# A Real-Time Histogram Equalization System with Automatic Gain Control Using FPGA

**Junguk Cho, Seunghun Jin, Key Ho Kwon and Jae Wook Jeon**
School of Information and Communication Engineering, Sungkyunkwan University,
300 Cheoncheon-dong Jangan-gu, Suwon, Gyeonggi-do 440-746, Korea
[e-mail: {ichead, coredev}@ece.skku.ac.kr, {khkwon, jwjeon}@yurim.skku.ac.kr]
*Corresponding author: Jae Wook Jeon

*Received May 4, 2010; revised June 20 and June 28, 2010; accepted July 1, 2010; published August 27, 2010*

### Abstract

High quality camera images, with good contrast and intensity, are needed to obtain the desired information. Images need to be enhanced when they are dark or bright. The histogram equalization technique, which flattens the density distribution of an image, has been widely used to enhance image contrast due to its effectiveness and simplicity. This technique, however, cannot be used to enhance images that are either too dark or too bright. In addition, it is difficult to perform histogram equalization in real-time using a general-purpose computer. This paper proposes a histogram equalization technique with AGC (Automatic Gain Control) to extend the image enhancement range. It is designed using VHDL (VHSIC Hardware Description Language) to enhance images in real-time. The system is implemented with an FPGA (Field Programmable Gate Array). An image processing system with this FPGA is implemented. The performance of this image processing system is measured.

***Keywords:*** Automatic gain control, image enhancement, FPGA, histogram equalization, VHDL

**DOI: 10.3837/tiis.2010.08.0011**

## 1. Introduction

Image processing systems are widely used in robots. They normally use a light-sensitive camera. High quality camera images, with good contrast and intensity, are needed to obtain the desired information. An image that is dark or bright needs to be enhanced. Enhancing the contrast of an image is an important task in image processing. Contrast enhancement can be accomplished by stretching the dynamic range of the image. Many techniques enhance the contrast. The histogram equalization technique, which flattens the density distribution of the image is the most popular method due to its effectiveness and simplicity [1][2][3][4][5][6][7]. In particular, the histogram equalization technique is widely used in medical image processing, infrared image processing, and radar image processing [8][9][10]. However, this technique cannot enhance images that are either too dark or too bright. In addition, it is difficult to perform histogram equalization in real-time using a general-purpose computer [8][11].

Kim et al. [3] proposed a block-overlapped histogram equalization system to enhance the contrast of an image sequence. Their system was implanted to target video camcorders. They did not show and measure their results and performance. Wang and Ye [4] presented a novel case of histogram specification, which can preserve the mean brightness with maximum entropy, in a continuous view. They did not implement a system using their algorithm. In addition, they did not mention real-time processing. Pichon et al. [5] proposed an extension of grayscale histogram equalization to color images. Their method is based on deforming a mesh in color space to fit the existing histogram and then map it to a uniform histogram. They did not mention real-time system implementation. Jin et al. [6] presented a multi-scale adaptive histogram equalization method. This showed promising results on chest CT interpretation. They did not show a system implementation. Kim et al. [7] presented a contrast enhancement algorithm derived from local histogram equalization. They did not measure performance. Almost all of the literature on histogram equalization is theoretical. Little effort has been made to design and implement a hardware system for real-time histogram equalization.

This paper proposes a histogram equalization technique with AGC (Automatic Gain Control) to extend the image enhancement range. AGC stretches the dynamic range of the image to the whole gray level range [12]. Therefore, the performance of the proposed histogram equalization technique with AGC is better than that of the corresponding technique without AGC. This paper designs a system for the proposed histogram equalization technique with AGC using VHDL (VHSIC Hardware Description Language) to enhance images in real-time. It implements this with an FPGA (Field Programmable Gate Array). The image processing system with the proposed histogram equalization with AGC improves performance 11.75 and 129.41 times over the software programs measured from the PC (Intel Pentium 4) and the embedded system (Intel PXA270 CPU), respectively.

Section 2 explains histogram equalization. An AGC technique based on the histogram is proposed to adjust image brightness. The histogram equalization technique with AGC is proposed and designed using VHDL in section 3. In section 4, the system is implemented with an FPGA and its performance is measured. Finally, our conclusions are presented in section 5.

## 2. Histogram Equalization

### 2.1 Conventional Histogram Equalization

An 8-bit gray image consists of 256 gray levels. The darkest pixel has the gray level of 0 and the brightest one 255. The histogram represents the number of pixels on the y-axis at each gray level on the x-axis in one image frame. The cumulative histogram represents the number of pixels on the y-axis in the interval between the gray level 0 and each gray level on the x-axis. The histogram and cumulative histogram of the image with 640×480 pixels, as shown in **Fig. 1**, are shown in **Figs. 2** and **3**, respectively.

The transformation function, $T(k)$, for the purpose of histogram equalization, using the cumulative histogram, is written as follows

$$T(k) = \sum_{j=0}^{k} p(j) = \sum_{j=0}^{k} \frac{n_j}{n}, \qquad k = 0, 1, 2, \cdots, 255 \tag{1}$$

where $p(j)=n_j/n$ is the probability density function at the gray level $j$, $n_j$ is the number of pixels at the gray level $j$, and n is the total number of pixels in one image frame. **Fig. 4** shows this transformation function for the image shown in **Fig. 1**.

Histogram equalization is performed using $T(k)$: Given one pixel of gray level k, its equalized result is $255 \times T(k)$. That is, one gray level of a pixel is transformed into another gray level [7].

**Fig. 1**. The image temple.

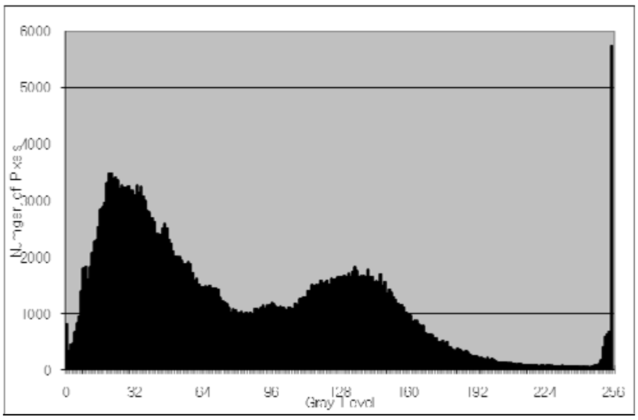



**Fig. 2**. Histogram of the image shown in **Fig.1**.

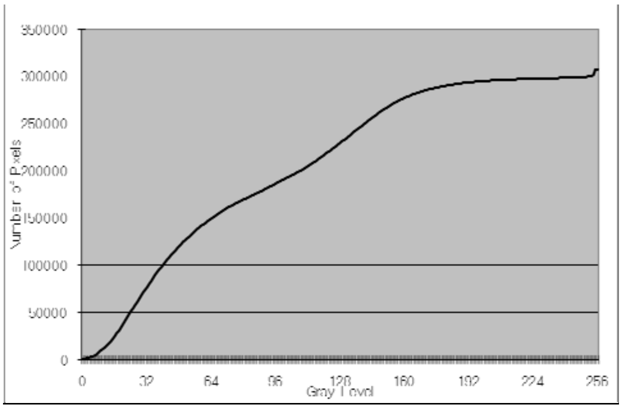



**Fig. 3**. Cumulative histogram of the image shown in **Fig. 1**.

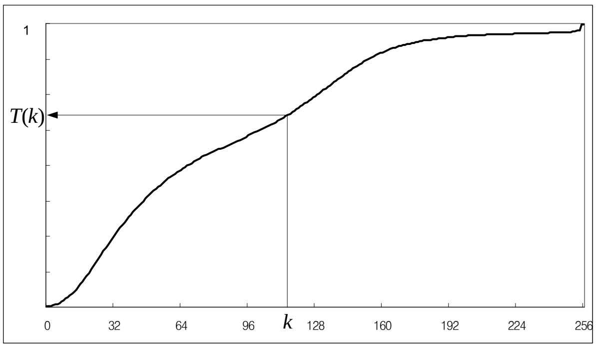



**Fig. 4**. Transformation function for histogram equalization.

## 2.2 Quantized Histogram Equalization

256 counters of 19 bits are needed If we design histogram equalization for an 8-bit gray image with $640\times480$ pixels using the transformation function described in (1), because one counter is necessary to count the number of pixels at each gray level and $2^{18} < 640\times480 < 2^{19}$.

The quantized histogram can be used to reduce the number of counters [8][11]. The quantized histogram of one image frame represents the number of pixels in each gray interval rather than at each gray level. **Fig. 5** shows the quantized histogram of the image shown in **Fig. 1** that is divided into eight intervals. Hereafter, we use the terminology, histogram, to denote the quantized histogram divided into eight intervals.

The transformation function, $T(k)$, for quantized histogram equalization is written as follows

$$T(k) = T(32(i-1)) + \frac{(T(32i) - T(32(i-1))}{(32i - 32(i-1))} \cdot (k - 32(i-1)) \tag{2}$$

for $32(i-1) \le k < 32i$, $k = 0,1,2,\ldots,255$, and $i = 1,2,\ldots,8$. **Fig. 6** shows this transformation function. Histogram equalization can be performed using $T(k)$: Given one pixel of gray level $k$, its equalized result is $255\times T(k)$.

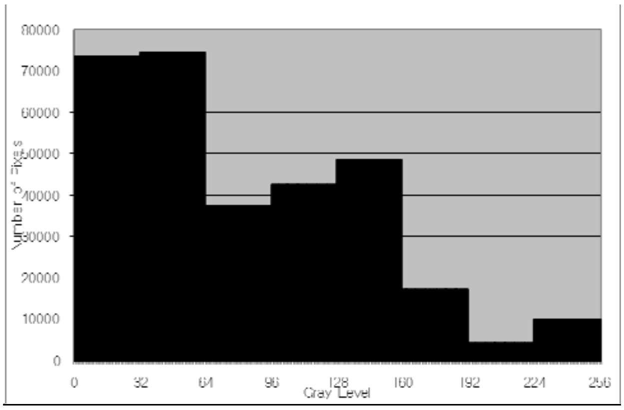



**Fig. 5**. Quantized histogram of the image shown in **Fig. 1** divided into eight intervals.

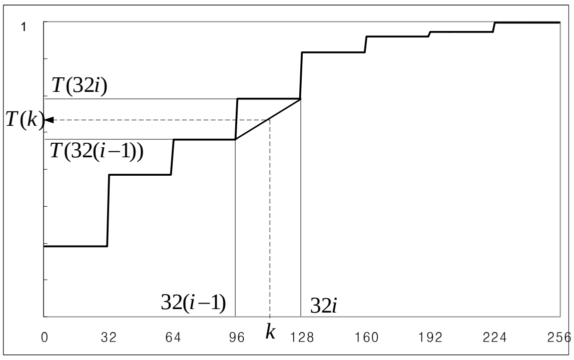



**Fig. 6**. Transformation function for quantized histogram equalization.

## 3. Histogram Equalization with Automatic Gain Control

### 3.1 Proposed Automatic Gain Control

Previous AGC techniques attempt to adjust the image sensor analog output value. These techniques have been used to build cameras that can produce high quality images. In contrast to these previous techniques, this paper proposes an AGC technique that does not adjust the sensor output value, but rather modifies the brightness of the image using its histogram. The image histograms of adjacent frames in a video sequence are similar to each other. Therefore current gain control values are expected to be very similar to the value obtained from the next image frame.

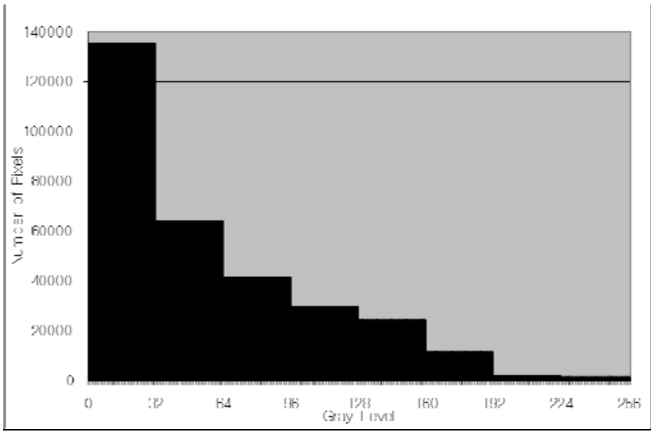
The proposed AGC can make dark images brighter or bright images darker as follows: The overall brightness of one image is determined by its histogram and then an appropriate gain is selected. Each pixel gray level in the image is then multiplied by this gain, thereby increasing or decreasing its intensity, resulting in a brighter or darker image.

The purpose of the automatic gain control algorithm is to stretch the dynamic range of the image to the whole gray level range. It uses the histogram data for each image to detect which images should be considered too dark or too bright. An appropriate gain factor is chosen whenever such an image is detected. This gain factor is simply multiplied with each pixel in the image, increasing or decreasing its intensity and resulting in a brighter or darker image. The gain should be as high or low as possible without losing image contrast. When the gain factor is too high, too many pixels become white, resulting in a loss of image data. When the gain factor is too low, too may pixels become black, resulting in a loss of image data. **Fig. 7**, showing three pictures of the same face, illustrates this. **Fig. 7-(b)** seems to have an appropriate brightness, while **Fig. 7-(a)** and **Fig. 7-(b)** have been amplified. The darkest, **Fig. 7-(a)**, is obviously too dark. Parts of the facial hair are no longer visible due to the low gain, where too many pixels have become black. The brightest, **Fig. 7-(c)**, is too bright. Parts of the skin are no longer visible due to the high gain, where too many pixels have become white. As shown in **Fig. 7**, the histogram of **Fig. 7-(e)** has a wide dynamic range of the image to the whole gray level range, while the histograms of **Fig. 7-(d)** and **Fig. 7-(f)** have a dynamic range of the images to too much dark gray level and too much bright gray level, respectively.

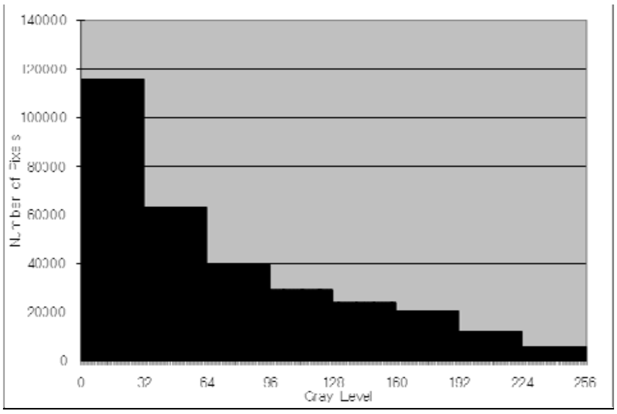
(a) Too dark image

(b) Appropriate brightness image

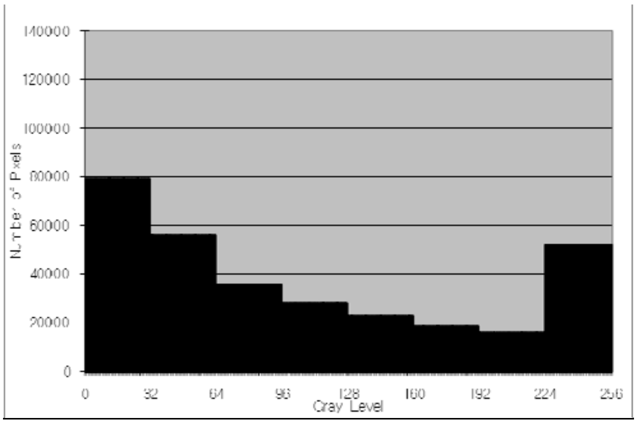
(c) Too bright image




(d) Histogram of too dark image




(e) Histogram of appropriate brightness image




(f) Histogram of too bright image

**Fig. 7**. Constrasting image's faces and their histograms.

A large number of images have been studied in Matlab to develop an algorithm for automatic gain control. Not all images need amplification. This requires a way to determine histogram patterns for those images that need amplification. Other reasons to study many images is to examine the extent to which an image needs amplification and which gain factor to use.

A dark or bright image can be recognized by studying its histogram. The quantized histograms used to describe the images consist of eight columns, where each column represents a number between 0 and 255. The algorithm becomes very complex when looking into all eight columns to decide whether an image needs to be amplified. To make the algorithm as simple as possible, the gain factor is chosen only based on the number of pixels in the histogram columns representing the darkest and brightest pixels. A high number of pixels in these columns would result in a high or low gain factor. *Amp_Col*($t$,1) represents the darkest column and *Amp_Col*($t$,8) represents the brightest column of the amplified image, respectively.

A test on the amplified image needs to be applied to achieve a gain factor suitable for the vast majority of the images. It is important to apply the right gain factor, not making it too high or too low; otherwise, data may be lost, if too many pixels become white or black. An amplified image can be compared with itself before it is amplified to prevent this data loss. Doing so, the increased or decreased brightness can be measured in equation (4) and (6).

**Fig. 8** shows how the gain is determined for each image. The first gain value for the first image is set to 1 and the following gain values are determined using **Table 1** and (3)-(6).

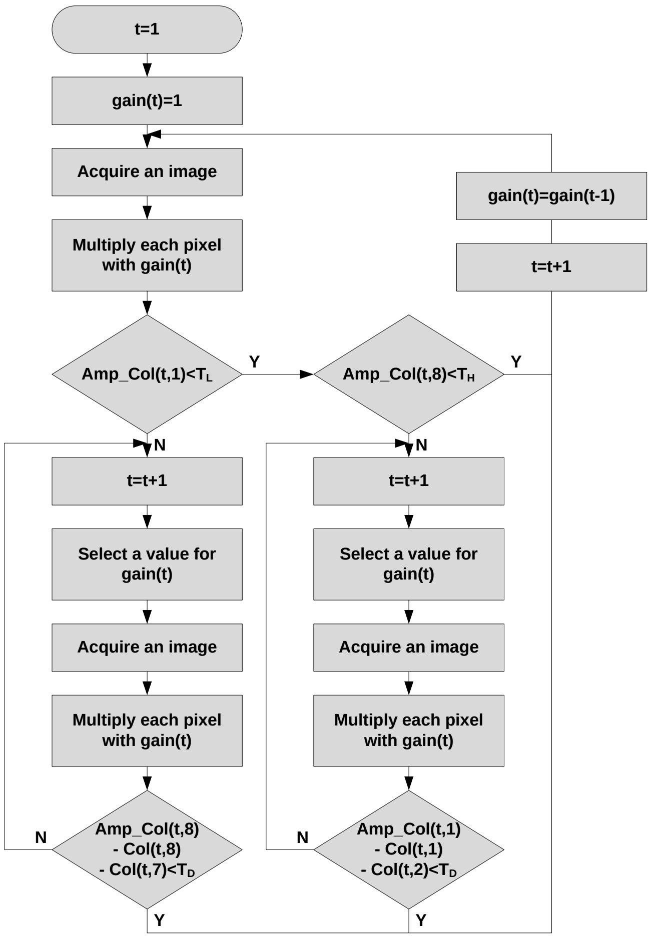



**Fig. 8**. Proposed automatic gain control.

**Table 1** has been attained from studies of a large number of images. **Table 1** shows the gain values of the ranges of each interval of *Amp_Col*($t$,1) representing the darkest column and *Amp_Col*($t$,8) representing the brightest column of the amplified image. The gain values are increased and decreased exponentially based on the number of pixels of *Amp_Col*($t$,1) and *Amp_Col*($t$,8), respectively. First, we made the ranges uniform for each interval from 25% to 100% of both *Amp_Col*($t$,1) and *Amp_Col*($t$,8). Then we adjusted the range of each interval by studying a large number of images.

$$Amp\_Col(t,1) < T_L \tag{3}$$

$$Amp\_Col(t,8) - Col(t,8) - Col(t,7) < T_D \tag{4}$$

$$Amp\_Col(t,8) < T_H \tag{5}$$

$$Amp\_Col(t,1) - Col(t,1) - Col(t,2) < T_D \tag{6}$$

**Table 1**. Lookup Table for the Gain Value.

| ***Amp_Col*(*t*,1): Number of Pixels (%)** | **Gain** |
|---|---|
| 76,800 (25) ≤ *Amp_Col*($t$,1) < 82,944 (27) | 1.25 |
| 82,944 (27) ≤ *Amp_Col*($t$,1) < 95,232 (31) | 1.55 |
| 95,232 (31) ≤ *Amp_Col*($t$,1) < 119,808 (39) | 1.93 |
| 119,808 (39) ≤ *Amp_Col*($t$,1) < 144,384 (47) | 2.41 |
| 144,384 (47) ≤ *Amp_Col*($t$,1) < 162,816 (53) | 3.00 |
| 162,816 (53) ≤ *Amp_Col*($t$,1) < 178,176 (58) | 3.74 |
| 178,176 (58) ≤ *Amp_Col*($t$,1) < 211,968 (69) | 4.66 |
| 211,968 (69) ≤ *Amp_Col*($t$,1) < 239,616 (78) | 5.88 |
| 239,616 (78) ≤ *Amp_Col*($t$,1) < 288,768 (94) | 7.22 |
| 288,768 (94) ≤ *Amp_Col*($t$,1) | 9.00 |
| ***Amp_Col*(*t*,8): Number of Pixels (%)** | **Gain** |
| 76,800 (25) ≤ *Amp_Col*($t$,8) < 82,944 (27) | 0.898 |
| 82,944 (27) ≤ *Amp_Col*($t$,8) < 95,232 (31) | 0.792 |
| 95,232 (31) ≤ *Amp_Col*($t$,8) < 119,808 (39) | 0.699 |
| 119,808 (39) ≤ *Amp_Col*($t$,8) < 144,384 (47) | 0.597 |
| 144,384 (47) ≤ *Amp_Col*($t$,8) < 162,816 (53) | 0.500 |
| 162,816 (53) ≤ *Amp_Col*($t$,8) < 178,176 (58) | 0.398 |
| 178,176 (58) ≤ *Amp_Col*($t$,8) < 211,968 (69) | 0.296 |
| 211,968 (69) ≤ *Amp_Col*($t$,8) < 239,616 (78) | 0.199 |
| 239,616 (78) ≤ *Amp_Col*($t$,8) < 288,768 (94) | 0.097 |
| 288,768 (94) ≤ *Amp_Col*($t$,8) | 0.046 |

For $1 \leq i \leq 8$, $Col(t,i)$ represents the number of pixels in the *i-th* interval of the *t-th* image histogram. That is, $Col(t,8)$ and Col($t$,7) represent the numbers of pixels in the first and second brightest intervals in the *t-th* image histogram, respectively. Likewise, $Col(t,1)$ and $Col(t,2)$ represent the numbers of pixels in the first and second darkest intervals in the *t-th* image histogram, respectively.

For $1 \leq i \leq 8$, $Amp\_Col(t,i)$ represents the number of pixels in the *i-th* interval in the histogram of the resultant image obtained by multiplying each pixel in the *t-th* image by the *t-th* gain. That is, $Amp\_Col(t,1)$ and $Amp\_Col(t,8)$ represent the numbers of pixels in the darkest interval and the brightest interval in the histogram of the resultant image obtained by multiplying each pixel in the t-th image by the *t-th* gain, respectively.

In the present case, both threshold values, $T_H$ and $T_L$, are 76,800, represent 25% of the total number of pixels in an image of 640×480 pixels. The threshold value, $T_D$, is 18,432 in an image of 640×480 pixels, representing 6% of the total number of pixels [13]. The value of $T_H$, $T_L$, and $T_D$ should be increased or decreased based on the size of the image to be enhanced while the percentage of $T_H$, $T_L$, and $T_D$ should be maintained. First, we obtained a threshold value from [13]. Then we studied a large number of images to find an appropriate threshold value in our algorithm. $T_H$, $T_L$, and $T_D$ have been attained from studies of a large number of images.

If (3) is satisfied, the resultant image obtained by multiplying each pixel in the *t-th* image by the *t-th* gain is not too dark, because $Amp\_Col(t,1)$ represents the darkest column of the amplified image. Therefore, if $Amp\_Col(t,1)$ is equal to or above $T_L$, the decreased brightness is considered too low and the gain factor is too low. If (3) is satisfied, then (5) is checked. If (5) is satisfied, the resultant image obtained by multiplying each pixel in the *t-th* image by the *t-th* gain is not too bright either, because $Amp\_Col(t,8)$ represents the brightest column of the amplified image. Therefore, if $Amp\_Col(t,8)$ is equal to or above $T_H$, the increased brightness is considered too high and the gain factor is too high. Therefore, the *t-th* gain value does not need to be changed and is used as the ($t$+1)-*th* gain. That is, each pixel in the ($t$+1)-*th* image is multiplied by the same gain value and then the histogram of the resultant image is checked again using equation (3).

If (5) is not satisfied after (3) is satisfied, then the resultant image obtained by multiplying each pixel in the *t-th* image by the *t-th* gain is too bright, because $Amp\_Col(t,8)$ represents the brightest column of the amplified image. Therefore, if $Amp\_Col(t,8)$ is equal to or above $T_H$, the increased brightness is considered too high and the gain factor is too high. That is, the *t-th* gain value is too large and, consequently, a new value for the ($t$+1)-*th* gain is determined, as shown in **Table 1**. Each pixel in the ($t$+1)-*th* image is multiplied by this new gain value and then the histogram of the resultant image is checked using (6). The left-hand side of (6) represents the increase in the number of pixels in the darkest interval of the resultant image histogram. If (6) is satisfied, then the ($t$+1)-*th* gain value is appropriate and can be used as the ($t$+2)-*th* gain value because $Amp\_Col(t,1)$-$Col(t,1)$-Col($t$,2) represents the increase in the darkest column of the amplified image. Otherwise, the ($t$+1)-*th* gain value is too small and therefore the higher gain value adjacent to the ($t$+1)-*th* gain value in **Table 1** is selected. This value is used as the ($t$+2)-*th* gain value. Each pixel in the ($t$+2)-*th* image is multiplied by the ($t$+2)-*th* gain and then the histogram of the resultant image is checked again using (6).

If (3) is not satisfied, then the resultant image obtained by multiplying each pixel in the *t-th* image by the *t-th* gain is too dark, because $Amp\_Col(t,1)$ represents the darkest column of the amplified image. Therefore, if $Amp\_Col(t,1)$ is greater than or equal to $T_L$, the increased brightness is considered too low and the gain factor is too low. That is, the *t-th* gain is too low and a new value for the ($t$+1)-*th* gain is determined, as shown in **Table 1**. Each pixel in the ($t$+1)-*th* image is multiplied by the new gain value and then the histogram of the resultant image is checked using (4). The left-hand side of (4) represents the increase in the number of the pixels in the brightest interval of the resultant image histogram. If (4) is satisfied, then the ($t$+1)-*th* gain value is appropriate and can be used as the ($t$+2)-*th* gain value, because $Amp\_Col(t,8)$-$Col(t,8)$-$Col(t,7)$ represents the increase in the brightest column of the amplified image. Otherwise, the ($t$+1)-*th* gain value is too large and therefore the lower gain value adjacent to the ($t$+1)-*th* gain value in **Table 1** is selected. This value is used as the ($t$+2)-*th* gain value. Each pixel in the ($t$+2)-*th* image is multiplied by the ($t$+2)-*th* gain. Then, the histogram of the resultant image is again checked using equation (4). This process will maintain the run status while the system is on, and is terminated when the system is off.

### 3.2 Proposed Histogram Equalization with Automatic Gain Control

The histogram equalization technique cannot enhance images that are either too dark or too bright. Therefore, this paper proposes a histogram equalization technique with AGC to extend the image enhancement range. In this case, an image is acquired from a camera, its histogram is generated, and an appropriate gain is selected, as described in section 3. Each pixel in the image is multiplied by this gain value and the resultant histogram is again generated. Image enhancement is performed using the resultant histogram, as described in section 2.

The performance of the proposed histogram equalization technique with AGC is better than that of the histogram equalization technique without AGC [14], since AGC stretches the dynamic range of the image to the whole gray level range. Histogram equalization stretches the dynamic range of the image to enhance the image contrast. The distributive property of the histogram presents the distribution of the pixels at the gray level. Therefore, it shows a specific property of the image. If the distributive property of the histogram is very different to one of the original image after histogram equalization, it will cause problems in the other image processing using the proposed image. Therefore, conserving the distributive property after enhancement processing is very important. **Fig. 9** compares the results of the histogram equalizations without or with AGC for one image. **Fig. 9-(a)** is a raw image obtained from a camera and **Fig. 9-(d)** is its histogram. **Fig. 9-(b)** is the result of histogram equalization without AGC and **Fig. 9-(e)** is its histogram. Histogram equalization with AGC and its histogram are shown in **Fig. 9-(c)** and **Fig. 9-(f)**, respectively. In **Fig. 9-(e)**, the distributive property of the histogram is very different to the one of the original image **Fig 9-(d)**. However, the distributive property of the histogram of **Fig. 9-(f)** is very similar to the one of the original image **Fig. 9-(d)**. In summary, the distributive property of the histogram is changed after histogram equalization. However, it is maintained after histogram equalization with AGC.

It is difficult to perform histogram equalization in real-time using a general-purpose computer [8][11]. This paper designed a system for the proposed histogram equalization technique with AGC using VHDL and implemented it with FPGA to enhance images in real-time. **Fig. 10** shows the block diagram of the proposed histogram equalization with AGC. It consists of several modules: a camera controller, histogram generators, an automatic gain controller, and an equalizer.

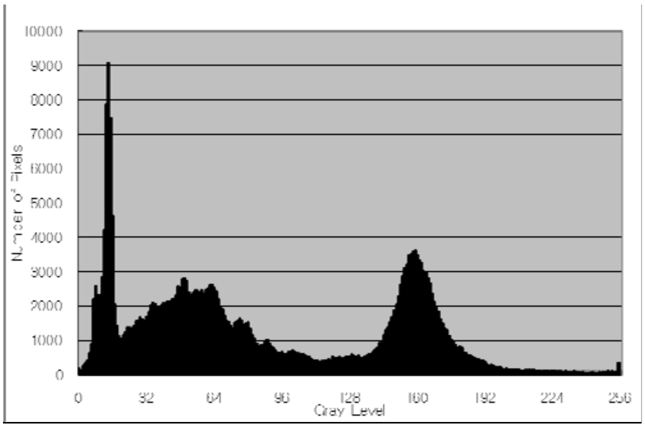
In the camera controller, the sync separator generates the signals required to control the camera and the A/D converter converts the analog image data to digital form.

Each histogram generator has eight counters of 19 bits. Each 19-bit counter calculates the number of pixels in one interval, as shown in **Fig. 5**. That is, the value of each 19-bit counter represents the height of one column in **Fig. 5**. After counting the number of pixels in each interval, the histogram lookup table stores the histogram. One histogram generator calculates the histogram of a raw image obtained from the camera. This histogram is used in the automatic gain controller. The other histogram generator calculates the histogram of the resultant image obtained by multiplying pixels of a raw image by a gain. This histogram is used in the equalizer and automatic gain controller.

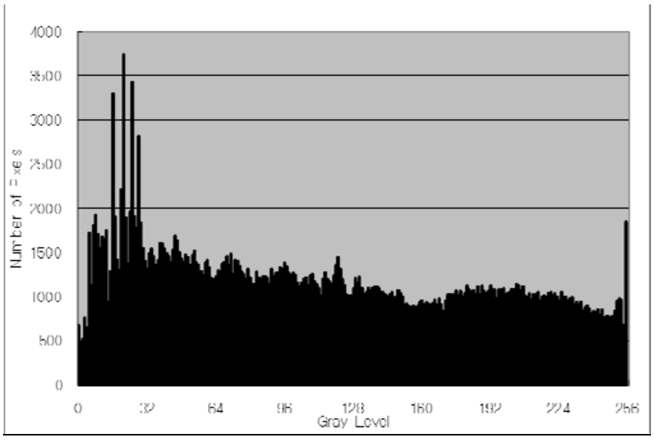
(a) Original image

(b) Image after histogram equalization without AGC

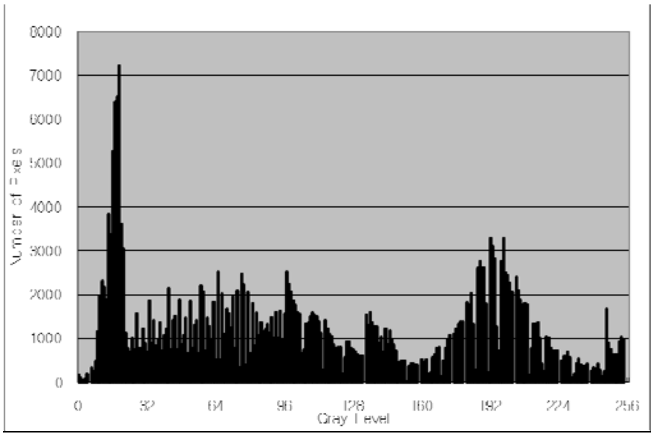
(c) Image after histogram equalization with AGC




(d) Histogram of original image




(e) Histogram of image after histogram equalization without AGC




(f) Histogram of image after histogram equalization with AGC

**Fig. 9**. Original image, result of histogram equalization without AGC, and result of histogram equalization with AGC.

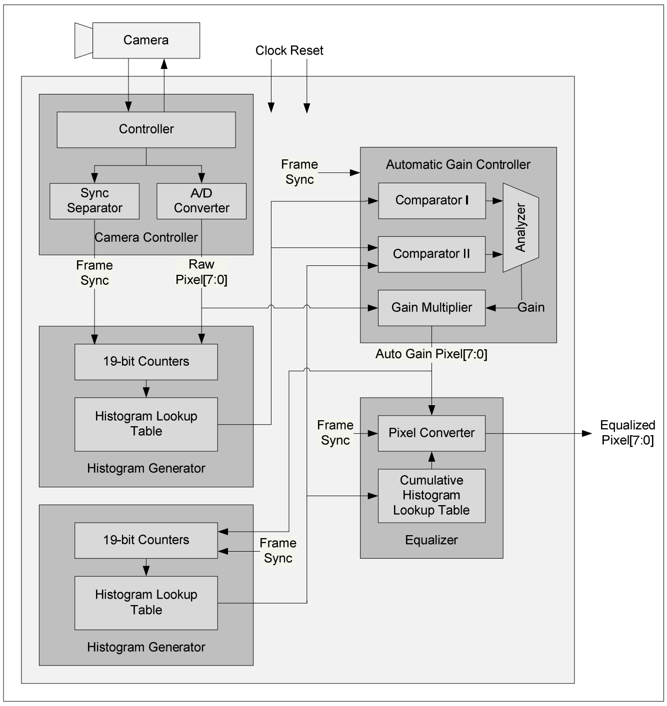



**Fig. 10**. Block diagram of the proposed histogram equalization with AGC.

In the automatic gain controller, the first comparator performs the operations corresponding to (3) and (5), while the second comparator performs the operations corresponding to (4) and (6). The analyzer selects an appropriate gain value, as shown in **Table 1**, according to the results obtained from the two comparators. The gain multiplier multiplies each pixel by this gain value and sends its result to the histogram generator and the equalizer.

The cumulative histogram for the image, which is obtained from the automatic gain controller, is generated in the equalizer. Its information is stored in the cumulative histogram lookup table. The pixel converter transforms the gray level of each input pixel into another gray level using the cumulative histogram lookup table, as shown in **Fig. 6**.

## 4. Implementation / Experiment

This paper designed a system for the proposed histogram equalization with AGC using VHDL. It was implemented with a Xilinx FPGA, XC2V6000-4CFF1152 that has 6M system gates, 76,032 logic cells, 3,024 Kbits Block SelectRAM, 168 18×18 Multipliers, and 1,104 I/O pins [15]. **Table 2** summarizes device utilization for the proposed histogram equalization with AGC, where each slice includes two 4-input function generators, carry logic, arithmetic logic gates, wide function multiplexers and two storage elements. **Fig. 11** shows an image processing system with an XC2V6000 used to implement the proposed system. This system can acquire images from an RS-170 camera and send both raw and processed images to a PC through the PCI (Peripheral Component Interconnect) bus.

A high frame processing rate and low latency are important for many applications that must provide quick decisions based on events that occur in a given scene [16]. When the image processing system with the proposed histogram equalization with AGC is applied to images containing 640×480 pixels obtained from an RS-170 camera, it can operate at up to 122.3 fps (frames per second) and send enhanced images to the PC. Histogram equalization without AGC can operate at up to 268.4 fps and send enhanced images to the PC. The performance of these systems is determined from the maximum delay time after synthesis.

**Table 2**. Device Utilization Characteristics for Histogram Equalization with AGC.

| Device Utilization Summary | | | | |
|---|---|---|---|---|
| Number of Slices: | 984 | out of | 33792 | 2% |
| Number of Slice Flip Flops: | 999 | out of | 67584 | 1% |
| Number of 4 input LUTs: | 1741 | out of | 67584 | 2% |
| Number of bonded IOBs: | 104 | out of | 824 | 12% |
| Number of MULT18X18s: | 18 | out of | 144 | 12% |
| Number of GCLKs: | 8 | out of | 16 | 50% |

The performance of these software programs was measured from the average time of 500 repeated operations in a PC: Intel Pentium 4 CPU (2.4 GHz), 1 GB DDR SDRAM PC2100 (266 MHz), Microsoft Windows XP professional, and Microsoft Visual Studio. These software programs were developed using Microsoft Visual C++. If histogram equalization without AGC is performed on the same camera image by a software program, it can operate at up to 149.7 fps. If histogram equalization with AGC is performed by a software program, it can operate at up to 10.4 fps.

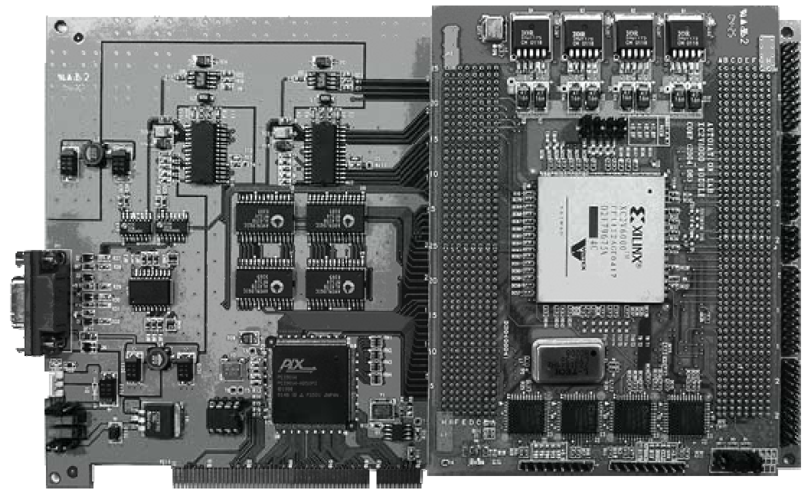


**Fig. 11**. An image processing system for the proposed histogram equalization with AGC.

The performance of these software programs was measured from the average time of 500 repeated operations in an embedded system: Intel PXA270 CPU (530 MHz), 128 MB SDRAM (133 MHz), Microsoft Windows CE 5.0, and Platform Builder for Microsoft Windows CE 5.0. These software programs were developed using Microsoft Embedded Visual C++. If histogram equalization without AGC is performed on the same camera image by a software program, it can operate at up to 31.5 fps. If histogram equalization with AGC is performed by a software program, it can operate at up to 0.945 fps. **Table 3** compares performance of histogram equalization with AGC systems.

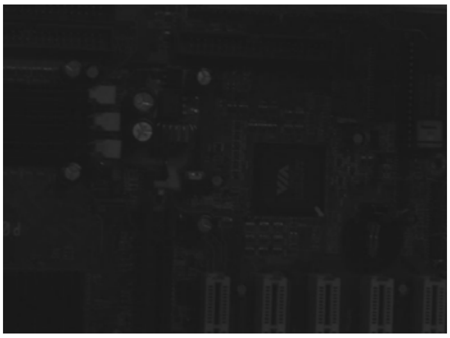
**Table 3**. Performance of Histogram Equalization with AGC System.

| **Proposed System, Virtex-II, VGA image** | **Performance** |
|---|---|
| System for histogram equalization without AGC | 268.4 fps |
| System for histogram equalization with AGC | 122.3 fps |
| **Pentium 4 CPU 2.4GHz, 1GMB DDRAM, VGA image** | **Performance** |
| Software program of histogram equalization without AGC | 149.7 fps |
| Software program of histogram equalization with AGC | 10.4 fps |
| **PXA270 CPU 520MHz, 128MB SDRAM, VGA image** | **Performance** |
| Software program of histogram equalization without AGC | 31.5 fps |
| Software program of histogram equalization with AGC | 0.945 fps |

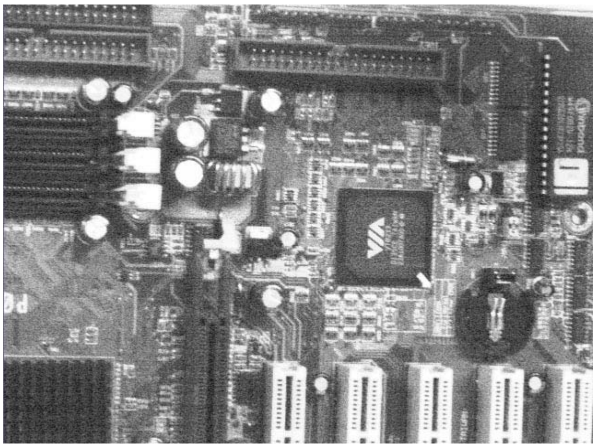
The image processing system with the proposed histogram equalization without AGC improves performance by 1.79 and 8.64 times over the software programs measured from the PC (Intel Pentium 4) and the embedded system (Intel PXA270 CPU), respectively. The image processing system, with the proposed histogram equalization with AGC, improves performance by 11.75 and 129.41 times over the software programs measured from the PC (Intel Pentium 4) and the embedded system (Intel PXA270 CPU), respectively.

The above software program in a PC for histogram equalization with AGC cannot process the images in real-time, since the image frame rate of an RS-170 camera is 30fps. In contrast, the image processing system with the proposed histogram equalization with AGC can process images from an RS-170 camera in real time.

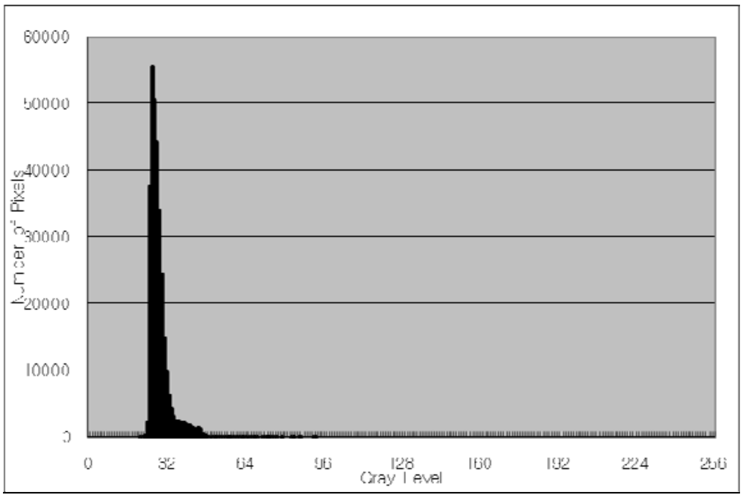
**Fig. 12** shows the results of histogram equalization with AGC for one image. **Fig. 12-(a)** is the raw image obtained from a camera under low illumination and **Fig. 12-(c)** is its histogram. **Fig. 12-(b)** is the result of histogram equalization with AGC and **Fig. 12-(d)** is its histogram. **Fig. 13** shows the results of histogram equalization with AGC for one image. **Fig. 13-(a)** is the raw image obtained from a camera under high illumination and **Fig. 13-(c)** is its histogram. **Fig. 13-(b)** is the result of histogram equalization with AGC and **Fig. 12-(d)** is its histogram. In these results, histogram equalization with AGC appropriately enhances images that are too dark or too bright.



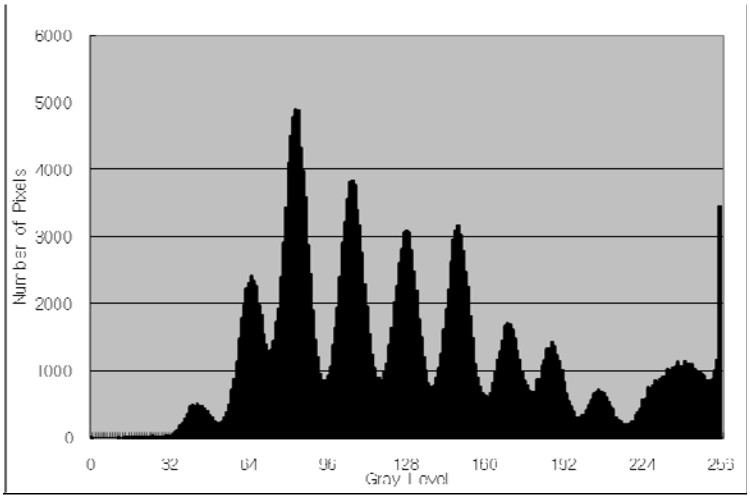
(a) Original image under low illumination



(b) Image after histogram equalization with AGC




(c) Histogram of original image under low illumination

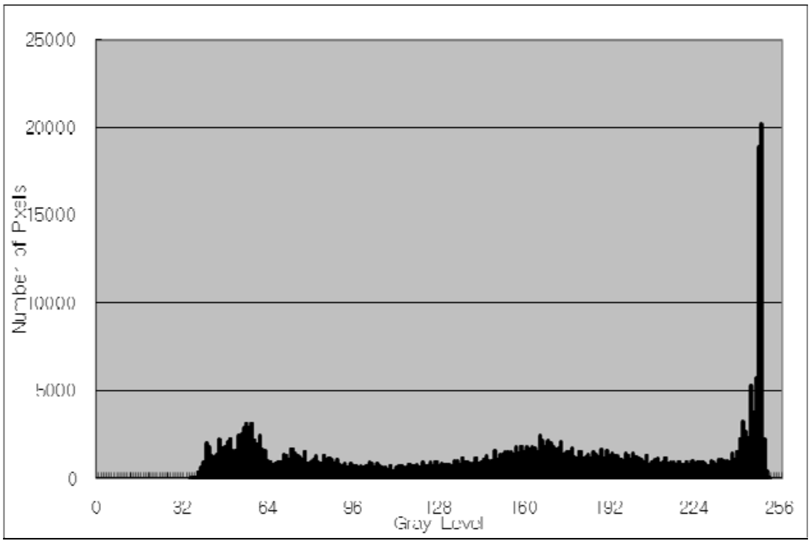



(d) Histogram of Image after histogram equalization with AGC

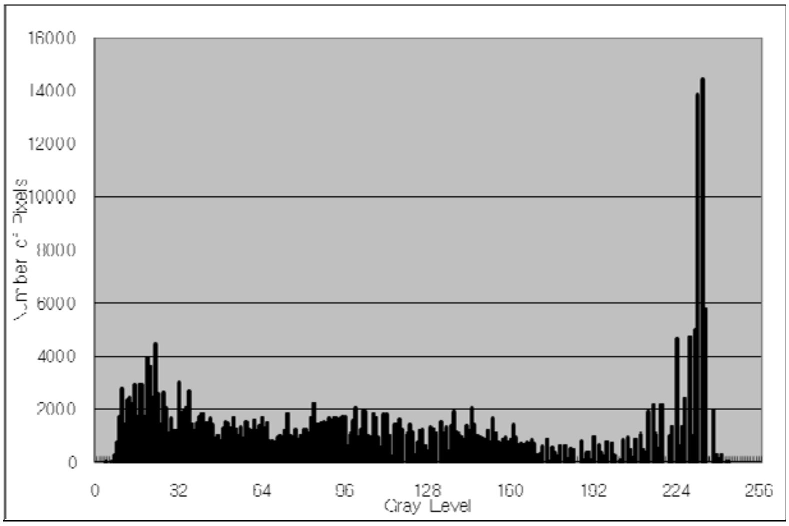
**Fig. 12**. Original image under low illumination, and result of histogram equalization with AGC.


(a) Original image under high illumination


(b) Image after histogram equalization with AGC




(c) Histogram of original image under high illumination



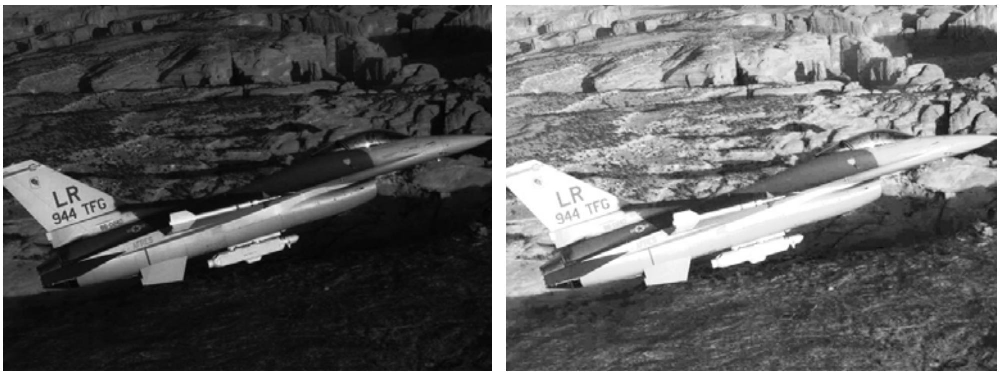

(d) Histogram of image after histogram equalization with AGC

**Fig. 13**. Original image under high illumination, and result of histogram equalization with AGC.

**Fig. 14** shows several resultant images obtained from the proposed histogram equalization with AGC and their original images.

(a) Image Temple

(b) Image Statues



(c) Image Airplane

**Fig. 14**. Original images, and results of histogram equalization with AGC.

## 5. Conclusion

This paper proposed a histogram equalization technique with AGC to extend the image enhancement range. The proposed histogram equalization technique with AGC performs better than the previous histogram equalization technique without AGC, since AGC can extend the dynamic range of an image to the whole gray level range. This paper designed a system for the proposed histogram equalization technique with AGC using VHDL. It implemented it with FPGA. This image processing system achieves faster equalization than the software implementation running on a general purpose PC and a general purpose embedded system. The designed system of the proposed histogram equalization technique with AGC can be applied to many high-level image processing tasks to reduce both cost and processing time.

## Acknowledgement

This research was performed for the Intelligent Robotics Development Program, one of the 21st Century Frontier R&D Programs funded by the Ministry of Commerce, Industry and Energy of Korea.

## References

[1] R. C. Gonzalez and R. E. Woods, "Digital Image Processing, 2nd ed. Reading," *MA: Addison-Wesley*, 1992.
[2] A. K. Jain, "Fundamentals of Digital Image Processing, Englewood Cliffs,' *NJ: Prentice-Hall*, 1989.
[3] T. K. Kim, J. K. Paik, and B. S. Kang," Contrast Enhancement System Using Spatially Adaptive Histogram Equalization with Temporal Filtering," *IEEE Trans. on Consumer Electronics*, vol. 44, no. 1, pp. 82-86, February 1998.
[4] C. Wang and Z. Ye, "Brightness Preserving Histogram Equalization with Maximum Entropy: A Variational Perspective," *IEEE Trans. On Consumer Electronics*, vol. 51, no. 4, pp. 1326-1334, November 2005.
[5] E. Pichon, M. Niethammer, and G. Sapiro, "Color Histogram Equalization through Mesh Deformation," in *Proc. of Conf. on Image Processing*, pp. 117-120, 2003.
[6] Y. Jin, L. M. Fayad, and A. F. Laine, "Contrast Enhancement by Multi-scale Adaptive Histogram Equalization," in *proc. of conf. on Signal Processing and Image Processing*, vol. 4478, pp.

206-213, 2001.
[7] J.-Y. Kim, L.-S. Kim and S.-H. Hwang, “An Advanced Contrast Enhancement Using Partially Overlapped Sub-Block Histogram Equalization,” *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 11, no. 4, pp. 475-484, April 2001.
[8] J. B. Zimmerman, S. M. Pizer, E. V. Staab, J. R. Perry, W. Mccartney, and B. C. Brenton, “An Evaluation of the Effectiveness of Adaptive Histogram Equalization for Contrast Enhancement,” *IEEE Trans. on Medical Imaging*, vol. 7, no. 4, pp. 304-312, December 1988.
[9] Y.-T. Kim, “Contrast Enhancement Using Brightness Preserving Bi-Histogram Equalization,” *IEEE Trans. on Consumer Electronics*, vol. 43, no. 1, pp. 1-8, February 1997.
[10] F. Hilger and H. Ney, “Quantile Based Histogram Equalization for Noise Robust Speech Recognition,” *in proc. of conf. on Speech Communication and Technology*, pp. 1135-1138, 2001.
[11] Y.-T. Kim, “Quantized Bi-Histogram Equalization,” in *Proc. of Conf. on Acoustics, Speech, and Signal Processing*, vol. 4, pp. 2797-2800, April 1997.
[12] R. Klette and P. Zamperoni, “Handbook of Image Processing Operators,” *NY: John Wiley & Sons*, 1996.
[13] K. R. Fowler, “Automatic Gain Control for Image-Intensified Camera,” *IEEE Trans. on Instrumentation and Measurement*, vol. 53, pp. 1057-1064, August 2004.
[14] E. J. Tacconi and C. F. Christiansen, “A Wide Range and High Speed Automatic Gain Control,” in *Proc. of Particle Accelerator Conf.*, vol. 3, pp. 2139-2141, May 1993.
[15] Xilinx, Inc., “Virtex-II Platform FPGAs: Complete Data Sheet,” available from www.xilinx.com, March 2005.
[16] J. I. Woodfill, G. Gordon, and R. Buck, “Tyzx Deepsea High Speed Stereo Vision System,” in *Proc. of Conf. on Computer Vision and Pattern Recognition*, pp. 41-45, June 2004.

**Junguk Cho** received the B.S., M.S., and Ph.D. degrees in the School of Information and Communication Engineering from Sungkyunkwan University, Suwon, Korea, in 2001, 2003, and 2006, respectively. From 2006 to 2007, he was a Research Instructor in the School of Information and Communication Engineering, Sungkyunkwan University. From 2008 to 2010, he was a Postdoctoral Scholar in the Department of Computer Science and Engineering, University of California, San Diego, La Jolla, CA. In 2010, he joined the Samsung Advanced Institute of Technology, Yongin, Korea as a Senior Researcher. His research interests include embedded systems, image processing, motion control, and system on a chip.

**Seunghun Jin** received the B.S., M.S., and Ph.D. degrees in the School of Information and Communication Engineering from Sungkyunkwan University, Suwon, Korea, in 2005, 2006, and 2009, respectively. In 2009, he joined the School of Information and Communication Engineering, Sungkyunkwan University as a Research Instructor. His research interests include image and speech signal processing, embedded systems, and real-time applications.

**Key Ho Kwon** received a B.S. degree in the Department of Electronics Engineering from Seoul National University, Korea, in 1975, a M.S. degree from Department of Electronics Engineering, Seoul National University, Korea, in 1978, and a Ph.D. degree in the Department of Electronics Engineering for Seoul National University, Korea, in 1989. He is currently the Professor at the School of Information and Communication Engineering, Sungkyunkwan University. His research interests include Artificial Intelligence, Fuzzy theory, Neural network, and Genetic evolutionary algorithm.

**Jae Wook Jeon** received the B.S. and M.S. degrees in the Department of Electronics Engineering from Seoul National University, Seoul, Korea, in 1984 and 1986, respectively, and a Ph.D. degree in the Department of Electrical Engineering from Purdue University, West Lafayette, IN, in 1990. From 1990 to 1994, he was a Senior Researcher at Samsung Electronics, Suwon, Korea. In 1994, he joined the School of Electrical and Computer Engineering, Sungkyunkwan University, Suwon, Korea, as an Assistant Professor, where he is currently a Professor. His research interests include robotics, embedded systems, and factory automation.