition with neural networks based on FPGAs," in *2017 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, Macau, 2017, pp. 1026–1031.
18. T. Saegusa and T. Maruyama, "Real-Time Segmentation of Color Images based on the K-means Clustering on FPGA," in *2007 International Conference on Field-Programmable Technology*, Kitakyusyu, Japan, 2007, pp. 329–332.
19. Y. Han and E. Oruklu, "Real-time traffic sign recognition based on Zynq FPGA and ARM SoCs," in *IEEE International Conference on Electro/Information Technology*, Milwaukee, WI, USA, 2014, pp. 373–376.
20. I. Chiuchisan, "A new FPGA-based real-time configurable system for medical image processing," in *2013 E-Health and Bioengineering Conference (EHB)*, IASI, Romania, 2013, pp. 1–4.
21. D. Patel, R. Parmar, A. Desai, and S. Sheth, "Gesture recognition using FPGA and OV7670 camera," in *2017 International Conference on Inventive Systems and Control (ICISC)*, Coimbatore, India, 2017, pp. 1–4.
22. R. Shandilya and R. K. Sharma, "FPGA implementation of image enhancement technique for Automatic Vehicles Number Plate detection," in *2017 International Conference on Trends in Electronics and Informatics (ICEI)*, Tirunelveli, 2017, pp. 1010–1017.
23. S. Blokzyl, M. Vodel, and W. Hardt, "FPGA-based approach for runway boundary detection in high-resolution colour images," in *2014 IEEE Sensors Applications Symposium (SAS)*, Queenstown, New Zealand, 2014, pp. 59–64.
24. P.-C. Cho, C.-T. Li, and W.-H. Chen, "Implementation of Low-Cost Vision-Based Gesture Recognition Systems Based on FPGA Approach," in *2012 International Symposium on Computer, Consumer and Control*, Taichung, Taiwan, 2012, pp. 329–332.
25. "ZedBoard." [Online]. Available: http://www.zedboard.org/product/zedboard. [Accessed: 11-Apr-2018].
26. "Zynq-7000 All Programmable SoC." [Online]. Available: https://www.xilinx.com/products/silicon-devices/soc/zynq-7000.html. [Accessed: 11-Apr-2018].
27. "Xillinux: A Linux Distribution for Zedboard, Zybo, MicroZed and SocKit." [Online]. Available: http://xillybus.com/xillinux. [Accessed: 11-Apr-2018].

28. "OpenCV: Contours in OpenCV." [Online]. Available: https://docs.opencv.org/3.1.0/d3/d05/tutorial_py_table_of_contents_contours.html. [Accessed: 12-Apr-2018].
29. "Vivado High-Level Synthesis." [Online]. Available: https://www.xilinx.com/products/design-tools/vivado/integration/esl-design.html. [Accessed: 12-Apr-2018].
30. V. Chernov, J. Alander, and V. Bochko, "Integer-based accurate conversion between RGB and HSV color spaces," *Comput. Electr. Eng.*, vol. 46, pp. 328–337, Aug. 2015.

AUTHORS PROFILE

Dino Dominic Ligutan earned his Bachelor of Science in Electronics and Communications Engineering at De La Salle University (DLSU) Manila. He is currently taking up his Masters in ECE at DLSU as a DOST-ERDT scholar. His research interests includes robotics, machine vision, machine learning and artificial intelligence.

Alexander C. Abad is a graduate of BS Electronics and Communications Engineering at St. Louis University-Baguio City. He finished his MS in Electronics and Communications Engineering at De La Salle University - Manila and currently taking up his PhD in ECE at DLSU. His field of interest includes robotics, machine intelligence, machine vision, mixed-signal electronics and IC design.



Melvin Cabatuan received the Bachelor of Science degree in Electronics and Communications Engineering from Cebu Institute of Technology University, Cebu, Philippines, in 2004; Master of engineering degree from NAIST located at Ikoma, Nara, Japan, in 2010, and Ph.D.

in Electronics and Communications Engineering (ECE) in De La Salle University in 2016. He joined the Electronics & Communications Engineering department of De La Salle University-Manila, Manila, Philippines in 2011, where he is currently an Associate Professor. His research interest involves machine learning applications to engineering problems. Dr. Cabatuan is an active member of the Institute of Electronics Engineers of the Philippines (IECEP) since 2004, and the current secretary of IEEE Republic of Philippines Section.

Cesar Llorente holds a Certificate in Machine Shop Technology from Negros Occidental College of Arts and Trade, a diploma in Electronics Engineering Technology from Technological University of the Philippines, a degree of Bachelor of Science in Electrical Engineering from Adamson University, a Master of Engineering from Pamantasan ng Lungsod ng Maynila and recently defended his dissertation in PhD-ECE from De La Salle University. After a stint in the semiconductor industry, he spent a good number of his youth in educating junior high school students in the practical arts of electronics and young electronics and computer engineering students at Don Bosco Technical College. He joined the ECE Department of DLSU last 2007 and served as the Program Coordinator of the Computer Engineering Program, and currently as Chair of the Electronics and Communications Engineering department. Engr. Llorente specializes in microcontrollers and FPGA-based embedded systems and in soft computing where he applies computational intelligence for the control of lighting systems and in optimizing land use and minimizing environmental impacts for hybrid electric power generation and in sizing hybrid renewable energy systems. He was a member of the team who build SIKAT and SINAG, the first two solar cars of the Philippines by DLSU. He is a member of Integrated Institute of Electrical Engineers, IEEE and a member of the Technical Committee on Computer Engineering of the Commission in Higher Education.

Elmer P. Dadios earned his Bachelor of Science in Electrical Engineering from the Mindanao State University in Marawi City, Philippines; and his Master of Science in Computer Science from De La Salle University (DLSU) Manila. He got his PhD from the Manufacturing Engineering, Loughborough University, United Kingdom. He is a consultant on software and hardware development in the area of robotics and intelligent systems application. His research interests include: artificial intelligence, evolutionary systems, fuzzy logic, neural networks, Robotics, Mechatronics and Manufacturing Processes. He joined DLSU in May 1992 and currently holds the rank of Full Professor 10 and a University Fellow. He edited and published four books on fuzzy logic systems with InTech (Croatia); He published over 290 technical papers in highly refutable journals and IEEE Xplore. Currently, he serves as editor of the Journal of Advanced Computational Intelligence and Intelligent Informatics (JACIII) published by the Fuji Technology Press, Ltd. (Tokyo, Japan); He is also the editor in chief of the Journal of Computational Intelligence and Engineering Application (JCIEA) published by DLSU Press, Manila, Philippines. He is a member IEEE R10 EXCOM and the chair of the IEEE Asia and Pacific Awards and Recognition Committee. Elmer has earned over 30 recognitions and distinctions from various international and national scientific award-giving bodies and professional organizations. Among the awards he garnered include: The 2019 top 100 scientists listed in Asian Scientist Magazine; The 2019 Leaders in Innovation Fellowship "Fellow" given by the United Kingdom Royal Academy of Engineering; The 2018 D. M. CONSUNJI AWARD for Engineering Research. Achievement Award from the National Research Council of the Philippines (NRCP), and an Outstanding Scientific Paper Award for his "Analysis of Colonic Histopathological Images Using Pixel Intensities and Hough Transform" from the National Academy of Science and Technology (NAST), Philippines. He was also a recipient of the Department of Science and Technology (DOST) 50 Men and Women of Science and Technology; The Department of Science and Technology (DOST) Scholar Achievers. He was the keynote speaker in the 2018 International Winter Symposium on Big-Data, Cybersecurity and IoT in Hokkaido University, Sapporo Japan; a keynote speaker in the the 6th International Conference on Robot Intelligence Technology and Applications (RITA2018) held in Putrajaya, Malaysia.