Development of Sign Signal Translation System Based on Altera's FPGA DE2 Board

Kuo Chue Neo *nkc99131@student.usm.my*
School of Electrical and Electronic Engineering,
Engineering Campus, Universiti Sains Malaysia,
14300 Nibong Tebal, Penang, Malaysia

Haidi Ibrahim *haidi_ibrahim@ieee.org*
School of Electrical and Electronic Engineering,
Engineering Campus, Universiti Sains Malaysia,
14300 Nibong Tebal, Penang, Malaysia

Wan Mohd Yusof Rahiman Wan Abdul Aziz *wanrahiman@eng.usm.my*
School of Electrical and Electronic Engineering,
Engineering Campus, Universiti Sains Malaysia,
14300 Nibong Tebal, Penang, Malaysia

Abstract

The main aim of this paper is to build a system that is capable of detecting and recognizing the hand gesture in an image captured by using a camera. The system is built based on Altera's FPGA DE2 board, which contains a Nios II soft core processor. Image processing techniques and a simple but effective algorithm are implemented to achieve this purpose. Image processing techniques are used to smooth the image in order to ease the subsequent processes in translating the hand sign signal. The algorithm is built for translating the numerical hand sign signal and the result are displayed on the seven segment display. Altera's Quartus II, SOPC Builder and Nios II EDS software are used to construct the system. By using SOPC Builder, the related components on the DE2 board can be interconnected easily and orderly compared to traditional method that requires lengthy source code and time consuming. Quartus II is used to compile and download the design to the DE2 board. Then, under Nios II EDS, C programming language is used to code the hand sign translation algorithm. Being able to recognize the hand sign signal from images can helps human in controlling a robot and other applications which require only a simple set of instructions provided a CMOS sensor is included in the system.

Keywords: Sign Signal Translation System, FPGA, Altera DE2 board, Mathematical Morphology.

1. INTRODUCTION

Hand gesture recognition studies especially hand gesture recognition for Human-Computer Interaction (HCI) and robot control have drawn a lot of attentions from researchers around the world in recent decades. Hand sign signal interpretation is also important as it is one of the mediums for hearing impaired community because verbal communication is almost impossible to exchange their ideas and to communicate with one another. However, until today there are still many issues faced by the researchers in the effort to recognize the hand gesture. Among them are the variation of the hand gesture appearance and the image processing speed as it involves many mathematical calculations.

The aim of this paper is to develop a sign signal translation system that is able to detect and translate the hand gesture (sign signal) from individual captured images. The detected sign signal is then translated into its corresponding numerical representation. Altera's DE2 board that features Cyclone II FPGA chip with Nios II soft core processor inside is used to construct the system. FPGA is used to construct the system as FPGA is considered to have the capability to

perform the image processing faster as discussed in [1]. Other advantages of using FPGA to build an embedded system also reviewed in [2]. Altera Quartus II, SOPC Builder and Nios II EDS software are utilized to build the system as these software provide user-friendly development environment that enables the hardware and software designer to create a system from scratch to a complete system within a short period of time and thus, speeding the time-to-market required as compared with the traditional design method.

2. LITERATURE REVIEW

In recent years, human hand gesture recognition that provides a natural way to interact and communicate with machines has grabs much attention of many researchers around the globe. Various algorithms and techniques for recognizing hand gesture had been introduced by the researchers. Traditionally, there are two main approaches introduced to recognize gesture which are glove-based approach and vision-based approach. Glove-based approach [3] usually requires the signer to put on gloves, sensors and so on that are used as measuring devices to model the hand postures whereas vision-based approach [4] uses video camera to capture images and apply a particular image processing algorithm to recognize hand posture.

Conventionally, for hand gesture recognition, the system should be consisting of four stages which are image acquisition, hand features extraction, processing extracted features and hand gesture recognition [5] [6] [7]. The block diagram shown in Figure 1 depicts the hand gesture recognition steps that are commonly applied by the researchers. In the first stage, sign signal, which is hand gesture, is captured by a CMOS sensor. Then, this is followed by the stage of hand image features extraction. Generally, sign signal is detected by segmenting out the hand region from image background by applying some appropriate algorithms for hand gesture detection. At this stage, the system will perform some computation to segment the hand shape from the image, which will later be used in matching algorithm for the sign recognition process. Some researchers paste colored markers on the hand and estimate the hand posture by analyzing the markers' positions from 2D images. This can improve the feature extraction process. The last stage is the sign recognition stage where hand gesture recognition algorithm is developed to perform the matching between the input sign signal with the trained images in the database.

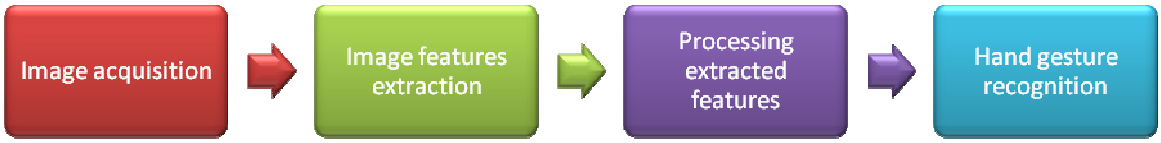


FIGURE 1: Conventional hand recognition system.

Paulraj et al. [5] presented a simple sign language recognition system that is capable of recognizing nine phonemes in English using a machine vision system. The system had been developed based on skin color segmentation and Artificial Neural Network (ANN). There are three processing stages in the system; preprocessing, feature extraction and gesture classification. Skin color detection and region segmentation are carried out during the preprocessing stage. Skin color of the hand is detected based on the RGB values in the image frame. The feature extraction stage extracts moment invariant features, obtained by calculating the blob in the set of image frames, from the right and left hand gesture images. The gesture classification stage then uses these features as its input to ANN to recognize the sign. It is reported that the average recognition rate for this system is 92.85%.

Asanterabi et al. [8] proposed a fast algorithm for vision-based hand gesture recognition for a robot control application. The system managed to recognize automatically a limited set of gestures from hand images in real time. Their approach involves segmenting the hand region, locating the fingers and classifying the gesture. The proposed system is invariant to translation, rotation and scale of the hand. Besides, the technique used does not require the storage of a hand gesture database. The system used Red:Green (R:G) ratio to decide whether a pixel is

belonged to the hand region or not. If a pixel with color intensities is within the thresholds, it is set to 1, else it is set to 0, which resulting the binary, black and white image output. Region-based segmentation is used to eliminate the erroneous decisions produced during the skin detection in the preceding step where an assumption is made that the largest connected white region area is corresponding to the hand. Therefore, a relative region size threshold is used to eliminate the unwanted regions. The threshold value is chosen as 20% from the total number of pixels in the white region. The outcome is the segmented hand region. Centre of gravity (COG) of the segmented hand region is then computed. Next, a circle with radius 0.7 of the farthest distance from the COG, which is the distance from the centroid to the tip of the longest active finger in a particular gesture, is drawn. The circle will intersect with all the active fingers in a particular gesture. By tracing the circle constructed, the number of transitions from 0-to-1 (black to white), are counted. Subtracting the total counted number by one (for the wrist), this gives the estimation of active fingers in the gesture segmented and leads to the gesture recognition. Since the proposed system is just counting the number of active fingers by drawing a circle and counting the intersection regardless which fingers are active, this system is robust to translation, rotation and scale of the hand.

Park et al. [9] implemented a real-time embedded FPGA-based gesture recognition system using 5DT data glove. This approach is used in order to reduce the problems of space limitations, movement limitations and lighting limitations. The architecture of the system consists of three main modules that are input module, recognition module and display module. The system recognizes the hand gesture by performing data calculations with a checksum function on the input data and compares the result to the header byte before proceeds to the matching process. The matching process compares the input hand gesture with the pre-defined hand gesture. Then, the result is displayed on the LCD screen.

Yu et al. [7] introduced an FPGA-based smart camera which is called GestureCam. This smart camera is capable to perform simple vision-based hand gesture recognition. GestureCam comprises an image capture unit (ICU), FPGA-based gesture recognition unit (GRU), and a host and display unit (HDU). A low-pass filter is applied in order to obtain a clearer skin image from the extracted skin color image. GestureCam applies contour tracing algorithm, which is based on inner boundary tracing algorithm (IBTA) with several medications, to extract hand contour for gesture classification. GestureCam classification module is using ANN and trajectory-based methods.

Wang et al. in [10] presented a sign language recognition system that uses tensor subspace analysis to model a multi-view hand gesture. The hand recognition process is achieved through color segmentation. Input image that is in RGB color space is converted to YCbCr color space to ease the process of detecting the skin that employs the Back Propagation (BP) networks model. The sign language recognition is modeled and recognized using tensor. Then, the matching process is carried out to identify the input hand gesture.

Paulraj et al. [4] proposed a system for translating the hand gesture for “Kod Tangan Bahasa Melayu” (KTBM) into voice signal. USB web camera producing RGB video stream format, with a screen bit depth of 24 bits and resolution of 320×240 pixels, is used to record the input hand gesture. The captured frames are processed in grayscale mode and an average filter is used to remove the unwanted noises present in the images. The system is capable of detecting the head region, left hand region and right hand region by applying a simple segmentation technique. Segmented images are then converted into binary images. Subsequent frames are modeled using a simple feature extraction technique based on discrete cosine transform (DCT). A simple neural network model is developed in order to classify the hand gesture.

Vanderlei et al. [11] developed a real time gesture recognition system based on FPGA for mobile robots. Input gesture images are captured using an ov7620 CMOS image sensor that is able to sustain an image rate of 30 frames per second (fps). The captured RGB (Red, Green, Blue) color images are converted to their corresponding HSI (Hue, Saturation, and Intensity) representations

before further transformed into binary symbols. However, only the H component is used since it gives good hand region detection. Then, the image is filtered and down-sampled to 32×24 pixels. The down-sample process is required to reduce the number of inputs for the ANN. To increase the neural network efficiency rate, the image is centralized by performing some computation. The hand gesture recognition is accomplished by applying RAM-based neural network, which is implemented in the FPGA.

Wing et al. [6] presented a real-time hand gesture recognition system based on Haar wavelet transform. They proposed a codeword scheme based on features of hand gestures for matching process. Besides, the system reduces the database size by standardizing the orientation of hands using the principal axis. In the project, recognition algorithm based on Haar wavelet representation has been developed. Hand images with resolution of 160×120 pixels are captured by using ICE digital webcam. Skin color approach is used to extract the hands from the image.

Mohandes et al. [3] proposed an image based system for recognizing Arabic sign language. First, the system detects the signer's face by using the Gaussian skin model. Centroid of the detected face region is taken as the origin for each frame and then the hands movement is tracked by applying region growing technique. Hidden Markov Model (HMM) is used to perform the classification of the signs during the recognition stage, through some computation based on the features extracted from the input images.

3. SYSTEM FUNCTIONALITY DESCRIPTIONS

The sign signal translation system created is capable to recognize a set of hand gestures, which is the sign signal, present in an image that captured using a camera. The sign signals are recognized based on the finger counts detected in the image. Our approach is based on the approach presented in [8]. Figure 2 depicts the general process flow of the system to detect and recognize the hand gesture. Each stage involves different image processing techniques and algorithms which will be explained in details in the subsequent subsections.

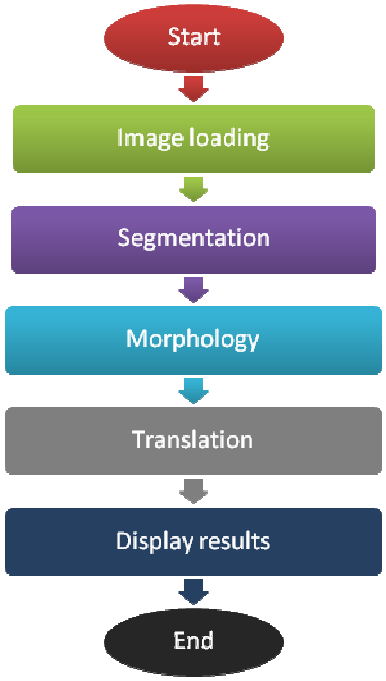


FIGURE 2: The process flow of the sign signal translation system.

3.1. Image Loading

A set of pre-captured images are used to test the functionality of the system. The captured images are stored as 24-bit-depth color images. In order to process the images, it is necessary to convert every pixel of the images into numerical raw data for processing. In this project, the raw data of the images are saved as 8-bit decimal numbers. A custom C program is created to split a single 24-bit RGB image into three single 8-bit images, corresponding to R, G and B channels. Each channel is stored in a separate raw data .txt file. This means that from one image, three .txt files will be generated. The pixel of the raw data of the hand image is arranged as 1D array. Then, the generated R, G and B raw data files are included into the Nios II EDS, a C/C++ programming workspace, as header files for further processing.

All the images are resized to 320×240 pixels, which is 76,800 pixels in total. This indicates that there are 76,800 elements in each 1D array, and each array element requires an 8-bit storage. In the source code, the width and the height of the image are supplied to calculate the corresponding coordinates (x,y) for each 1D array element. These coordinates is extremely important especially during the dilation and erosion operations using a 3×3 pixels structuring element. Since the image size is defined as 320×240, thus, address size required to store the image is 320×240×4 that is equal to 307,200 (0x0004B000). The total number of pixels is multiplied by 4 because the memory address variable in the source code is defined as 32-bit integer (int) that is equivalent to 4 bytes.

It is worth noting that there are 18 switches in total available on the DE2 board. These switches are used to load the corresponding image to the system for the recognition process. To achieve this, each switch is assigned to a specific input image.

3.2. Image Segmentation

First, the image is converted into 8-bit grayscale from 24-bit RGB by using equation (1).

$$Grayscale = (R+G+B)/3 \quad (1)$$

Image segmentation is carried out by using simple thresholding technique. Threshold for the grayscale value is set at 50 to segment the hand region. The thresholding value is decided based on trial-and-error experiment conducted on several images. For pixels with grayscale value less than 50, they will be set as `0´, which is black. On the other hand, for pixels with value greater than or equal to 50, the pixels value is set as `255´, which represents the white color. The IOWR instruction as defined in library file <io.h> is used to write the binary value into SDRAM starting at the base address 0x00800000. This address actually has been declared automatically in the library file <system.h> by the SOPC Builder.

3.3. Binary Image Morphology

The purposes of carrying out the binary image morphology are to repair and smooth the segmented image, for better fingers detection in the next stage. The morphology operation chosen to be applied is the closing operation, using a structuring element of 3×3 pixels. Closing operation is defined as dilation, followed by erosion, both using 3×3 pixels structuring element. Erosion process can remove salt noise (i.e. noise presented as white pixels), and make the fingers' area thinner. Dilation process can repair breaks and intrusions. Combination of both morphological processes (i.e. closing operation) produces a smooth contour of the hand region and fills small gaps and holes that might appear in the binary images. During the dilation process, processed pixels will be stored in the new address in the SDRAM starting at address 0x00900000 as defined in the source code. During the erosion process, processed pixels will be stored at the address starting at 0x00800000. This means that the pixels stored after the gray-scaling operation will be overwritten. This approach is used to optimize the SDRAM usage.

3.4. Sign Signal Translation

In this stage, the height and width of white region which represents the hand region and centroid, $(\bar{x}, \bar{y})$ of the region is determined. The centroid is calculated by using equation (2).

$$\bar{x} = \left(\sum_{i=0}^{k} x_i \right) / k \quad \text{and} \quad \bar{y} = \left(\sum_{i=0}^{k} y_i \right) / k \tag{2}$$

where x_i and y_i are coordinates of the i-th pixel in the hand region, and k presents the total number of pixels defined by this region. The farthest distance of white pixel from the centroid is also been determined.

Using the information regarding to the centroid and the hand area, a circle with radius 70% from the farthest distance value is constructed. This circle is used for detecting the active fingers. The radius 70% from the farthest distance is selected based on the experiment conducted on several images that shows that this distance is most likely to intersect with all active fingers. The same value also purposed in [8]. The number of fingers intersect the circle will be the count that represented by the image captured. Counting is done by calculating the number of black to white (`0´to `255´) transitions by tracing the circle constructed in the previous step. The number of the transitions accumulated will be deducted by one due to the wrist. Number after the subtraction will be the number representing the sign detected in the image.

4. DESIGN ARCHITECTURE

In this project, default IP cores provided in SOPC Builder and Altera University Program IP cores are used to implement the system. The version of the Quartus II used in this work is 10.1 SP1. The version of the NIOS II EDS used is 10.1 SP1 as well. Figure 3 depicts the block diagram of the system created.

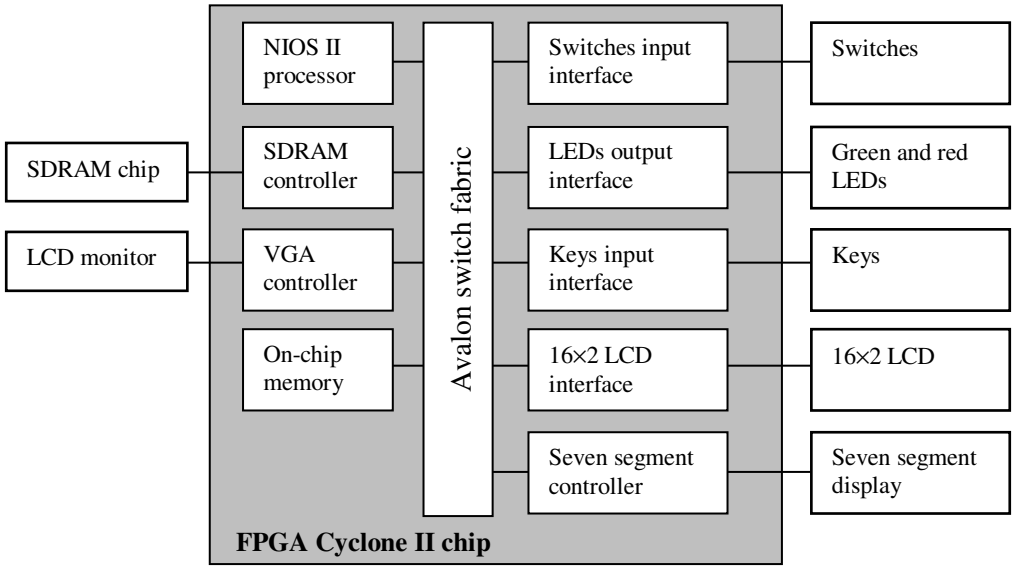


FIGURE 3: The system block diagram implemented on DE2 FPGA board.

4.1. SOPC Builder

SOPC Builder is used to automatically generate and interconnect the desired components on the DE2 board. Several IP cores are used to build the system. Nios II processor used in this system is Nios/f type. In addition, IP cores from University Program are used and the setting has been configured to suit the needs and functionality of the system. SDRAM is used to store the image for processing since it provided a large capacity for storage, which is 8M bytes. On-chip memory is assigned as the reset vector and exception vector for the system. SRAM is used as a pixel buffer for video out. There are several PIO included as well namely seven segment displays, LEDs, switches and the buttons. Figure 4 shows the summary of the IP cores used to build the sign signal translation system. From the figure, the base address of each module can be seen as well as the IRQ. The smaller the IRQ number, the higher the priority of the interrupt request.

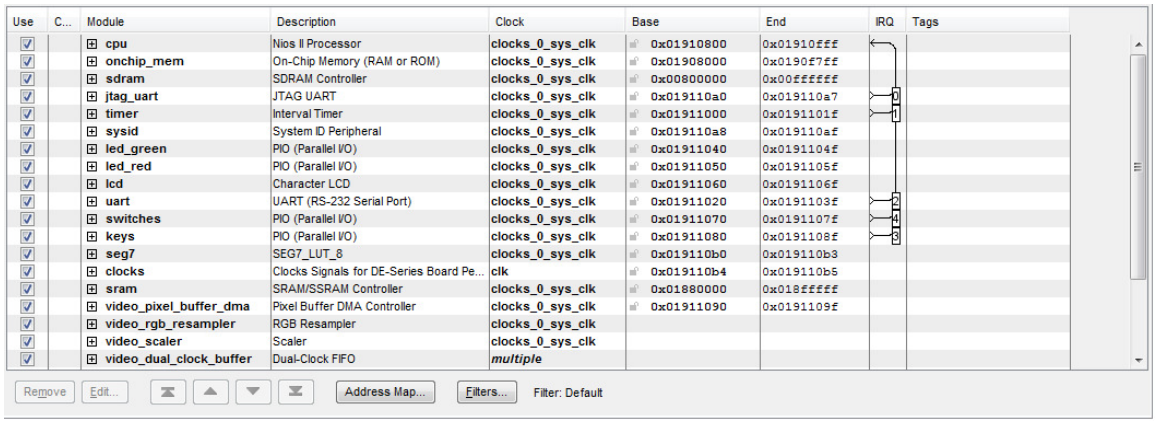


FIGURE 4: Summary of the component used in SOPC.

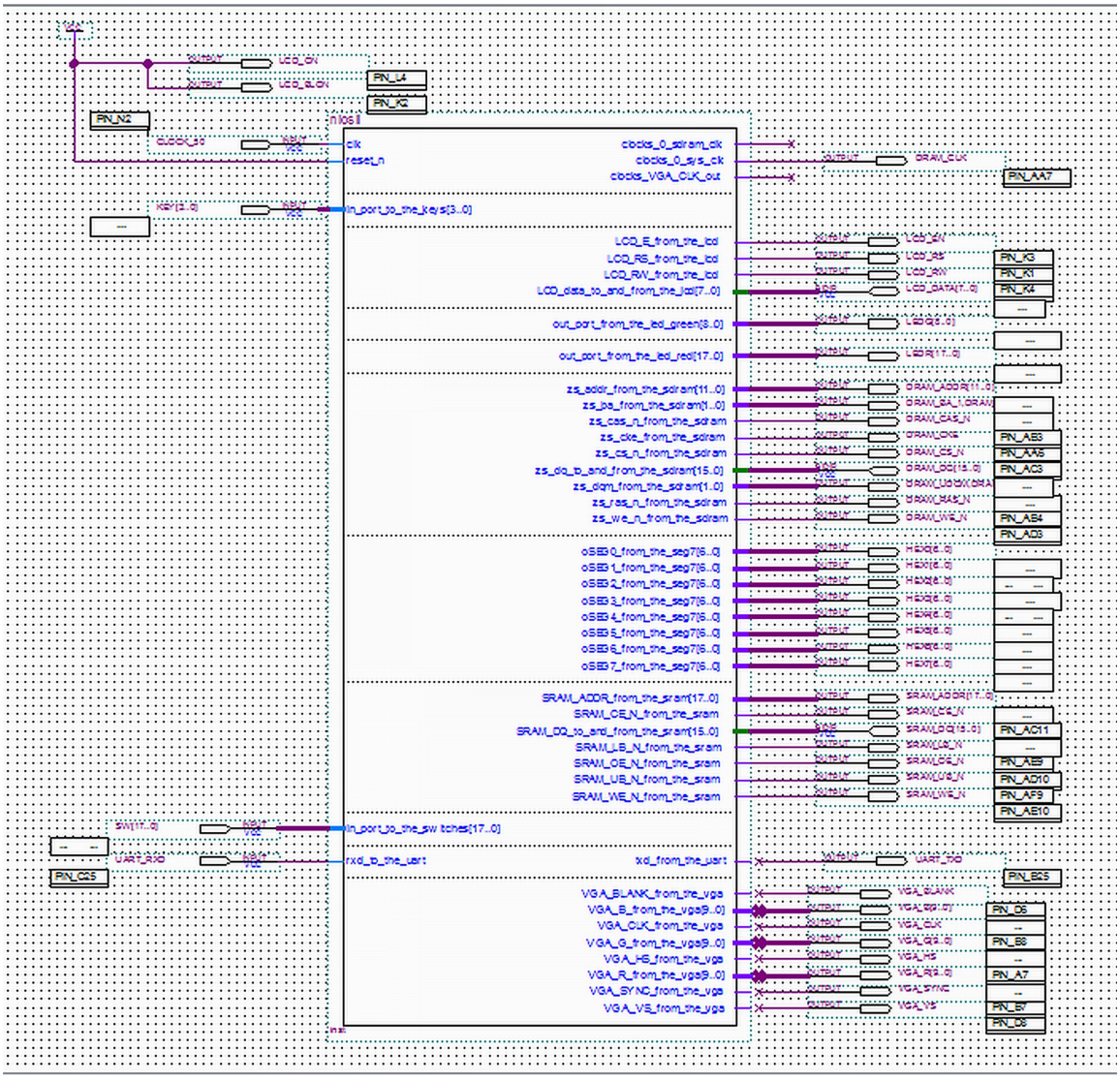


FIGURE 5: The top-level schematic for the system on FPGA DE2 board.

Clock IP core provided by the Altera University Program is used to generate three different clock signals for SDRAM controller IP core, VGA controller IP core and the system itself to make sure that the system is working properly. SDRAM controller core requires input clock signal with -3.00 ns whereas VGA controller core requires only 25MHz for the VGA to display the 640×480 image on the LCD monitor screen correctly as stated in the user manual.

In the SOPC Builder, Pixel Buffer DMA Controller IP core is used to retrieve the image pixel from the SRAM by using its Avalon memory-mapped master interface. Retrieved pixel then is sent to RGB Resampler IP core via its Avalon streaming interface. Pixel retrieved from SRAM is 16-bit, thus, RGB Resampler IP core is used to change pixel format received from Pixel Buffer DMA Controller IP core from 16-bit to 30-bit RGB. Next, in order to display the image in 640×480, Scaler IP core is included into the SOPC. The purpose of including the Dual-Clock FIFO IP core is to buffer the pixel received from Scaler IP core at the input clock frequency of 50MHz and then send out the pixel to the VGA Controller IP core at output clock frequency of 25MHz. This is to ensure the VGA Controller IP core send the pixel to the LCD monitor at the appropriate frequency so that the image is displayed correctly.

4.2. Quartus II 10.1 SP1

There are several methods to instantiate the design. In this project the components used in the SOPC are instantiated in Quartus II by using the schematic block diagram as shown in Figure 5. Location of each input and output pin of the schematic block diagram on the DE2 board is assigned accordingly and carefully as wrong pin assignment might damage the board after programming. Pin planner is used to complete the task. There are two pins without pin assignment which are the clocks_0_sdram_clk and clocks_VGA_CLK_out as can be observed in the figure. For both pins the connection is done internally by the SOPC Builder. After done assigning the pins, the project is compiled. The design is downloaded to the DE2 board using the programmer after the project is successfully compiled.

4.3. Nios II EDS 10.1 SP1

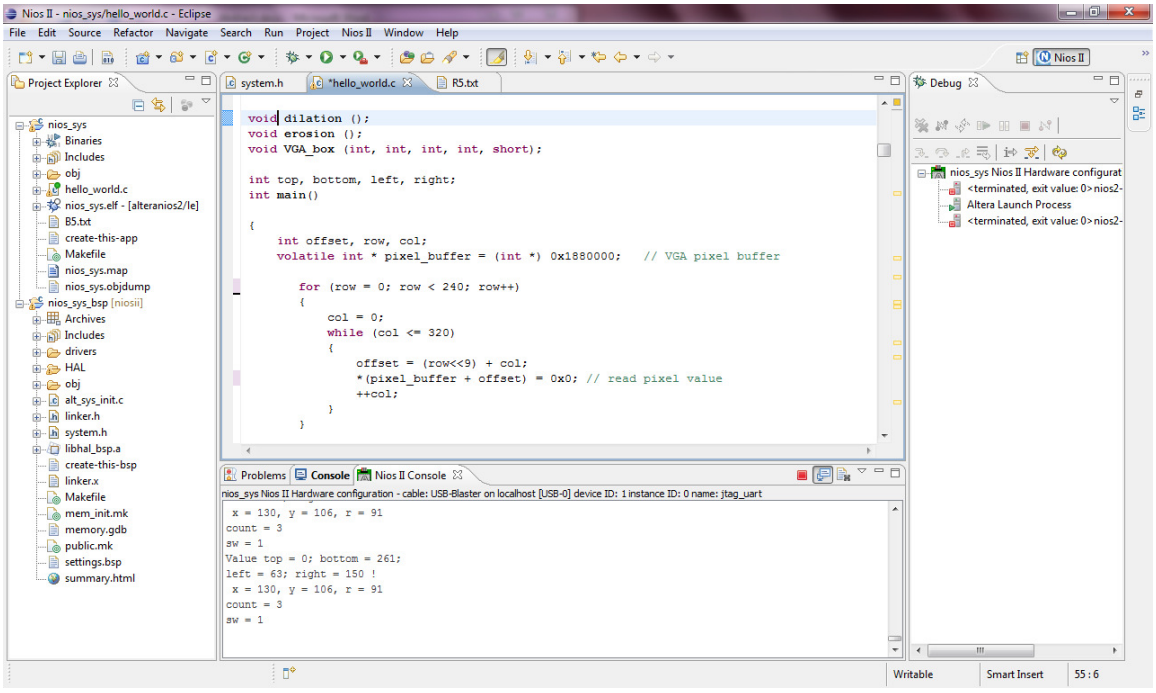


FIGURE 6: Nios II EDS programming environment.

In the Nios II EDS, the C programming language is used to program the system. To code the image processing algorithms, it is important to go through the header files especially the <system.h> header file which automatically generated after the system library is built. In this header file, base address for each module created in SOPC Builder is defined so that designer can access the module just by including the base address or the symbolic name in the source code. Another important header file is <altera_avalon_pio_regs.h>. It contains the important functions to write values to the PIO registers for controlling the PIO devices' activities on the DE2 board. By knowing all the base address of the modules included in the SOPC Builder and the functions in the header, the algorithm can be created to access the memory to read and write the pixel value and control the peripherals on the DE2 board. Figure 6 shows a portion of the C code written in the Nios II EDS. It can be seen that, there are some message displayed in the Nios II console that is to verify the result and check the correctness of the system.

5. RESULTS AND DISCUSSIONS

Figure 7 shows the hardware setup of this project. LCD monitor is connected to DE2 board via VGA connector whereas laptop is connected to DE2 board via JTAG-UART to establish the communication. LCD monitor is used to display the result after each image processing operation is completed. The purpose of this LCD monitor is to evaluate each image processing operation and the algorithm built. The laptop with Altera's Quartus II, SOPC Builder and Nios II EDS installed is where the system is built. After the algorithm is completely coded, it is programmed to the DE2 board through the Quartus II software. Then, the system is ready for evaluation.

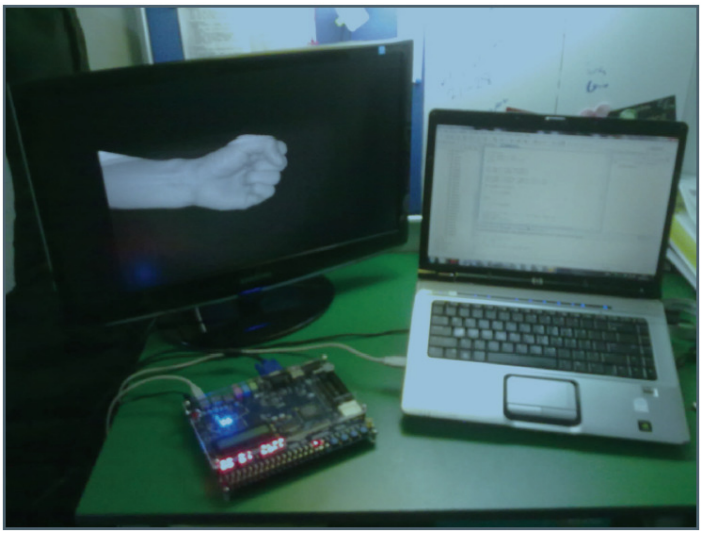
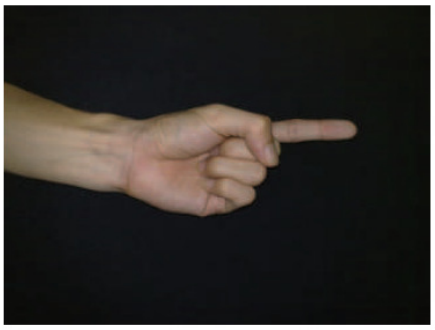


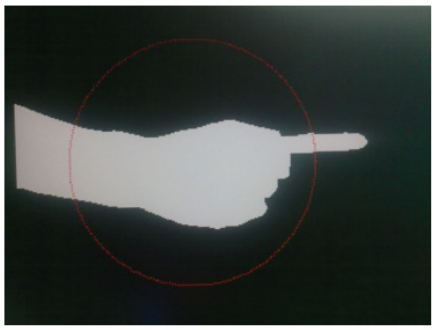
FIGURE 7: Sign signal translation system setup.

Before conducting the experiment, there are a few assumptions made when feeding the hand images to the system. First of all, the system assumes all the hand images are captured under a controlled environment, where the images of hand are taken against a uniform black background. The condition is required to obtain a better contrast and noise free image, which will ease the segmentation process during the image pre-processing stage. Secondly, the system assumes that the wrist of the hand is exposed in the images since the algorithm will counts the total number of black-to-white transitions and then the number will be subtracted by 1 which is the wrist. Next, the system assumes that the images are all taken within a particular distance, preferable around 30cm away from the camera. Figure 8 to Figure 12 show some of the results obtained when these assumptions are fulfilled. As shown by these figures, the system correctly identify the numerical hand sign signals for numbers '1', '2', '3', '4', and '5'.

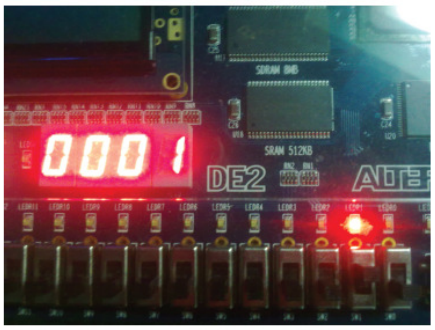


(a) Original image.

(b) Binary image generated after thresholding operation.

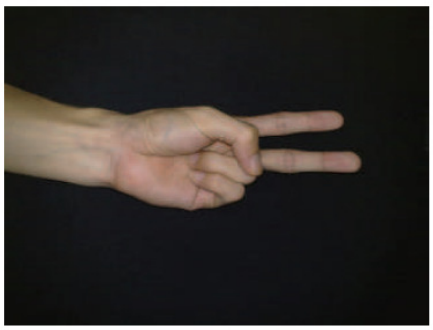


(c) Cirle constructed after morphology operations.



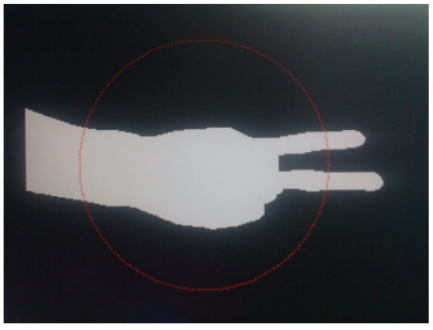
(d) Translation result.

FIGURE 8: Sign signal translation for number '1'.

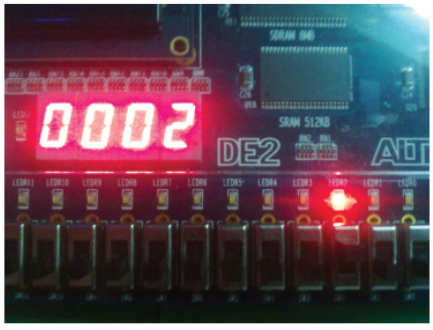


(a) Original image.

(b) Binary image generated after thresholding operation.

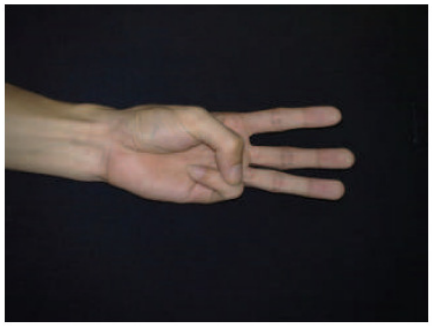


(c) Circle constructed after morphology operations.



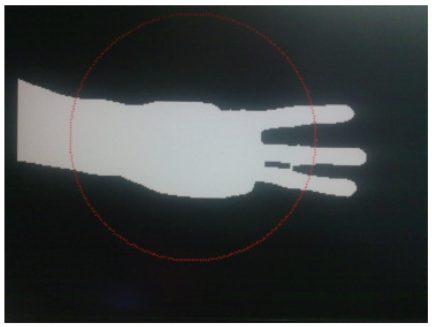
(d) Translation result.

FIGURE 9: Sign signal translation for number '2'.

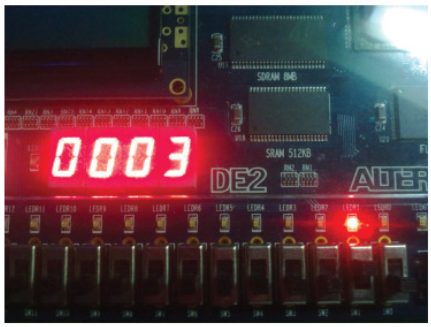


(a) Original image.

(b) Binary image generated after thresholding operation.



(c) Circle constructed after morphology operations.



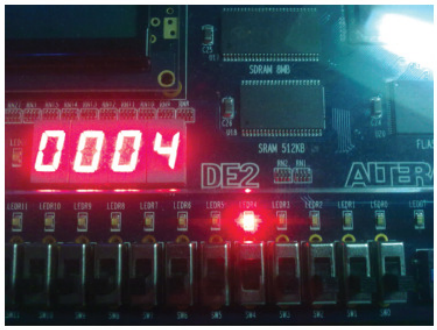
(d) Translation result.

FIGURE 10: Sign signal translation for number '3'.

(a) Original image.

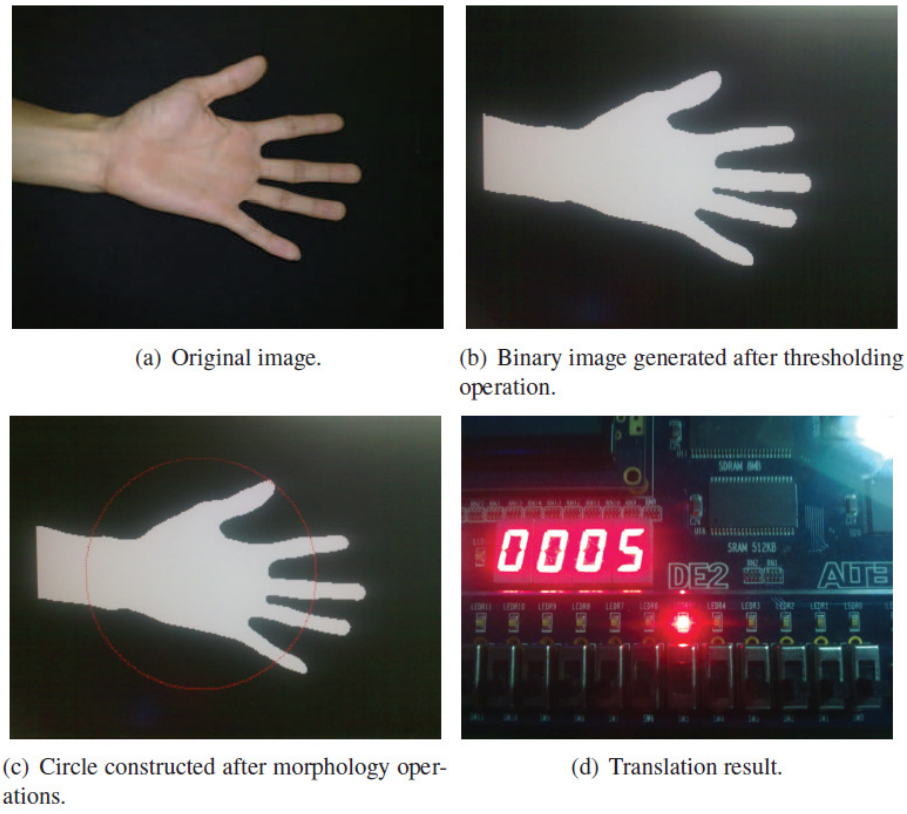
(b) Binary image generated after thresholding operation.

(c) Circle constructed after morphology operations.



(d) Translation result.

FIGURE 11: Sign signal translation for number '4'.



(a) Original image.

(b) Binary image generated after thresholding operation.

(c) Circle constructed after morphology operations.

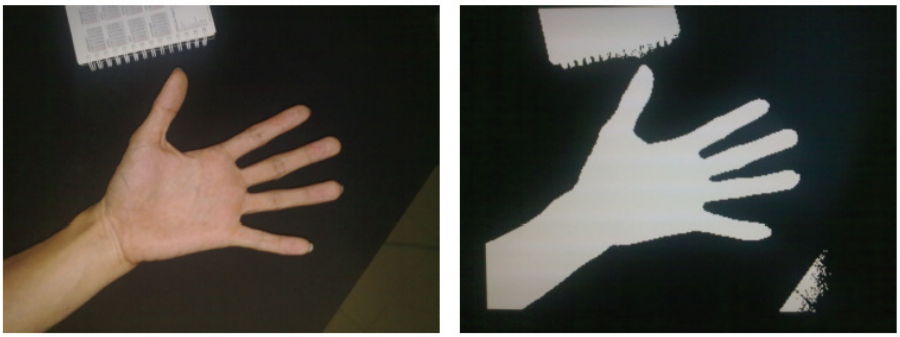
(d) Translation result.

FIGURE 12: Sign signal translation for number '5'.

Two additional experiments have been carried out to investigate the performance of the system if the abovementioned assumptions are being ignored. The first experiment shows the importance of having a uniform black background during image acquisition. Figure 13 show the original input image captured against background that is not completely black as images before. By observing Figure 13(b), it can be clearly seen that there are unwanted white regions present in the binary image generated after the thresholding process. In Figure 13(c), the circle for detecting the active fingers is failed to intersect with the active fingers. This is due to the presence of the unwanted white regions that affected the calculation of the COG.

Generally, the system's failures in translating hand gesture were due to the noise pixels present in the binary image. The noise will cause the calculation of the centroid of the hand region in binary image goes wrong and leads to the failure. Moreover, the system does not apply any filter in the algorithm to reduce the noise. Therefore, it is important to avoid too many unwanted pixels which mostly are the pixels that have the skin-like color. Besides, the experiment also shows that the dilation and erosion operations applied in this algorithm are only capable to remove pepper noise that is smaller than 3×3 pixels. Thus, the unwanted white big regions as shown in Figure 13(b) cannot be removed by these morphology operations.

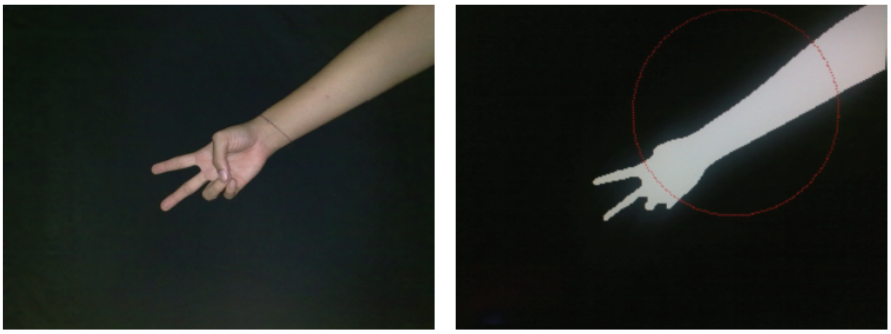
The second experiment shows the significance of the distance between hand and camera. Figure 14 is the hand gesture image that is captured by using a camera from 60cm away. It is very obvious that the circle constructed failed to intersect the active fingers as shown in Figure 14(b). This is due to the radius of the circle constructed was not big enough as the image was captured from 60cm away. The distance exceeds the recommended distance which is around 30cm. This shows that the distance of the image is captured and the wrist and arm exposed are the factors that should be taking into consideration for the system to recognize the hand gesture correctly. From several tests, it was found that the preferable distance should be around 30cm in order for the system to translate the hand gesture correctly.



(a) Original image captured. (b) Binary image after thresholding operation.

(c) Circle constructed.

FIGURE 13: The effect of bright structure on the background towards sign signal translation.



(a) Original image captured. (b) Binary image and circle constructed.

FIGURE 14: The distance limitation test.

6. CONCLUSION

The experiments conducted justify the functionality of the system. The system is able to detect the hand presents in the image and successfully translate the sign signal. On the other hand, the result also proved that the DE2 board and the PIO peripherals such as switches, seven segment display and LCD monitor are working properly as expected without showing any signs of errors. Furthermore, the experiment results tell that the algorithm built for the sign signal translation purpose is working as well. Nevertheless, there are still limitations on the system such as other complicated hand sign signal such as those used by the hearing and speech impaired community cannot be identified. Presence of noises, the image capturing controlled distance and the wrist exposed on the captured image are among the factors that affect the performance of the system.

However, by successfully interpreting the numerical number posed by the signer, several instructions can be applied into each hand sign signal provided the system manage to run in real time and CMOS sensor is attached to the system. For example, the sign signal can be used to navigate the music played, to instruct a robot to move right, left, stop and so on.

ACKNOWLEDGMENT:

The authors would like to thanks Mr Ahmad Nazri Ali for his useful suggestions. This work was supported in part by the Universiti Sains Malaysia's Short Term Research Grant with account number 304/PELECT/60311013 and by Incentive Grant (Postgraduate Students) with project number 1001/PELECT/8022006.

REFERENCES:

[1] M. Shirvaikar, and T. Bushnaq, "A comparison between DSP and FPGA platforms for real-time imaging applications", in *Real-Time Image and Video Processing 2009,* San Jose, CA, USA, 2009, pp. 724406.

[2] S. Asona, T. Maruyama, and Y. Yamaguchi. "Performance comparison of FPGA, GPU and CPU in image processing", in *Field Programmable Logic and Applications (FPL 2009),* 2009, pp. 126-131.

[3] M. Mohandes, S. I. Quadri, and M. D. King. "Arabic sign language recognition an image based approack", in *21st International Conference on Advanced Information Networking and Applications Workshops, 2007 (AINAW'07),* 2007, pp. 272-276.

[4] M. P. Paulraj, S. Yaacob, H. Desa, and W. Majid. "Gesture recognition system for Kod Tangan Bahasa Melayu (KTBM) using neural network", in *5th International Colloquium on Signal Processing and Its Applications (CSPA 2009),* 2009, pp. 19-22.

[5] M. P. Paulraj, S. Yaacob, M. S. Zanar Azalan, and R. Palaniappan. "A phoneme based sign language recognition system using skin color segmentation", in *6th International Colloquium on Signal Processing and Its Applications (CSPA 2010),* 2010, pp. 1-5.

[6] W. K. Chung, W. Xinyu, and Y. Xu. "A realtime hand gesture recognition based on Haar wavelet representation", in *Proceedings of the 2008 IEEE International Conference on Robotics and Biomimetics,* Washington, DC, USA, 2008, pp. 336-341.

[7] Y. Shi, and T. Tsui. "An FPGAbased smart camera for gesture recognition in HCI applications", in *Computer Vision-ACCV 2007,* 2007, pp. 718-727.

[8] A. Malima, E. Ozgur, and M. Cetin. "A fast algorithm for vision-based hand gesture recognition for robot control", in *14th IEEE Signal Processing and Communications Applications,* 2006, pp. 1-4.

[9] I. K. Park, J. H. Kim, and K. S. Hong. "An implementation of an FPGA-based embedded gesture recognizer using a data glove", in *Proceedings of the 2nd International Conference on Ubiquitous Information Management and Communication (ICUIMC'08),* 2008.

[10] S. J. Wang, D. C. Zhang, C. C. Jia, N. Zhang, C. G. Zhou, and L. B. Zhang. "A sign language recognition based on tensor", in *Second International Conference on Multimedia and Information Technology (MMIT)*, 2010, pp. 192-195.

[11] V. Bonato, A. K. Sanches, M. M. Fernandes, J. M. P. Cardoso, E. D. V. Simoes, and E. Marques. "A real time gesture recognition system for mobile robots", in *ICINCO 2004,* 2004, pp. 207-214.