IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received February 20, 2017, accepted March 23, 2017, date of publication April 28, 2017, date of current version June 28, 2017.

Digital Object Identifier 10.1109/ACCESS.2017.2699665

The Design and Implementation of an Automatic Tonsillitis Monitoring and Detection System

PRANITHAN PHENSADSAENG AND KOSIN CHAMNONGTHAI, (Senior Member, IEEE)
Department of Electronics and Telecommunication Engineering, King Mongkut's University of Technology Thonburi, Bangkok 10140, Thailand
Corresponding author: Kosin Chamnongthai (kosin.cha@kmutt.ac.th)

ABSTRACT An automatic tonsillitis monitoring and detection system aims for personal use and requires mobility, a compact size, and a light weight with reliable functionality. This paper proposes the design and implementation of automatic tonsillitis monitoring and detection system. In this system, a pipeline concept is utilized for data transmission among processes, and parallel processing is employed for feature extraction by dividing an image into an appropriate number of blocks for processing. The performance of the proposed method was evaluated by experiments using prototype software and hardware systems of our design, and the results show accuracies of 96.4% and 91.8%, respectively, for 159 tonsillitis samples.

INDEX TERMS Bio-medical imaging, FPGA implementation, fuzzy system, tonsillitis detection, VHDL.

I. INTRODUCTION

Tonsillitis is a viral disease that can lead to ischemic heart disease, chronic obstructive pulmonary disease (COPD), and kidney disease, which rank as the top 1, 5, and 10 causes of death, respectively [1]. In addition, tonsillitis can lead to pneumonia, for which it is estimated that 58% of children are taken to an appropriate health provider [2]. Tonsillitis is sometimes caused by bacteria in addition to viruses entering the body. Ordinarily, bacteria and viruses enter the body via the mouth and nose and are filtered in the tonsils, where white blood cells from the immune system destroy the viruses and bacteria by producing inflammatory cytokines that lead to fever. To prevent and cure tonsillitis, many people, especially children and the elderly, need tonsil monitoring and tonsillitis detection to be periodically performed by expert medical doctors.

Due to the shortage of medical doctors, especially in the countryside and within an aging society, it is inconvenient to periodically consult doctors; therefore, an automated tonsillitis monitoring and detection system is needed. To be practical, the automated system should be compact and portable with low power consumption.

In related research, Horio *et al.* [3] proposed the study of a Mucous Membrane Diagnosis Support System using probabilistic relaxation. Using intraoral images, they studied the shapes of patches of vitiligo and judged them to be lacy or non-lacy with high accuracy. They extracted the vitiligo areas using a dynamic threshold and found they were adequately extracted even when the lighting condition and angles changed in the diagnosis support system for mucous membrane disease in the oral cavity. In a study of computer-aided diagnosis of tonsillitis using size and color [4], Phensadsaeng *et al.* employed redness of color and tonsil size to detect abnormalities by considering changing trends of those features daily and then applying fuzzy logic as a decision method. Images of tonsillitis processed with 84% accuracy for tonsillitis detection as a result. In a previous study of feature extraction using gradient and watershed methods for tonsillitis diagnosis [5], three approaches were proposed as follows: finding the gradient image value, segmenting the image regions and their boundaries using the watershed technique, and merging the two procedures to calculate the number of pixels in each region. As a result, the study yields better performance and works well for 2D images. It achieved an accuracy of at least 90% compared with that of a doctor's diagnosis. [5].

As mentioned earlier, our research project is focused on implementing an automatic tonsillitis monitoring and detection system that is compact in size, portable and has low energy consumption. To meet these goals, the system requires a hardware solution. In designing and implementing a hardware system for monitoring and detection, Jin *et al.* [6] proposed a pipelined data path for high-speed face detection using FPGAs. This proves that the architecture of a pipeline and pyramid scheme can work efficiently for detection in terms of hardware.

Many object detection methods employ the concept of a pipeline when implementing a hardware system for feature

extraction using the Adaboost algorithm and Haar-like features [7]–[13] to compute efficiently and to be robust against errors.

Adaboost and Haar-like features are suitable for the repeat data that contains many positive and negative training images. With tonsillitis, the tonsillitis glands have various shapes and are located in oral cavity areas with similar color intensity. This leads to a difficult training stage; therefore, this study will only use a weighted threshold of the area of interest [14], a scan interval between sub-windows [15] and a parallel architecture [16] for Adaboost and Haar-Like features. These methods are applied to images whose backgrounds are fixed.

However, in the context of tonsillitis detection, the backgrounds vary among patients, so these methods are inappropriate.

To solve this problem, we designed a VLSI architecture for the detection system using a line buffer technique to filter the tonsillitis size in a preprocessing step and developed a hardware component for feature extraction by comparing screened tonsillitis candidates with a database.

However, the design architecture was based on data series processing, which required processing time.

Authors hence designed another architecture [17] for differentiating the two stages of healthy tonsils and tonsillitis using pipelined and parallel processing to improve the processing time. In this architecture, an image of tonsil was divided into n blocks, and a window was used to detect the tonsillitis border in each block. Using this hardware, pipeline processing was performed within the windows, all the blocks were processed in parallel, and decisions were performed using comparators.

This architecture was initially intended for detecting only two stages: healthy tonsils and tonsillitis. However, early stage tonsillitis detection is crucial for preventive treatment. This paper therefore targets an architecture design for detecting three stages including healthy tonsils, tonsillitis, and its early stage. In the hardware design, hardware for fuzzy logic was implemented for decision making, and circuits were added for consideration of both global and local thresholding in order to precisely determine the tonsillitis boundary.

This paper is organized as follows. Analysis of the problem and basic concepts are explained in chapter 2. The system, including both software-based and hardware-based tonsillitis monitoring, are described in chapters 3 and 4, respectively. The experiments are mentioned in chapter 5, and a discussion and conclusion are provided in chapters 6 and 7, respectively.

II. PROBLEM ANALYSIS AND BASIC CONCEPT

To develop an automated system for tonsillitis detection and classification in three stages, healthy tonsils, tonsillitis, and early stage tonsillitis, healthy tonsils and tonsillitis are anatomically analyzed in order to study different characters of those three stages.

From previous reports [4], we learned about changes in tonsil size, color, and intensity on its boundary and created ways to employ this information in tonsillitis detection. To implement an algorithm for hardware, we looked for redundancy in tonsillitis boundary segmentation, divided an image into n blocks, and used a window for scanning each pixel using pipeline processing within the window while processing all blocks in parallel. The detail of analysis and our basic concept is described as follows.

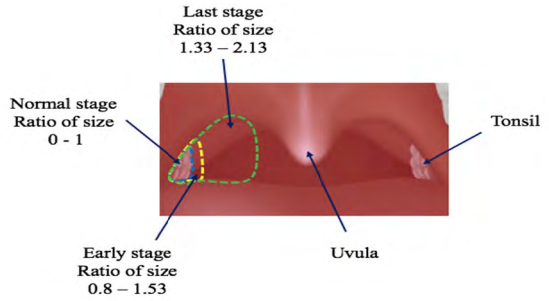


FIGURE 1. A ratio of size and color of inflamed tonsil.

A. PHYSICAL ANALYSIS OF TONSILLITIS

Information about the pattern of tonsillitis that could be used for detection and classification was learned from previous work [4].

Fig. 1 illustrates the ratios of sizes and colors of inflamed tonsils for the normal, early, and last stages of tonsillitis. As seen in Fig. 1, the features affected by tonsillitis include uvula size, tonsil size, and tonsil color. These three features can be used as input to a fuzzy logic system to determine whether the tonsillitis has formed. However, there are two problems that might bias tonsillitis detection. The first problem is the uvula size varies little when affected by tonsillitis.

This implies that it will be difficult to detect tonsillitis using only features from one input image, especially in its early stages. The second problem is that normal tonsils vary in size all the time. Referring to a medical reference [18], if the tonsil becomes inflamed, its size will continuously grow to a maximum size. Additionally, the rate of growth varies depending on the seriousness of the infection and the individual.

B. TONSILLITIS IMAGE ANALYSIS

In addition to the variation of tonsil size between normal and abnormal stages, tonsillitis exhibits different features in terms of color and intensity. As shown by the intensity values (red, green, and blue) of the pixels profiled along the horizontal axis in Fig. 2, tonsillitis exhibits higher intensity in blue channel and lower intensity in green and red respectively. This feature is useful for classification.

However, when viewing the color channels separately, tonsillitis is not differentiated in the blue and red channels as seen in bottom row in Fig. 3. The best segmentation of the boundary occurs in the green channel, shown in the middle image in the bottom row. The boundary of the tonsil considered to be affected by tonsillitis will be defined by the red channel.

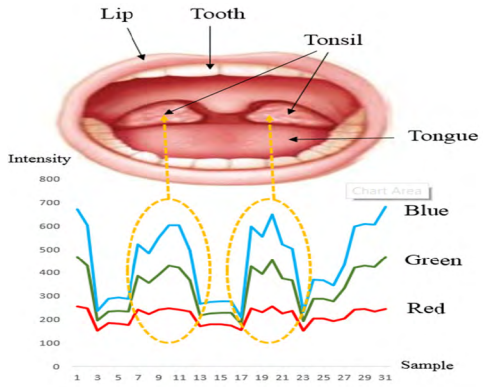


FIGURE 2. Intensity of tonsillitis detection.

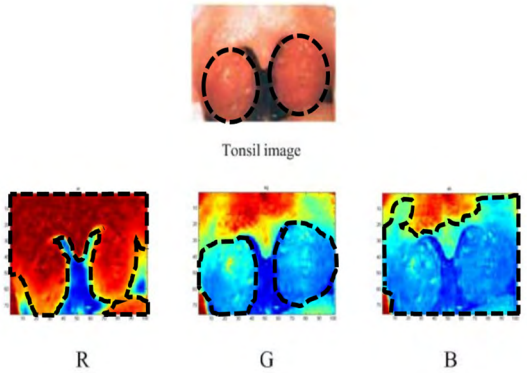


FIGURE 3. RGB color space of tonsils.

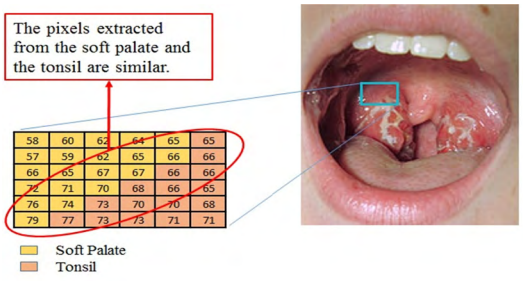


FIGURE 4. Tonsil pixel in the oral cavity.

Although segmentation of the tonsil boundary appears to be possible in general cases, in some cases of tonsillitis, the image exhibits similar intensities around the boundary, as shown by a sample in Fig. 4. With this type of image, it is difficult to distinguish and measure the extent of tonsillitis because of ambiguity in the image. The image exhibits blurry boundaries, textured objects, and non-homogeneous regions. To resolve this problem, we focus on local areas and reconsider the problem using tools such as watershed, gradient, and so on during preprocessing.

C. ANALYSIS FOR HARDWARE IMPLEMENTATION

A tonsillitis detection system requires a CCD camera to capture the input image in the oral cavity. The distance (D) is set to approximately 10 cm. The image is processed using the FPGA architecture shown in Fig. 5.

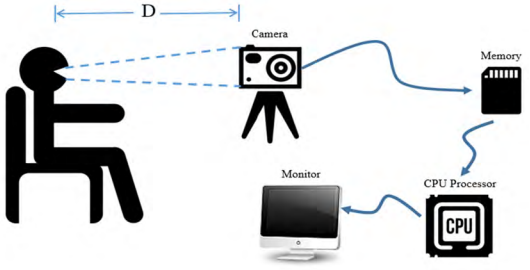


FIGURE 5. A conceptual tonsillitis detection hardware system.

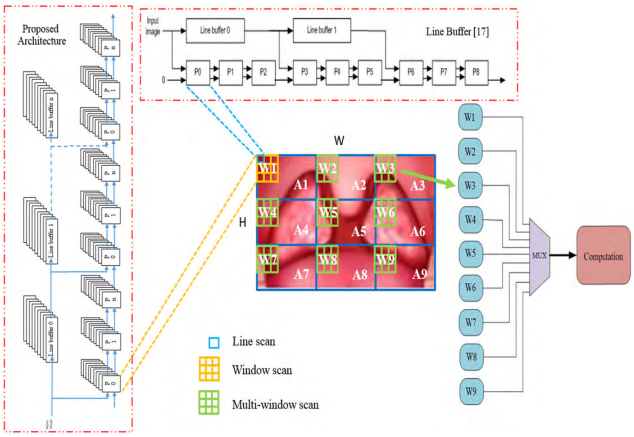


FIGURE 6. Scan methods: Line scan, Window scan, Multi-window scan.

In the design of an algorithm for the preprocessing step for tonsillitis segmentation, a window is generally created and scans from left to right and top to bottom in the image. The tonsil boundary is detected locally at each window position. This concept can be utilized in both software and hardware. With hardware, this processing occurs in series as shown in the upper row in Fig. 6. However, in the hardware design, it can be processed in parallel by increasing the number of processors so that a tonsillitis image can be divided into a set of areas A (= W×H) as shown in the middle of Fig. 6. Its formula is shown as follows.

$$A = H \times W \tag{1}$$

The scanning time (T) is obtained as follows.

$$T = \frac{A}{B} = \frac{A}{1} \tag{2}$$

where A is an image area which equals the image height (H) multiplied by the image width (W). Additionally, the time scan (T) equals the image area (A) divided by number of scanned blocks (B). In this study, the value of the scanned blocks (B) is equal to 1 because the process scans each pixel one by one.

In each block (A1, A2, A3, ... An), a window consisting of M×N pixels (Fig. 7) is scanned in for processing. A number of registers are constructed to access a buffer for local segmentation as seen by the hardware design shown on the left side of Fig. 6.

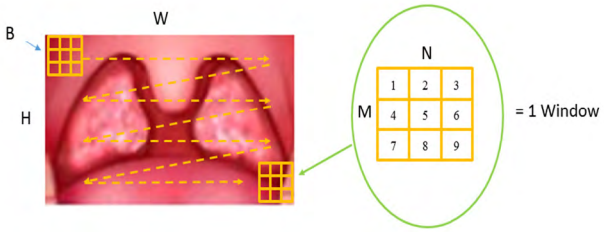


FIGURE 7. Window scan method.

The scanning of the window is shown by dash line in Fig. 7. A block consists of M×N pixels as shown in (3). The formula for the window scanning time is calculated in (4), in which the image area (A) is divided by M×N.

$$B = M \times N \tag{3}$$

$$T = \frac{A}{B} = \frac{A}{M \times N} \tag{4}$$

During scanning of the window, the size and color of the tonsil are calculated to detect the tonsil area by dividing the input image into A blocks, in which each block is scanned by an M×N mask. The outputs are then passed to the decision-making circuit. For example, if a 3×3 window is used, the processing time is calculated as follows.

$$A = \sum_{i=1}^{n} A_i = A_1 + A_2 + A_3 + \cdots + A_n \tag{5}$$

$$\text{If } n = 9, A = A_1 + A_2 + A_3 + \cdots + A_9 \tag{6}$$

$$C = \sum_{i=1}^{m} W_i = W_1 + W_2 + W_3 + \cdots + W_m \tag{7}$$

$$\text{If } m = 9, C = 9 + 9 + 9 + 9 + 9 + 9 + 9 + 9 + 9 = 81 \tag{8}$$

$$T = \frac{A}{C} = \frac{A_1 + A_2 + A_3 + \cdots + A_9}{81} \tag{9}$$

Where B, C, and W represent total pixels in a window, total pixels in multi-windows, and total number of windows respectively.

Comparing equations (9) and (4), the multi-window scanning method uses less time than the window scanning method by a factor of approximately 9. In addition, a comparison of equations (9) and (4) shows that scanning by the multi-window method uses less time by a factor of approximately 81.

III. PROPOSED SOFTWARE-BASED TONSILLITIS MONITORING SYSTEM

To diagnose tonsillitis, the region of interest is defined by tonsil size and color. The software-based tonsillitis monitoring system is designed to employ that information for tonsillitis detection as follows.

A. DATAFLOW OF SOFTWARE-BASED TONSILLITIS MONITORING SYSTEM

Based on the basic concept of relying on tonsil size and color changes, as shown by the flowchart in Fig. 8, the tonsil boundary needs to be accurately extracted. The process is started by preprocessing for noise reduction followed by tonsil boundary extraction. From the tonsil boundary extraction, color information obtained from green channel is used to globally detect the boundary candidates, and the candidates are evaluated locally to confirm the accurate position of tonsil boundary based on the gradient and watershed methods. The red color and size of the bounded region is then processed by fuzzy logic with the fuzzy model fixed in advance for making a decision about the stage of the tonsillitis.

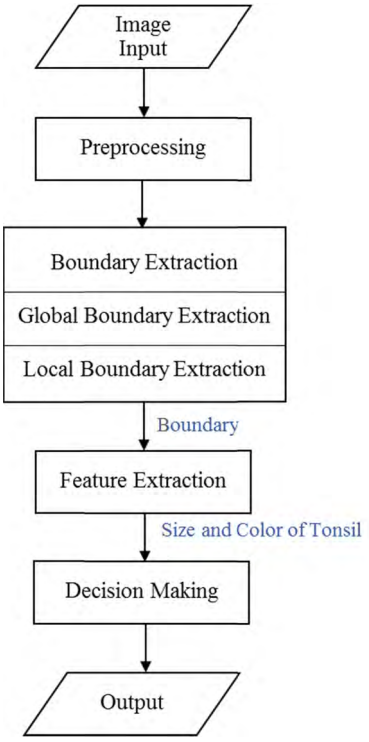


FIGURE 8. Block diagram of tonsillitis monitoring system.

B. PREPARATION OF TONSILLITIS DATA

To make a decision about the stage of the tonsillitis, features representing the stages have to be determined in advance. Users should collect data of healthy, early and last stages of tonsillitis and process them in the same manner using the proposed tonsillitis monitoring method. That is, to start from with preprocessing followed by boundary extraction and feature extraction. The data about tonsils size and the redness of color are finally calculated on average to define the range as shown by the sample range of sizes and color in Table 1 and 2 respectively. An example of part of the database used in data preparation is shown in Fig. 9.

C. PREPROCESSING AND TONSIL BOUNDARY DETECTION

Before beginning the process of tonsil boundary detection, the input images are filtered to diminish noise using the threshold technique. Boundary detection is then performed using global and local processing. As mentioned in Chapter II, the green channel is suitable for boundary detec-

TABLE 1. Size range of the tonsil gland.

Tonsil Gland	Size	Normalized Size (Ratio)
Normal stage (0 – Ns)	0 - 1	0 - 0.46
Early stage (E_{S1} – E_{S2})	0.8 - 1.53	0.37 – 0.71
Last stage (L_S – 1)	1.33 - 2.13	0.62 - 1

TABLE 2. Color range of the tonsil gland.

Tonsil Gland	Red Color Level	Normalized Size (Ratio)
Normal stage (0 – N_C)	131.3 - 255	0 – 0.49
Early stage (E_{C1} – E_{C2})	50 - 161	0.37 – 0.81
Last stage (L_C – 1)	0 - 90	0.65 - 1

FIGURE 9. Tonsil database.

tion and is used to find boundary candidates during global processing. The boundary candidates are then confirmed or removed during local processing. The boundary determined using the watershed and gradient methods has some error due to low pixel contrast near the boundaries. These errors are expected to be reduced during local processing. The flowchart for the local processing of tonsil boundary detection is depicted in Fig. 10.

Local processing for tonsil boundary detection starts by converting a tonsil image into a gray scale image as shown in Fig. 10. The gray scale image is then transformed into a gradient magnitude image using a Sobel edge detector, and the gradient magnitude is then transformed using watershed to enhance the edges in the process of calculating the gradient magnitude. At the next step of marking the foreground of the object, the background and foreground of the tonsil image are subtracted using a global gray-scale threshold. Tonsil candidates are then extracted using a local threshold, which is located at the highest range of the global gray-level threshold. In the last step of tonsil segmentation, another local threshold is employed to select tonsil boundaries from the tonsil boundary candidates.

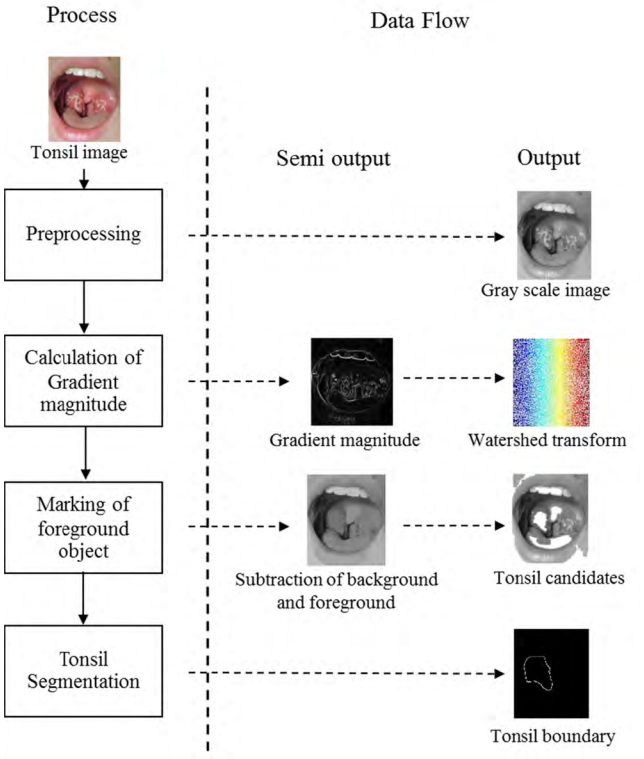


FIGURE 10. Local process of tonsil boundary detection.

D. FEATURE EXTRACTION

According to medical professionals [19] information about redness and the area of the tonsil boundary are employed as features for tonsillitis diagnosis. We propose to implement a practical expert system for tonsillitis detection so these key features are used in this paper. There exist many feature extraction methods such as Grayscale Level Co-Occurrence Matrix (GLCM) [20], Color Histogram [21], Edge Pixel Neighborhood Information (EPNI) [22], Histograms of Edge Directions (HED) [23], Scale Invariant Feature Transform (SIFT) [24], Grid Methods [25], Contour Based Method [26], Region-Based Method [27], and so on. In the feature extraction process for this system, users are free to select the appropriate method to extract features in their application. In this paper, the authors recommend utilizing GLCM as an appropriate feature extraction method because of its simplicity and robustness to noise.

E. DECISION MAKING

Since our goal is to develop an expert system for automated tonsillitis detection, the fuzzy model used for decision making in this paper has to be created based on medical knowledge. Referring to [19] and [28], which are considered medical knowledge, changes of tonsil size and red color are important features for classifying a tonsil into normal (S_0, C_0), early (S_1, C_1), and last (S_2, C_2) stages as shown in Fig. 11(a) and 12(a), respectively. Both information about tonsil size and red color intensity are then

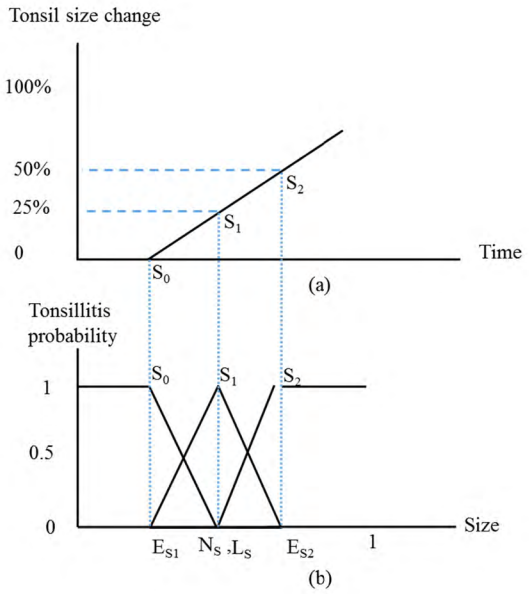


FIGURE 11. Membership function of tonsil size.

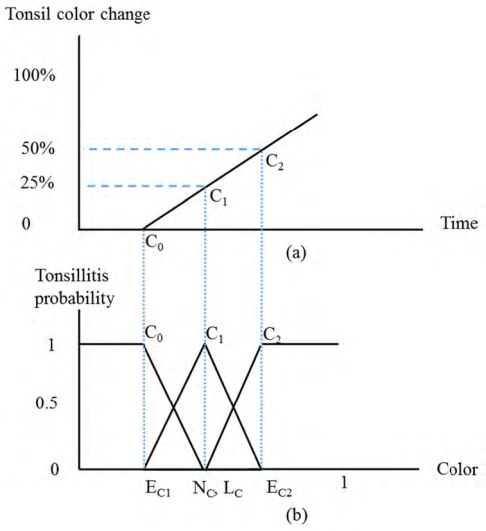


FIGURE 12. Membership function of tonsil color range.

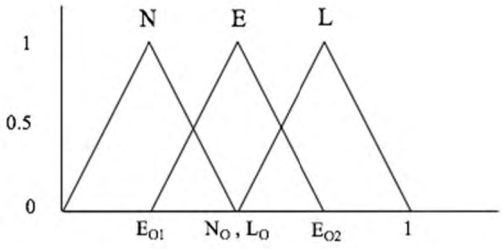


FIGURE 13. Membership function of tonsillitis detection.

normalized to the same range of 0-1 as shown in horizontal axes of Fig. 11(b) and 12(b), respectively. Medically, tonsillitis status, which is divided into the S_0 (normal), S_1 (early), and S_2 (last) stages, has the highest probabilities of 0%, 25% and 50%, respectively, of occurring with tonsil size change, as shown in Fig. 11(a), and probabilities of 0%, 25%, and 50%, respectively, of occurring with tonsil-color change, as shown in Fig. 12(a). Subsequently, the high-probability stages (S_0, S_1, S_2, C_0, C_1, and C_2) are projected as 1 for the highest probability (shown by dash line) of the membership function, and the probability of each stage decreases gradually from the highest probability, as shown in Fig. 11(b) and Fig. 12(b). In the membership model, E_{s1}, E_{s2}, Ns, and Ls are assumed to be the minimum of the early stage, the maximum of early stage, the maximum of normal stage, and the minimum of last stage, respectively as shown in Fig. 11. Similarly, for the red color intensity shown in Fig. 12, E_{c1}, E_{c2}, N_c, and L_c are assumed to be the minimum of the early stage, the maximum of the early stage, the maximum of normal stage, and the minimum of last stage, respectively. The probability of being in the normal stage when tonsil size is in the range between 0 to E_{s1} and of being in a last stage when in the range between E_{s2} to 1 is assumed to be 100%. Their amplitudes are set at 1.

Their probabilities are gradually decreased during the overlapping period with the early stage such that the slopes vary from S_0 to N_s and from S_2 to L_s. For the early stage, the range is from E_{s1} to E_{s2} on the horizontal axis. However, the probability of being in the early stage reaches the highest level at S_1 so that the probability becomes 1, and the probability decreases gradually in both left and right sides, reverting to the probabilities of the normal and last stages. The model of the early stage is therefore designed appear like a bisecting triangle, of which the base is from E_{s1} to E_{s2}, and the tip has an amplitude of 1, as shown in Fig. 11(b). Similarly, the model for the membership function based on the red color intensity is plotted as shown in Fig. 12(b).

Although both tonsil size and red color intensity are normalized separately as explained above, the ranges of normal, early, and last stages of both features do not have the same standard. We need to normalize again before determining the stage of tonsillitis.

To normalize tonsil size and color level into the same scale, we compute the averages as shown in the following equations. The results are plotted in Fig. 13.

$$E_{O1} = \frac{E_{S1} + E_{C1}}{2} \tag{10}$$

$$E_{O2} = \frac{E_{S2} + E_{C2}}{2} \tag{11}$$

$$N_O = \frac{N_S + N_C}{2} \tag{12}$$

$$L_O = \frac{L_S + L_C}{2} \tag{13}$$

On the other hand, the medical knowledge for tonsillitis diagnosis is converted into a fuzzy inference rule as shown in the Table 3. These inference rules are converted into the following rules.

TABLE 3. Fuzzy inference rules.

Size	Color		
	Normal	Early	Last
Normal	Normal	Early	Last
Early	Early	Early	Last
Last	Last	Last	Last

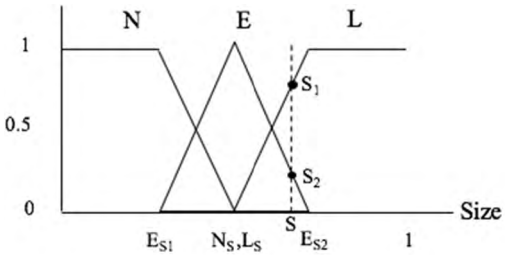


FIGURE 14. Example of size input.

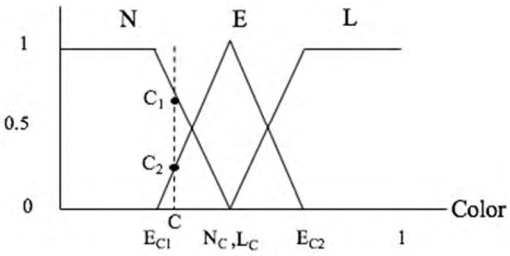


FIGURE 15. Example of color input.

1. IF size = N AND color = N THEN Decision = N
2. IF size = N AND color = E THEN Decision = E
3. IF size = N AND color = L THEN Decision = L
4. IF size = E AND color = N THEN Decision = E
5. IF size = E AND color = E THEN Decision = E
6. IF size = E AND color = L THEN Decision = L
7. IF size = L AND color = N THEN Decision = L
8. IF size = L AND color = E THEN Decision = L
9. IF size = L AND color = L THEN Decision = L

For instance, suppose S and C are size and red color intensity, respectively. The S and C values S_1, S_2, C_1, and C_2 then become values in the amplitude axis as shown in Fig. 14 and 15. These values are applied in the fuzzy inference rules shown in Table 3, and the true rules are obtained as follows.

4. IF size = E AND color = N THEN Decision = E (S_2,C_1)
5. IF size = E AND color = E THEN Decision = E (S_2,C_2)
7. IF size = L AND color = N THEN Decision = L (S_1,C_1)
8. IF size = L AND color = E THEN Decision = L (S_1,C_2)

The minimum and maximum values of those rules are as follows.

1) Input fuzzy

4. Min (S_2 , C_1) = S_2
5. Min (S_2 , C_2) = C_2
7. Min (S_1 , C_1) = S_1
8. Min (S_1 , C_2) = C_2

2) Output fuzzy

N = 0
E max = S_2
L max = S_1

These values are indicated by the colored area in Fig. 16. The center of gravity (COG) of the colored areas is calculated using the following equations.

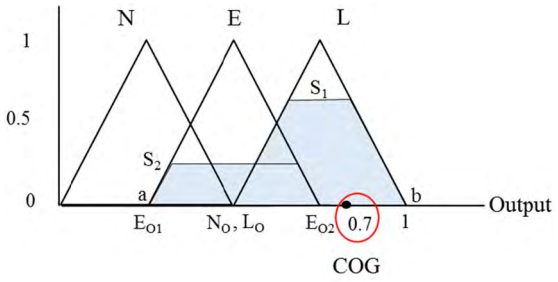


FIGURE 16. Example of output.

$$COG = \frac{\int_a^b \mu A(x)\, x dx}{\int_a^b \mu A(x)\, dx} \tag{14}$$

$$COG = \frac{\sum_{x=a}^{b} \mu A(x)\, x}{\sum_{x=a}^{b} \mu A(x)} \tag{15}$$

$$COG = \frac{(N_O \times S_2) + (L_O \times S_2) + (E_{O2} \times S_1)}{(S_2 + S_2 + S_1)} \tag{16}$$

$$COG = \frac{(0.5 \times 0.2) + (0.5 \times 0.2) + (0.8 \times 0.6)}{(0.2 + 0.2 + 0.6)} \tag{17}$$

In this case, the COG of 0.7 is obtained and finally plotted in Fig. 16, which indicates the probability.

IV. PROPOSED HARDWARE-BASED TONSILLITIS MONITORING SYSTEM

In some applications, such as in using a personal mobile device for Tonsillitis detection, the system requires a small size, a light weight, and a low cost with high computation speed. A hardware implementation based on the software algorithm is one of solutions. This would help to decrease the size, weight, cost and processing time. In hardware design, it is absolutely better to select a data processing hardware type that can efficiently process a large amount of data. That pipeline is chosen in this paper. According to the aforementioned algorithm, three processes are required. The first one is preprocessing. The second one is feature extraction such as Gabor filter [29], [30] and Edge detection technique [31]–[35]. And the last one is classification using fuzzy logic [36]. The pipeline and the required processes are designed at the first step in the block diagram shown

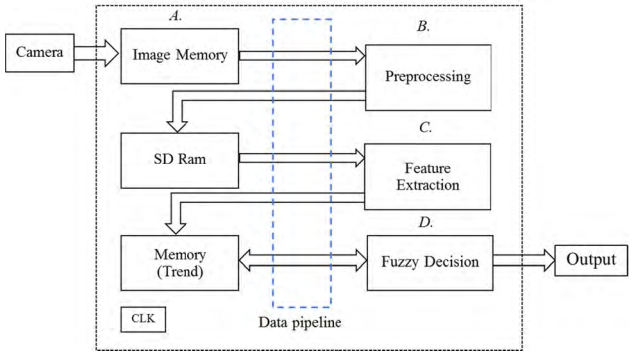


FIGURE 17. Overall architecture of tonsillitis detection.

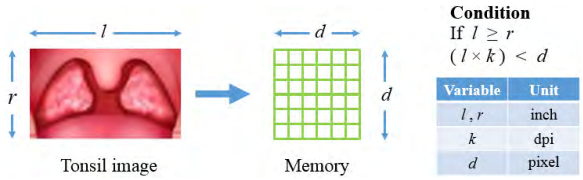


FIGURE 18. Image memory.

in Fig. 17. Each block, which needs to function like the software system mentioned above, is designed as explained in the following sub-sections.

It starts with capturing oral images using a camera in the upper left of Fig. 17. The images are sent to storage via the data pipeline in the preprocessing block. The output of preprocessing is temporarily stored in SD RAM. The stored images are processed to extract features, which include the tonsil size and color in the feature extraction block. The output is passed through the pipeline for storage in memory. Finally, the tonsillitis stages are determined in the fuzzy decision block, where a diagnosis is output. The details of each block are described as follows.

A. IMAGE MEMORY

In selecting image memory, users should determine the needs of their application such as image size, lens focal length usage environment and so on.

The image specifications that a user should look for include the memory resolution, data read/write speed, and so on. To determine the resolution, as shown in Fig. 18 the field of view (for example $r \times l$ square inch; $l \geq r$) is converted into pixels ($(l \times k) \times (r \times k)$) using the perspective transformation based on the lens focal length. Users may find the minimum image area corresponding to a pixel, i.e., the image resolution $(d \times d)$ is determined to cover the image area ($(l \times k) \times (r \times k)$). The image size determines the processing time per frame. The number of bits in an image should be divided by the total expected processing time so that a specification for data read/write speed is obtained. With regards to the operating voltage, usage conditions such as mobile, at home, in hospital, and so on determine battery weight and power. The operating voltage has to match the specifications of the battery and the design board.

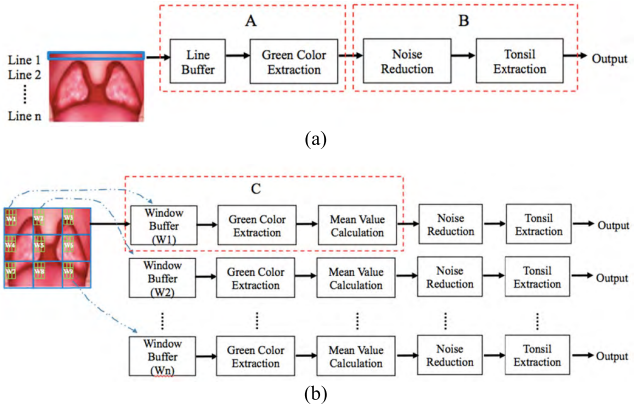


FIGURE 19. Scan architecture. (a). Series architecture (b). Parallel architecture.

B. PREPROCESSING

To enhance and extract a tonsillitis diagnosis from an image that is assumed to be noisy, there are two approaches recommended in this paper for the hardware design, i.e., the series and parallel architectures shown in Fig. 19. In the series approach shown in Fig. 19 (a), a line buffer covering one row of an image is installed. The temporarily stored data of each line is processed to extract only the green channel, reduce noise by filtering, identify the tonsil, and perform tonsillitis extraction. In the parallel processing approach shown in Fig. 19 (b), an appropriate number of buffers are provided instead of the line buffer. The hardware for preprocessing is similar to the series approach, which links the buffers with the blocks set up in parallel. Users who apply this method in applications requiring efficiency should select the parallel approach, with the trade-off in the cost of the devices. The details are described in the following sub-section.

1) LINE BUFFER AND GREEN COLOR EXTRACTION

In the series architecture, image data is fed through a buffer to the next process for extraction of the green channel, as shown within the dash-line rectangle (A) in Fig. 19 (a). The buffer length is needed for optimizing the feed speed and process-waiting time. If the buffer length is as long as the width of the image, the data feeding time and device investment cost are issues to consider. On the other hand, if only a one-bit buffer is installed, the next process may wait for data due to the delay in the data feed. This paper therefore optimizes the buffer length to be one line of the image, which is a so called "line buffer" as shown in Fig. 20. Data in one line of an image is input to the store while waiting for the next process. In addition, the green channel is automatically picked up from registers for the process of Green Color Extraction inside dashed-line rectangle (A) in Fig. 19 (a), and the green intensity value is transferred to the next process in the dashed-line rectangle (B).

2) NOISE REDUCTION AND TONSIL EXTRACTION

Before extracting the tonsil boundary, it is useful to reduce noise with a noise-reduction filter. The process of noise

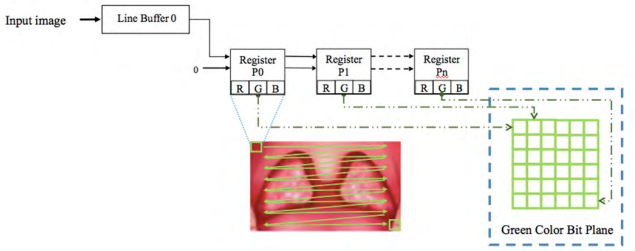


FIGURE 20. Line buffer and green color extraction.

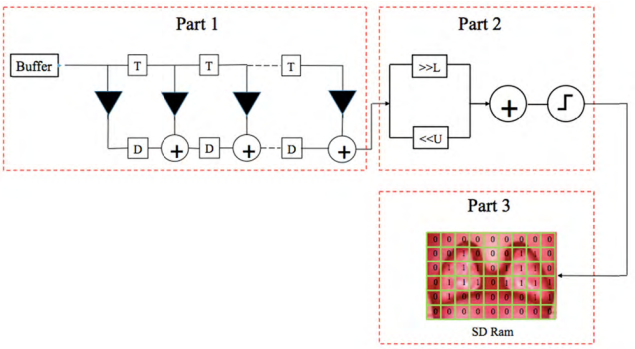


FIGURE 21. Noise reduction and tonsillitis extraction, and SD Ram.

reduction and tonsil extraction is shown within the dashed-line rectangle (B) in Fig. 19 (a). Finite Impulse Response (FIR), which is a well-known, simple filter, is recommended for use in the hardware design of noise reduction is shown in Part 1 of Fig. 21, and the circuit for scanning and detecting the tonsil boundary is shown in Part 2 in Fig. 21. The intensity of the green channel stored in the buffer is filtered in Part 1 of Fig. 21 and the signal without noise is filtered in the determined range of L (low) - U (up). An intensity that is higher than the threshold is classified as shown in Part 2 of Fig. 21. Pixels above the threshold are defined as tonsil size represented by "1" in the SD Ram in Part 3 of Fig. 21, and the background is revealed by "0".

3) WINDOW BUFFER, GREEN COLOR EXTRACTION AND MEAN VALUE CALCULATION

In processing using the parallel architecture approach, blocks for the window buffer, green color extraction and mean value calculation are installed to perform noise reduction and are shown surrounded by the dash-line rectangle (C) in Fig. 19 (b). First, an image is divided into many blocks, and each block is scanned by 3×3 window in all the memory registers (P0, P1, P2, ... Pn) as shown in Fig. 22. The memory registers are linked to the circuit for green color extraction and the mean value calculation, respectively. The mean value is used to represent the center pixel of the window, and thus the method functions as a low-pass filter for filtering noise. The noise-reduced image is then transferred to the next process for tonsil-boundary detection, which was mentioned in the series approach.

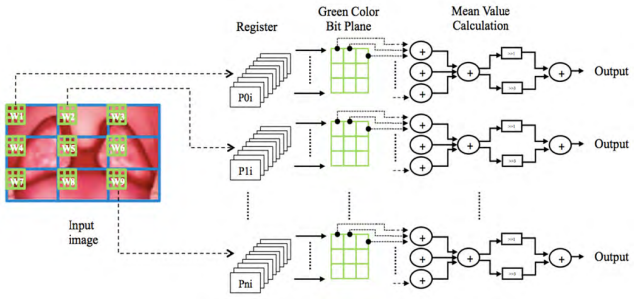


FIGURE 22. Window buffer.

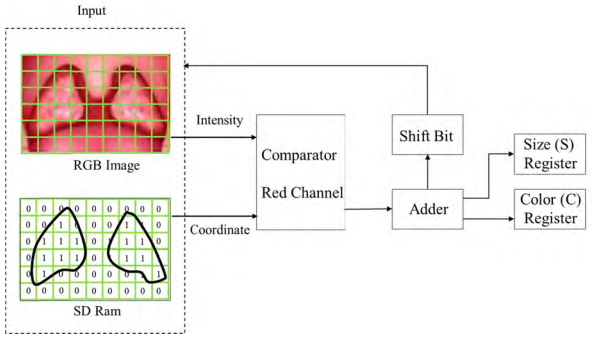


FIGURE 23. Tonsillitis detection architecture - feature extraction.

C. FEATURE EXTRACTION

In the process of feature extraction, the RGB image and the tonsil-area bit plane obtained from image memory (explained in A) and SD Ram (explained in B (2)), respectively, are input to extract the red color intensity and tonsil area in order to accumulate (by adder) total amounts in size register and color register as shown in Fig. 23. Simultaneously, shift bit functions serve to shift the image pixels.

D. FUZZY DECISION

Similar to the logic of the aforementioned software, the threshold for fuzzy decisions is designed based on the logic of the fuzzy membership function depicted in Fig. 13. To implement fuzzy processor hardware, users are free to employ existing architectures that are suitable to their applications, such as those of Mamdani [32], Sugeno [37], and so on. Comparing the Mamdani and Sugeno architecture-design styles, the Mamdani architecture, which is popularly utilized, determines the COG of the maximum and minimum values as output, while the Sugeno architecture calculates the output from weight of rule and level of output. Based on performance, the accuracy of the Mamdani architecture is reliable, while the number of gates and power consumption of the Sugeno architecture has a greater advantage. Users should consider the advantages and disadvantages to determine which is suitable for their application before selection. In practice, input data including the number of pixels representing the tonsil size (S_t), the red channel intensity (C_t) and data from earlier times (S_{t-1}, C_{t-1}) are fed to membership function circuits (MFC) as shown in the fuzzy input part on left sides of Fig. 24 and Fig. 25.

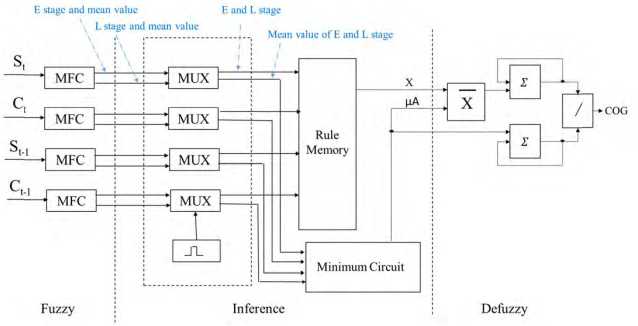


FIGURE 24. Architecture of the fuzzy processor (Mamdani).

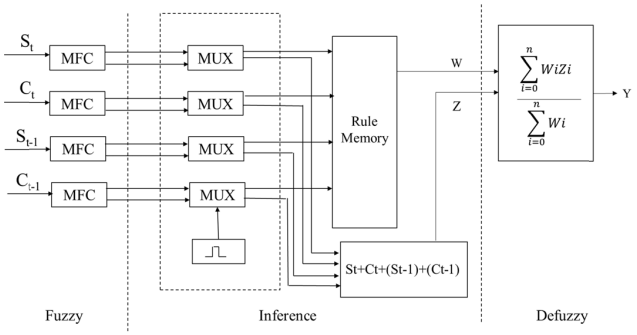


FIGURE 25. Architecture of the fuzzy processor (Sugeno).

According to the membership function shown in Fig. 13, values of N, E, and L are obtained and transferred to MUXs in the inference part, and a signal representing the "E" and "L" stages and another showing its mean value are forwarded to the rule-based memory and the minimum circuit, respectively, as shown in Fig. 24 and Fig. 25. While the "E" and "L" of both features are used to determine the stage of tonsillitis based on the fuzzy rules shown in Table 3, mean values are selected the minimized one at the minimum circuit. Consequently, the final decision of fuzzy logic is made by the circuit in defuzzy part of Fig. 24 and Fig. 25 based on equation (16).

V. EXPERIMENTAL RESULTS

To evaluate the performance of the proposed tonsillitis monitoring and detection system, we constructed the prototype system shown in Fig. 26 according to the specifications provided in Table 4. The high resolution sensor that was selected is intended for applications where bright light is available, and a popular basic CPU was chosen. To fix the patient's head and open mouth so they would remain stationary, the patients place their mouths over a hole on the black box, and fix their heads in place using a head-locking band as shown in Fig. 27. The patient's head is guaranteed to maintain the same position with an open mouth. By programing the ISE, the circuit modules for tonsillitis detection based on block searching are designed as shown in Fig. 28.

In general, designed hardware is evaluated by device capacity which is measured by number of logic cells. In this paper designed Mamdani- and Sugeno-based hardware consume logic cells by 3K-19K, and 2.9K-18K, respectively,

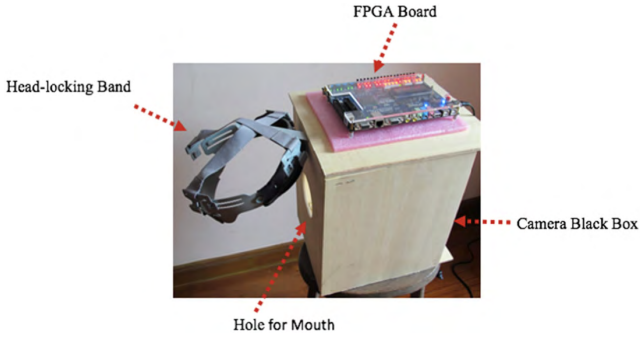


FIGURE 26. Constructed prototype system.

TABLE 4. Experimental conditions.

Item	Configuration
CCD Sensor	Resolution of 1280 × 960 pixels with 256 gray levels
Image Memory	Size of 480 × 640 pixels with 256 pixel gray levels
CPU	Core i7 processor 1.8 GHz, Ram 4 GB
Software	Matlab v. 2012, ISE

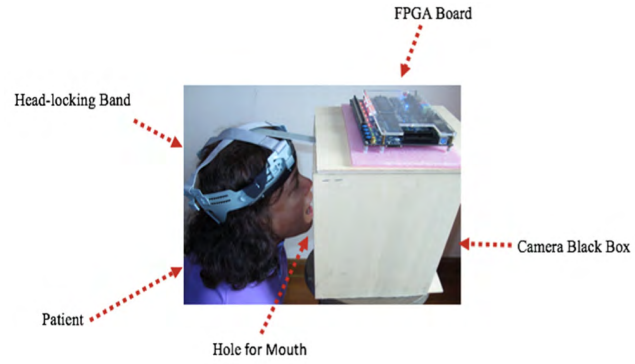


FIGURE 27. The prototype with tester.

TABLE 5. Implementation results using memory for tonsillitis detection.

	Logic Cells	Power (mW)
Mamdani	3 K – 19 K	276 - 5028
Sugeno	2.9 K – 18 K	276 - 4776

and power by 276-5028 mW and 276-4776 mW, respectively, as shown in Table 5.

Sample wave forms of data transitions during tonsillitis detection are illustrated by the outputs from the circuit and are shown in Fig. 29. The data input is from the multi-window scan, which is separated into 9 blocks. With each iteration, input data are fed to the next stage, called the pipeline. Then, the data is preprocessed to extract features and forwarded to the fuzzy decision module.

All tonsil images for learning and testing are shown in Fig. 30. These tonsil images were collected at a hospital in Bangkok, Thailand, and some were retrieved from open-access websites. The experimental results from our software

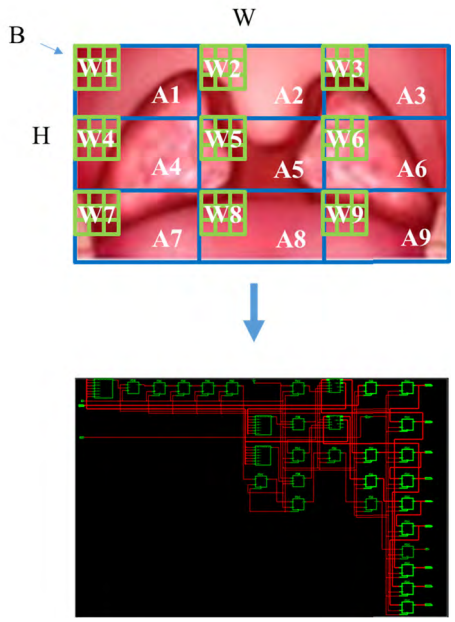


FIGURE 28. Circuit modules using multi-window scanning.

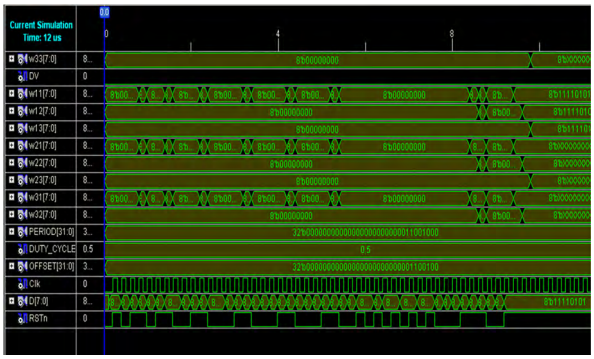


FIGURE 29. Waveform of data transitions in tonsillitis detection.

and hardware systems are shown in Table 6 and 7. A total of 137 tonsil images, including 24 depicting early stage and 113 depicting last stage tonsillitis, were mixed with 22 normal stage images. Out of 159 images, 30 were used for training, and the rest for testing. The results generated by the software and hardware are shown in Table 6, and 7, where the 2nd column shows sub-totals for the number of normal, early, and last-stage tonsillitis images, respectively. The decision results for the Mamdani and Sugeno systems are depicted by circles and triangles, respectively, as revealed in the normal-stage (N), early-stage (E), and last-stage (L) in the following columns. The accuracy for determining normal, early and last stages was 86.3, 75, and 96.4% respectively.

VI. DISCUSSION

This paper proposes an automated tonsillitis detection and monitoring system using both software and hardware.

To implement the software system, the green channel, which is already proven for detecting obvious features, is selected, and its gradient watershed features are extracted. The extracted features are classified using fuzzy logic into

FIGURE 30. Tonsillitis images for testing.

TABLE 6. Implementation results using software for tonsillitis detection.

State	Tonsil Image	Decision					
		M	S	M	S	M	S
Normal	22	20	19	2	3	0	0
Early	24	3	5	21	19	0	0
Last	113	2	1	1	2	110	110
Accuracy (%)		90.9	86.3	87.5	79.1	97.3	97.3

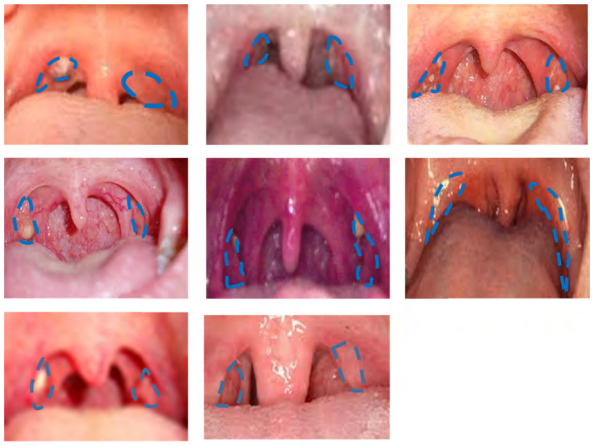
Note: M= Mamdani, S= Sugeno

normal tonsils, and early and last stage tonsillitis. The software system shows acceptable accuracy, especially for early stage tonsillitis, which is vitally important for patients. One reason for the accuracy is the software system is running on desktop computer, which has full ability to compute, while the hardware system generates a few errors due to computational limitations. The software system is considered to work appropriately at hospitals and clinics with large spaces and

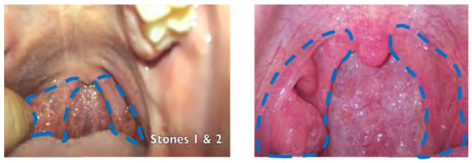
TABLE 7. Implementation results using hardware for tonsillitis detection.

State	Tonsil Image	Decision					
		M	S	M	S	M	S
Normal	22	19	18	3	4	0	0
Early	24	6	8	18	16	0	0
Last	113	2	2	2	3	109	108
Accuracy (%)		86.3	81.8	75	66.6	96.4	95.5

Note: M= Mamdani, S= Sugeno



(a) Early stage detected as normal stage



(b) Last stage detected as normal stage

(c) Last stage detected as early stage

FIGURE 31. Samples of error.

services for many patients. On the other hand, the hardware system, which is needed to be compact in size, should be matched with home applications or portable use.

In the hardware implementation, as with the aforementioned software system, the green channel is utilized, and the techniques of block scanning, parallel processing and piping between processes are applied in order to realize real-time processing.

To evaluate the performance of our system, the hardware system was designed and implemented on FPGAs. The experimental results for 159 samples are shown in Table 7, where the accuracy of identifying normal, early, and last stages of Mamdani is 86.3, 75, and 96.4% respectively while the accuracy of Sugeno is 81.8, 66.6, and 95.5% respectively. Examples of errors where a normal stage is detected as early, an early stage is detected as normal, and a last stage is detected as early are illustrated in Fig. 31 (a), (b), and (c), respectively. In investigating the causes of error, all the samples were found to have low-contrast, and all the errors occur because of an inappropriate threshold value, which is not matched with those low-contrast images. To avoid these errors, we reconsider the threshold value and perform experiments by manually applying a threshold that was learned from other data for the same patient. All the samples were then successfully classified into the correct stage. It is recommended that users set a patient specific threshold when using the system in order to minimize errors.

For fuzzy decision making, Mamdani- and Sugeno-style rules were demonstrated in this paper. Mamdani-style is a more complicated computation than Sugeno-style but gets better accuracy, as shown in Table 6 and 7. Users should consider selecting one of these methods.

On the other hand, the experimental results reveal the system consumes 16 Megabytes of memory, 3-19 K logic gates, and approximately 5,028 milli watts power. There are trade-offs in computation time, and some redundancy of memory and logic gate usage considered in time sharing against the loss reduction, which is left for future work.

VII. CONCLUSION

This paper presents a method for the design and implementation of an automated system for tonsillitis monitoring and detection using fuzzy logic. The system is demonstrated both in software and in hardware. The software system utilizes watershed and gradient transformations for image enhancement and fuzzy logic for decision making. In the hardware design, data is pipelined between processes, and parallel processing is employed for feature extraction by dividing the image into an appropriate number of blocks. The proposed architecture using fuzzy processors of Mamdani and Sugeno has been designed using VHDL language for simulation on ISE. The experimental results performed by the software and hardware systems with 159 samples have 96.4% and 91.8% accuracies, respectively.

REFERENCES

[1] WHO and UN Partners. (Jan. 2015). *Thailand: WHO Statistical Profile, Country Statistics and Global Health Estimates*, accessed on Jun. 15, 2015. [Online]. Available: http://www.who.int/gho/countries/tha.pdf?ua=1

[2] Global Health Observatory (GHO) Data. WHO. *Care Seeking for Pneumonia, WHO*, accessed on Jun. 15, 2015. [Online]. Available: http://www.who.int/gho/child_health/prevention/pneumonia_text/en/

[3] K. Horio *et al.*, "Mucous membrane diseases diagnosis support system using probabilistic relaxation," in *Proc. Int. Symp. Commun. Inf. Technol. Conf. (ISCIT)*, Tokyo, Japan, 2010, pp. 287–291.

[4] P. Phensadsaeng, P. Kumhom, and K. Chamnongthai, "A computer-aided-diagnosis of tonsillitis using tonsil size and color," in *Proc. Int. Symp. Circuits Syst. (ISCAS)*, Kos, Greece, 2006, pp. 5563–5566.

[5] P. Phensadsaeng and K. Chamnongthai, "A feature extracting method using gradient and watershed for tonsillitis diagnosis," in *Proc. Conf. Intell. Signal Process. Commun. Syst. (ISPACS)*, Kuching, Malaysia, 2014, pp. 297–301.
[6] S. Jin, D. Kim, T. T. Nguyen, D. Kim, M. Kim, and J. W. Jeon, "Design and implementation of a pipelined datapath for high-speed face detection using FPGA," *IEEE Trans. Ind. Informat.*, vol. 8, no. 1, pp. 158–167, Feb. 2012.
[7] C.-R. Chen, W.-S. Wong, and C.-T. Chiu, "A 0.64 mm^2 real-time cascade face detection design based on reduced two-field extraction," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 19, no. 11, pp. 1937–1948, Nov. 2011.
[8] V. Suse and D. Ionescu, "A real-time reconfigurable architecture for face detection," in *Proc. Int. Conf. ReConFigurable Comput. FPGAs (ReConFig)*, Riviera Maya, Mexico, Dec. 2015, pp. 1–6.
[9] C. Gao, S.-L. L. Lu, T. Suh, and H. Lim, "Field programmable gate array-based Haar classifier for accelerating face detection algorithm," *IET Image Process.*, vol. 4, no. 3, pp. 184–194, Jun. 2010, doi: 10.1049/iet-ipr.2009.0030.
[10] A. Leelasantitham and S. Kiattisin, "A diagnosis of tonsillitis using image processing and neural network," *Int. J. Appl. Biomed. Eng.*, vol. 2, no. 2, pp. 36–42, 2009.
[11] C. Gao and S.-L. Lu, "Novel FPGA based Haar classifier face detection algorithm acceleration," in *Proc. Int. Conf. FPL. Field Program. Logic Appl.*, Heidelberg, Germany, 2008, pp. 373–378.
[12] T. Aoki, E. Hosoya, T. Otsuka, and A. Onozawa, "A novel hardware algorithm for real-time image recognition based on real AdaBoost classification," in *Proc. Int. Symp. Circuits Syst. (ISCAS)*, Seoul, South Korea, 2012, pp. 1119–1122.
[13] K. Padmaja and T. N. Prabakar, "FPGA based real time face detection using AdaBoost and histogram equalization," in *Proc. Int. Conf. Adv. Eng., Sci. Manage. (ICAESM)*, Nagapattinam, India, 2012, pp. 111–115.
[14] M. Yagi and T. Shibata, "An image representation algorithm compatible with neural-associative-processor-based hardware recognition systems," *IEEE Trans. Neural Netw.*, vol. 14, no. 5, pp. 1144–1161, Sep. 2003.
[15] R. C. Luo and H.-H. Liu, "Design and implementation of efficient hardware solution based sub-window architecture of Haar classifiers for real-time detection of face biometrics," in *Proc. Int. Conf. Mechatronics Autom. (ICMA)*, Xi'an, China, 2010, pp. 1563–1568.
[16] J. Cho, B. Benson, and R. Kastner, "Hardware acceleration of multi-view face detection," in *Proc. IEEE 7th Symp. Appl. Specific Processors (SASP)*, San Francisco, CA, USA, Jul. 2009, pp. 66–69.
[17] P. Phensadsaeng, W. Chiracharit, and K. Chamnongthai, "A VLSI architecture of color model-based tonsillitis detection," in *Proc. Int. Symp. Commun. Inf. Technol. (ISCIT)*, Incheon, South Korea, 2009, pp. 693–696.
[18] P. Phensadsaeng, P. Kumhom, and K. Chamnongthai, "VLSI architecture for tonsillitis detection system," in *Proc. 9th Int. Conf. Intell. Technol. (InTech)*, Surat Thani, Thailand, Jan. 2008.
[19] S. K. Ng, D. L. Y. Lee, A. M. Li, Y. K. Wing, and M. C. F. Tong, "Reproducibility of clinical grading of tonsillar size," *Int. J. Otorhinolaryngol. Head Neck Surgery*, vol. 136, no. 2, pp. 159–162, Feb. 2010.
[20] B. Pathak and D. Barooah, "Texture analysis based on the gray-level co-occurrence matrix considering possible orientations," *Int. J. Adv. Res. Elect. Electron. Instrum. Eng.*, vol. 2, pp. 4206–4212, Sep. 2013.
[21] N. Sharma, P. Rawat, and J. Singh, "Efficient CBIR using color histogram processing," *Int. J. (SIPIJ)*, vol. 2, no. 1, pp. 94–112, Mar. 2011, doi : 10.5121/sipij.2011.2108.
[22] P. Sharma, M. Diwakar, and N. Lal, "Edge detection using Moore neighborhood," *Int. J. Comput. Appl.*, vol. 61, no. 3, pp. 26–30, Jan. 2013.
[23] G. Wang, Z. Xiong, Y. Lan, and C. Ye, "Object detection using edge direction histogram features," *J. Inf. Technol.*, vol. 12, no. 24, pp. 8275–8280, 2013.
[24] Y.-Y. Wang, Z.-M. Li, L. Wang, and M. Wang, "A scale invariant feature transform based method," *J. Inf. Hiding Multimedia Signal Process.*, vol. 4, no. 2, pp. 73–89, Apr. 2013.
[25] Z. Hu, "An image processing and reconstruction method using multiple grid SVR," *J. Comput. Inf. Syst.*, vol. 11, pp. 4491–4496, Jun. 2015.
[26] F. M. Abubakar, "A study of region-based and contour-based image segmentation," *Signal Image Process., Int. J. (SIPIJ)*, vol. 3, no. 6, pp. 15–22, Dec. 2012.
[27] M. Kaur and P. Goyal, "A review on region based segmentation," *Int. J. Sci. Res.*, vol. 4, pp. 3194–3197, Apr. 2015.
[28] B. K. Venkatesha, B. S. Yogeesha, and M. Asha, "Clinical grading of tonsils: Does it truly represent total tonsil volume in patients with recurrent tonsillitis," *Int. J. Otorhinolaryngol. Head Neck Surgery*, vol. 3, pp. 354–358, Jan. 2017.
[29] A. Oksiya, "Medical image segmentation for disease detection using digital filter," *Int. J. Tech. Res. Appl.*, vol. 2, pp. 193–196, Aug. 2014.
[30] R. D. Thakur, "Real time VHDL design of high speed Gabor filter for disease detection," *Int. J. Adv. Res. Comput. Sci. Softw. Eng.*, vol. 4, pp. 229–234, Dec. 2014.
[31] S. R. Dixit and A. Y. Deshmukh, "Design strategies for various edge detection techniques for disease detection," *Int. J. Sci. Eng. Appl. Sci.*, vol. 1, pp. 498–505, Nov. 2015.
[32] P. R. Possa, S. A. Mahmoudi, N. Harb, C. Valderrama, and P. Manneback, "A multi-resolution FPGA-based architecture for real-time edge and corner detection," *IEEE Trans. Comput.*, vol. 63, no. 10, pp. 2376–2388, Oct. 2014.
[33] M. Qasaimeh, A. Sagahyroon, and T. Shanableh, "FPGA-based parallel hardware architecture for real-time image classification," *IEEE Trans. Comput. Imag.*, vol. 1, no. 1, pp. 56–70, Mar. 2015.
[34] G. Liu and Z. Shi, "Embedded implementation of real-time skin detection system," in *Proc. Int. Conf. Transp., Mech., Elect. Eng. (TMEE)*, Changchun, China, Dec. 2011, pp. 2463–2466.
[35] D. Sangeetha and P. Deepa, "FPGA implementation of cost-effective robust Canny edge detection algorithm," *J. Real-Time Image Process.*, vol. 13, pp. 1–14, Mar. 2016, doi: 10.1007/s11554-016-0582-2.
[36] R. M. Fikry, S. A. Shehata, S. M. Elaraby, M. I. Mahmoud, and F. E. A. El-Samie, "Fast and robust real-time face detection system using FPGA-based general purpose fuzzy processor," *Int. J. Inf. Technol. Elect. Eng.*, vol. 2, pp. 30–38, Feb. 2013.
[37] U. Javed, M. M. Riaz, A. Ghafoor, and T. A. Cheema, "Local features and Takagi–Sugeno fuzzy logic based medical image segmentation," *Radioengineering*, vol. 22, no. 4, pp. 1091–1097, Dec. 2013.

PRANITHAN PHENSADSAENG received the B.Eng. degree in electronic engineering from Bangkok University, Bangkok, Thailand, in 2002, and the M.Eng. degree in electrical engineering from the King Mongkut's University of Technology Thonburi, Bangkok, Thailand, in 2006, Where he is currently pursuing the Ph.D. degree in electronic and telecommunication engineering.

From 2006 to 2017, he is involved in several national and international research projects in biomedical image processing. His research interests include medical imaging, image and signal processing, and FPGA architecture.

KOSIN CHAMNONGTHAI is professor with the Electronic and Telecommunication Engineering Department, Faculty of Engineering, King Mongkut's University of Technology Thonburi, and serves as the President Elect of the ECTI Association, from 2016 to 2017. He served as an Editor of the *ECTI E-magazine* from 2011 to 2015, an Associate Editor of the ECTI-CIT Transactions, from 2011 to 2016, an Associate Editor of the ECTI-EEC Transactions, from 2003 to 2010, an Associate Editor of ELEX (IEICE Trans), from 2008 to 2010, and the Chairman of the IEEE COMSOC Thailand, from 2004 to 2007.

He received the B.Eng. degree in applied electronics from the University of Electro-Communications, Tokyo, Japan, in 1985, the M.Eng. degree in electrical engineering from the Nippon Institute of Technology, Saitama, Japan, in 1987, and the Ph.D. degree in electrical engineering from Keio University, Tokyo, in 1991. His research interests include computer vision, image processing, robot vision, and signal processing. He is a member of TRS, IEICE, TESA, ECTI, AIAT, APSIPA, and EEAAT.