Journal of The Korea Society of Computer and Information
Vol. 21 No. 11, pp. 39-47, November 2016
www.ksci.re.kr
http://dx.doi.org/10.9708/jksci.2016.21.11.039

FPGA implementation using a CLAHE contrast enhancement technique in the termal equipment for real time processing

Jin-Hyun Jung*

Abstract

In this paper, we propose an approach for real time computation of rayleigh CLAHE using a FPGA. The contrast enhancement technique should be applied in thermal equipment having a low contrast image. And thermal equipment must be processed in real time. The CLAHE is an improved algorithm based Histogram Equalization, but the HW design is complex. A value greater than a given threshold in CLAHE should be equally distributed on the other histogram bin, this process requires iterations for the distribution. But implementation of this processing in the FPGA is constrained, so this section was implemented on the assumption of the histogram distribution or modified the operation process or implemented separately in the CPU. In this paper, we designed a distinct redistribution operation in two stages. So FPGA was designed for easy, this was designed to be distributed evenly without the assumptions and constraints. In addition, we have designed a CLAHE with the rayleigh distribution to the FPGA. The simulation shows that the proposed method provides a better image quality in the thermal image.

▸ Keyword : Histogram equalization, Rayleigh CLAHE, FPGA design.

I. Introduction

모든 물체는 절대온도 0도(0K = 273.16℃) 이상에서 적외선 에너지를 방출하며, 열상장비는 물체에서 방출되는 적외선 에너지를 감지하여 표적과 배경 간 온도차를 영상의 명암차로 재현하는 장치이다. 따라서 열상장비는 주/야간 제한없이 사용 가능하고 군사적으로 관측 장비 또는 사격 통제 장비로서 운용되고 있다. 하지만 열상장비는 불균일 보정(Non-Uniformity Correction) 처리 과정과 더불어 열영상의 밝기값 분포가 한 영역에 집중되어 있는 저대비 영상이므로 영상 가독성을 향상시키기 위한 방법이 필수적으로 필요하다. 그래서 열상장비는 불균일 보정과 명암대비 향상 기법이 선행되어야만 영상기반의 탐지/추적/분류와 같은 추가 신호처리가 수행될 수 있고 우리가 원하는 관측장비 또는 사격장비로서 운용 가능하게 된다. 이처럼 열상장비에서 명암대비 향상기법은 가장 기본으로 이뤄져야 하는 과정이기에 많은 연구가 이뤄지고 있다[1-4].

영상의 명암대비 향상기법으로 히스토그램 평활화(HE, Histogram Equalization)가 있다. 히스토그램 평활화는 입력 영상 전체의 밝기값 분포(히스토그램)가 집중되어 있는 것을 균일하게 분포되도록 영상의 밝기값을 변환하는 것으로써 알고리즘이 단순하여 보편적으로 사용되었다. 하지만 밝은 밝기값의 분포와 어두운 밝기값의 분포가 같이 존재하는 상태에서는 충분한 화질 개선의 효과를 얻을 수 없었다. 이러한 문제점을 개선하고자 적응 히스토그램 평활화[5](AHE, Adaptive Histogram Equalization)가 제시되었다. 적응 히스토그램 평활화는 영상을 여러 개의 블록으로 나누어 블록별로 HE를 수행하는 방법이다. 그래서 적응적 히스토그램 평활화에서는 블록별로 개별 변환함수가 구해지고 밝은 밝기값을 가지는 영역과 어두운 밝기값을 가지는 영역 모두에 대해서 영상개선 효과를 얻을 수 있었다. 하지만 적응적 히스토그램 평활화에서는 명암대비 향상이 너무 지나쳐서 노이즈가 증가되고 과도한 명암대

• First Author: Jin-Hyun Jung, Corresponding Author: Jin-Hyun Jung
*Jin-Hyun Jung(jinhyun77.jung@hanwha.com), Hanwha Systems Co.
• Received: 2016. 11. 09, Revised: 2016. 11. 16, Accepted: 2016. 11. 25.

비로 인해 전체적으로 영상 화질을 저하시킬 수 있다[6].

AHE에서 과도한 명암대비가 발생하는 원인은 영상의 특정 밝기값의 분포가 지나치게 모여있는 경우, 변환함수의 기울기가 급격하게 커지게 되어 과도한 명암대비 향상이 발생하게 되는 것이다. 대비 제한 적응 히스토그램 평활화(CLAHE, Contrast Limited Adaptive Histogram Equalization)는 밝기값의 분포(히스토그램)에서 지정된 한계점(이하 ClipLimit값)을 초과하는 경우에 히스토그램값을 ClipLimt값으로 제한하여 변환함수의 기울기를 제한하며, 이는 지나친 명암대비 향상을 억제하여 노이즈 증가 및 화질 저하를 막을 수 있다[7,8].

하지만 CLAHE는 히스토그램값이 ClipLimit값을 초과하는 경우에 ClipLimit값을 초과하는 양(Excess값)만큼 전체 히스토그램에 균등하게 배분되어 ClipLimit값을 넘지 않도록 하는 과정이 필요하다. 이 재분배 과정은 회귀적 연산 과정으로 이루어져야 한다[9]. 그리고 재분배 연산 과정은 CPU에서는 쉽게 구현 가능하겠지만, 실시간 처리를 위한 FPGA 구현에서는 설계의 복잡성으로 인해, 히스토그램 분포를 가정하거나[10], 연산과정을 변형하여 구현하였다[11].

본 논문에서는 회귀적으로 연산해야만 하는 재분배 연산 과정을 FPGA 설계가 용이하도록 2단계 연산과정으로 구분하여 설계함으로써, 어떤 가정 및 제약 없이 ClipLimit값이 균등하게 배분될 수 있도록 하였다.

그리고 본 논문에서 적용하고자 하는 열영상 장비는 열영상 특성상 영상화면의 대부분이 어두운 밝기값을 가지는 배경으로 구성되며, 관심있는 물체 또는 타겟은 작은 영역에만 분포되어 있으며 배경보다 밝은 밝기값을 가진다. 그런데 CLAHE는 기본적으로 HE기반이라 히스토그램 분포를 uniform 형태로 변환하게 되며 그렇게 되면 어두운 배경과 밝은 물체의 밝기값 차이가 상대적으로 적어지게 되어, 물체 식별 및 탐지에 어려움이 발생하게 된다.

본 논문에서는 이러한 문제점을 해결하고자 uniform 분포를 가지는 CLAHE가 아니라, rayleigh 분포를 가지는 CLAHE를 사용하였다. rayleigh 분포를 가지는 CLAHE는 uniform 분포를 가지는 CLAHE 알고리즘에서 히스토그램 정합[12]이 추가 수행되어야 하며, 그렇게 함으로써 영상 대부분을 차지하는 배경이 너무 밝게 변환되지 않도록 하여 물체 탐지가 용이하도록 하였다.

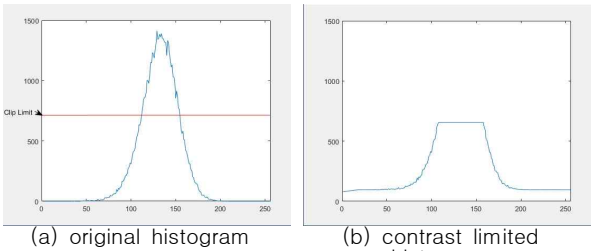
본 논문의 구성은 다음과 같다. 2장에서는 CLAHE에 대해서 간략히 설명하며, 3장에서는 rayleigh CLAHE를 FPGA로 구현하는 과정을 단계별로 설명하며, 본 논문에서 제안하는 Excess값 재분배 연산과정을 FPGA 설계하는 방안에 대해 설명한다. 그리고 구현된 로직을 가지고 시뮬레이션한 영상 결과와 FPGA 구현에 사용된 리소스를 분석한다. 마지막으로 4장에서 결론을 맺는다.

II. Preliminaries

1. Related works

1.1 Contrast Limited Histogram Equalization

대비제한 히스토그램 평활화는 히스토그램 평활화[4](HE)



(a) original histogram (b) contrast limited histogram

Fig. 1. The redistribution of Excess amount

와 유사하나 Fig. 1에서와 같이 입력 영상의 밝기값 분포(히스토그램)가 ClipLimit값을 초과할 경우에는 히스토그램을 ClipLimit값으로 제한하고 초과된 양(Excess값)은 전체 히스토그램에 균등하게 배분하여 ClipLimit값을 넘지 않도록 하는 과정이 추가된 것으로 구현 절차는 아래와 같다.

1. 입력영상 전체로부터 각 pixel의 밝기값(i)을 읽는다.
2. 해당 pixel의 밝기값에 대해서 발생 빈도수(히스토그램: h(i))를 구한다.
$$h(i) = n_i \quad (i = 0, 1, 2 \ldots, L-1)$$
3. h(i)가 ClipLimit값을 초과하면 h(i)는 ClipLimit값으로 제한하고 ClipLimit값을 초과하는 값은 전체 히스토그램에 균등하게 배분한다.
4. 히스토그램에서 영상 밝기값의 확률 밀도 함수(P(i))를 구한다.
$$p(i) = \frac{n_i}{N}$$
5. 확률 밀도 함수에서 누적 분포 함수를 구하고 누적분포함수를 변환함수로 사용하여 새로운 밝기값(y(v))을 계산한다.
$$y(v) = (L-1) \times Cdf(v) = (L-1) \sum_{i=0}^{v} p(i)$$
$$(v = 0, 1, 2 \ldots, M-1)$$

n_i는 i의 밝기값을 갖는 픽셀의 개수이고 N은 전체 픽셀 수, L-1과 M-1은 픽셀의 최대 밝기값이다.

1.2 Contrast Limited Adaptive Histogram Equalization

CLAHE[9]는 Fig. 2에서 보는 바와 같이 영상을 균일한 크기의 블록으로 분할하고 각 블록에 대해서 대비제한 히스토그램 평활화(CLHE)를 수행하는 방법이다. Fig. 2에서는 1280x1024 해상도의 영상을 256x256 크기의 블록으로 분할하였으며 4개의 행과 5개의 열로 분할되어 총 20개의 서로 겹치지 않은 블록이 생성되었다.

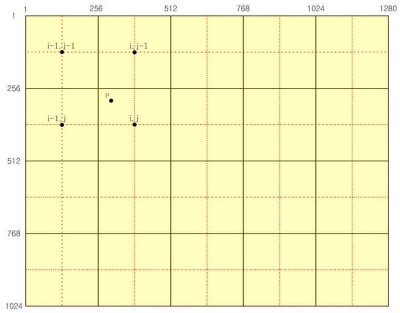


Fig. 2. Dividing the image with a resolution of 1280x1024 into block size of 256x256

분할된 각 블록에 대해서 CLHE를 수행하게 되면 각 블록마다 uniform 분포를 가지는 변환함수를 생성하게 된다. 그러나 rayleigh 분포를 가지는 CLAHE를 구현하기 위해서는 히스토그램 정합을 추가 수행한다.

히스토그램 정합[12]은 식 (1)와 같이 uniform 분포를 가지는 변환함수(누적 분포함수, Cdf(v))에서 히스토그램의 분포가 rayleigh 확률 밀도함수가 되도록 rayleigh 확률밀도함수를 가지는 누적 분포함수의 역함수를 적용한다.

$$f(v) = L \times Cdf^{-1}(Cdf(v)) \quad (1)$$

Cdf(v)는 CLHE를 수행해서 구한 누적분포함수이고 $Cdf^{-1}(x)$는 rayleigh 확률밀도함수를 가지는 누적분포함수의 역함수이다. Fig. 3의 좌측 그림은 CLHE를 수행해서 구한 누적분포함수(변환함수)에 의해 밝기 변환된 히스토그램 분포이며, 우측 그림은 rayleigh 역함수를 적용한 누적분포함수(변환함수)에 의해서 밝기 변환된 히스토그램 분포이다. Fig. 3에서 보는 바와 같이 CLHE를 수행했을 때, 히스토그램의 분포가 전체 밝기값으로 퍼져 있으며 상대적으로 밝은 밝기값을 가지는 영역이 많아짐을 알 수 있다. 하지만 CLHE를 수행한 결과에서 rayleigh 히스토그램 정합을 추가 수행하였을 때는 밝은 밝기값을 가지는 영역이 상대적으로 적은 것을 알 수 있다. 이는 앞서 설명한 바와 같이 열영상에서는 탐지가 더 용이함을 알 수 있다.

마지막으로 각 블록별로 식 (1)에 의해서 구해진 변환함수를 가지고 개별적으로 명암대비 향상을 수행하게 되면 서로 다른 변환함수로 인해서 경계면마다 서로 다른 밝기값을 가지는 blocking effect가 초래된다. 그래서 Fig. 4와 같이 인접한 네 개 블록의 변환함수 결과값에 대해 식 (2)과 같이 이중 선형 보간법을 수행하여 CLAHE 연산을 완료한다.

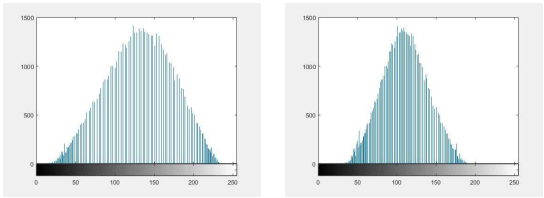


Fig. 3. Histogram of applying transformation function of uniform distribution and rayleigh distribution

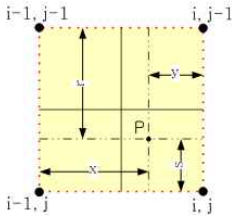


Fig. 4. Bilinear interpolation with adjacent blocks at Pixel P from (i, j) region

$$P_{new} = \frac{s}{r+s}\left(\frac{y}{x+y} f_{i-1,j-1}(P_{old}) + \frac{x}{x+y} f_{i,j-1}(P_{old})\right) + \frac{r}{r+s}\left(\frac{y}{x+y} f_{i-1,j}(P_{old}) + \frac{x}{x+y} f_{i,j}(P_{old})\right) \quad (2)$$

P_{old}는 변환되기 전의 영상 밝기값, f()함수는 식 (1)에 의해 구해진 변환함수이며 x, y, r, s는 인접 블록 중심과의 거리이다.

III. The Proposed Scheme

본 논문에서는 CLAHE 알고리즘의 Excess값을 재배분하는 회귀적 연산과정에서 FPGA 설계가 용이하면서도 어떤 가정 및 제약 없이 ClipLimit값이 균등하게 배분될 수 있는 방법을 제안한다. 그리고 rayleigh 분포를 가지는 CLAHE를 FPGA로 구현하여 실시간 처리가 가능하도록 하였다.

1. Design parameter selection and implementation overview

본 논문에서는 현재 열영상 장비에서 많이 사용되는 1280x1024 해상도를 기준으로 픽셀 깊이(pixel depth)는 9bit, 픽셀 클럭은 54MHz에 프레임율은 30fps로 가정하여 실시간 처리가 가능하도록 FPGA 설계하였다. 소형화를 고려하여 FPGA 내부에서 모든 연산 과정이 이루어지도록 설계하였다.

Rayleigh CLAHE에서 선정해야 할 파라미터로 크게 분할 블록수와 ClipLimit, rayleigh 분포 설정값이 있다. ClipLimit값은 히스토그램의 밀도값을 제한하는 한계값으로 식 (3)와 같이 한계점 파라미터 α에 의해서 결정된다.

$$ClipLimit = \frac{N}{L}(1 + \alpha \times (L-1)) \quad (3)$$

N은 블록 전체 픽셀의 개수이고, L은 영상 밝기 값의 범위이다. 한계점 파라미터 α는 0에서 1 사이의 실수이고 0의 값을 가질 때는 히스토그램 분포를 모두 동일하게 배분하는 것이고,

1의 값을 가질 때는 히스토그램의 밀도값을 제한하지 않고 원본 히스토그램을 유지하게 된다. ClipLimit값을 자동으로 설정하기 위한 연구도 이뤄지고 있지만 본 논문에서는 사용자가 설정 가능하도록 하였으며, 실험적으로 528을 기본값으로 설정하였다. 또한 rayleigh 분포 설정값 역시 실험적으로 0.4로 설정하였으며 사용자가 수정 가능하도록 설계하였다. 이는 열상장비를 운용하는 사용자의 환경에 맞게 유동적으로 변경 가능하도록 운용의 폭을 넓힐 수 있도록 하였다.

그리고 분할 블록수는 메모리 리소스와 rayleigh 역함수를 계산하는 floating 연산처리 시간을 고려해서 4x5(=20개)로 분할하였다. 분할 블록수가 많아지면 작은 영역으로 세분화되어 명암대비 향상이 이뤄짐으로 작은 영역도 선명하게 표현되는 장점이 있지만 사용되는 메모리량이 많아지는 단점이 있다. 그리고 rayleigh 역함수는 floating 연산을 수행해야만 하고 FPGA 내부의 DSP48 모듈을 사용해야 하는데, 블록 개별적으로 rayleigh 역함수를 수행하기에는 DSP48 모듈의 사용량이 너무 많아지고 공용으로 사용하기엔 실시간 처리가 힘들어지는 제약이 따른다. 그래서 본 논문에서는 1280x1024 해상도를 가지는 영상에 대해서 4x5개로 분할하여 실시간 처리가 가능하도록 하였다.

Fig. 5는 rayleigh CLAHE의 전체 FPGA 설계를 개략적으로 보여 주고 있다. Fig. 5의 좌측으로부터 영상 입력이 되면 각 블록에 대해 개별적으로 CLHE를 수행한다. 앞서 영상을 4개의 행과 5개의 열로 블록 분할한 것처럼 Fig. 5에서도 4개의 행으로 분리되어 있으며, 각 행에는 5개 메모리가 단계별로 배치되어 있고 각 블록별로 1개의 메모리가 할당되어 순차적으로 처리된다.

Rayleigh CLAHE의 각 블록은 총 6단계를 거치면서 수행하는데 1, 2, 3, 5단계에서 개별적으로 할당받은 메모리를 활용하여 단계별 연산이 수행된다. 첫 번째 단계의 메모리는 ClipLimit값에 의해 히스토그램값이 제한되어 저장된다. 첫 번째 행의 블록 영상이 입력되는 동안에는 Fig. 5의 스위치로 표현한 것과 같이 첫 번째 행의 1단계 메모리에만 저장된다. 두 번째 행의 블록 영상이 입력되는 시간부터 첫 번째 행의 1단계 메모리에 저장된 히스토그램값이 2단계 메모리에 임시저장 된 후, 히스토그램의 ClipLimt값을 초과하는 픽셀의 개수(Excess값)가 각 히스토그램에 균등하게 배분되는 과정을 거치게 된다. 히스토그램이 균등하게 배분되면 3단계 메모리에 누적하며 더하는 Cdf연산을 수행하게 된다. 저장된 Cdf 함수는 5단계 메모리에 저장되기 전에 4단계에서 rayleigh 역함수를 floating 연산으로 수행하게 되며, 5단계에 최종 저장될 때는 fixed point로 저장한다. 이 과정은 5개의 블록 메모리에 대해서 순차적으로 연산된다.

마지막으로 6단계에서는 새로 입력되는 영상의 밝기값에 의해서 5단계에 저장된 변환함수로 밝기값을 변환하고 인접 블록 4개에 대해서 이중 선형 보간을 수행하여 최종 우리가 원하는 밝기값으로 변환한다.

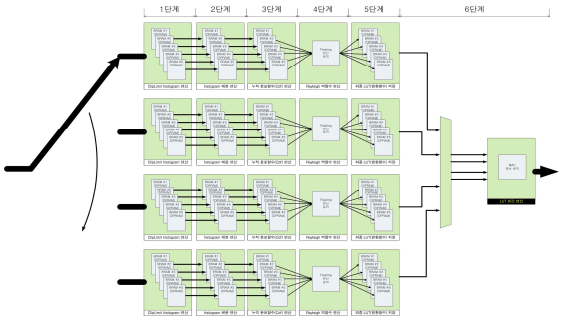


Fig. 5. Simplified schematic of the rayleigh CLAHE

2. FPGA design and implementation

Rayleight CLAHE는 총 6단계로 구분되어 FPGA 설계 되었으며 각 단계별 FPGA 설계 과정을 설명하고자 한다.

2.1. The 1st stage of CLAHE

첫 번째 단계는 입력영상 밝기값에 대해서 ClipLimit값으로 제한된 히스토그램을 구하는 단계로서 히스토그램은 FPGA 내부의 DPRAM을 이용하여 구현한다. Fig. 6에서와 같이 pixel의 밝기값을 DPRAM의 주소값으로 입력하여 pixel 밝기값의 히스토그램을 읽어서 1을 더해 줌으로서 입력 영상에 대한 히스토그램을 구하게 된다. 만약 히스토그램값이 ClipLimit값보다 큰 경우에는 ClipLimit값으로 히스토그램값을 제한하며 Excess값을 1 증가시킨다. 그리고 본 로직에서는 다음 입력영상의 히스토그램을 구하기 전에 DPRAM을 0으로 초기화 할 수 있도록 설계되었다.

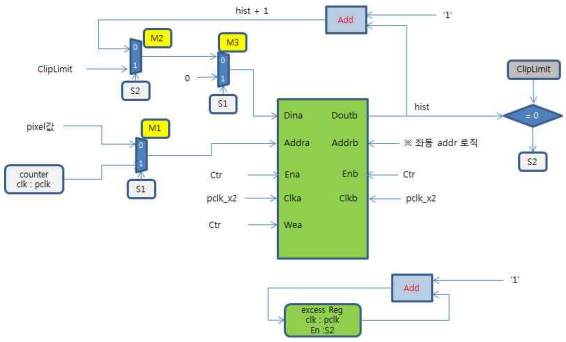


Fig. 6. Simplified schematic of the histogram generation and excess counter

2.2. The 2nd stage of CLAHE

두 번째 단계는 1단계에서 구한 ClipLimit값으로 제한된 히스토그램에 ClipLimit값을 초과하는 픽셀의 개수(Excess값)를 전체 히스토그램값에 균등하게 분배하는 과정으로 본 논문에서 제안하는 절차는 다음과 같다.

Excess값 재분배 첫 번째 연산과정

1. 이전 단계에서 구한 히스토그램값은 FPGA내부의 DPRAM에 저장하고 Excess값은 16bit의 레지스터에 저장한다.
2. 16bit Excess값에서 pixel depth보다 큰 상위 bit값(이하 히스토그램 배분값)을 전체 히스토그램값에 더해줌.
3. 만약 DPRAM에 저장된 히스토그램값이 ClipLimit이라면 DPRAM값은 그대로 유지하고 Excess값이 저장된 레지스터에 히스토그램 배분값을 더해줌.
4. 만약 DPRAM에 저장된 히스토그램값이 ClipLimit보다 작지만 히스토그램 배분값을 더했을 때, ClipLimit보다 크다면 DPRAM에는 ClipLimit값을 저장하고, Excess값이 저장된 레지스터에는 ClipLimit값을 초과한 값만 더해줌.
5. 위 과정을 DPRAM에 저장된 히스토그램 전체에 대해서 수행하고 만약 동시에 계산된 Excess값에서 상위 bit값이 0보다 큰 값이라면 절차 2~4를 반복 수행함.

Excess값 재분배 두 번째 연산과정

6. 만약 Excess값의 상위값이 0이라면 Excess값의 하위값에 의해서 배분신호를 생성해서 균일하게 배분함. (배분신호는 Excess값의 bit 위치에 따라 배분 간격을 다르게 생성. Excess값의 하위값에서 최상위bit가 1이라면 히스토그램 bin값을 2^1씩 건너뛰면서 히스토그램값에 1을 더해주고, 최상위 bit에서 n번째 위치한 bit가 1이라면 히스토그램 bin값을 $2^{(n+1)}$씩 건너뛰면서 히스토그램값에 1을 더해주도록 신호 생성)

위 과정을 의사코드 절차로 표현하면 다음과 같다.

```
/* 초기값 저장 */
   next_excess[15..0] = 1단계 excess값[15..0];
   //[15..0] 표시는 16bit 크기의 레지스터 표시

/* Excess값 재분배 첫 번째 연산과정 */
   next_addval[6..0] = next_excess[15..10];
   //Excess의 상위 bit값만 히스토그램 배분값으로 사용
   //pixel_depth 9bit 일 때, 상위 bit값.

   next_excess[15..0] = "0000000" & ...
                              next_excess[8...0];

   while(next_addval != 0) {
    for(i=0; i<2^pixel_depth; i++) {
      if( bram_histogram(i) >= ClipLimit )
          bram_histogram(i) = bram_histogram(i);
          next_excess = next_excess + next_addval;
      elseif(bram_histogram(i)+next_addval>ClipLimit)
          bram_histogram(i) = ClipLimit;
          next_excess = next_excess +
            (bram_histogram(i)+next_addval-ClipLimit);
      else
          bram_histogram(i) = bram_histogram(i) +
                                  next_addval;
    }
    next_addval[6..0] = excess[15..10];
    next_excess[15..0] = "0000000" &
                               next_excess[8...0];
  }
/* Excess값 재분배 두 번째 연산과정 */
  for(i=0; i<pixel_depth; i++){
    if( next_excess(8-i) == 1) {
      for(j=2^i+1; j<2^pixel_depth; j += 2^(i+1) ){
          bram_histogram(j) = bram_histogram(j) +1
      }
    }
  }
```

Excess값 재분배의 첫 번째 연산과정은 Excess값이 전체 히스토그램 bin에 균등한 값(위 절차에서 next_addval값)을 배분하기에 충분히 큰 값이기 때문에 히스토그램값이 ClipLimit값을 넘는지에 관심을 두고 excess/2^{pixel_depth}값의 정수부분이 0보다 크다면 연산을 반복한다. Excess값 재분배의 두 번째 연산과정은 Excess값이 전체 히스토그램 bin에 균등한 값으로 배분하기에는 부족하다. 그래서 Excess의 값에 의해서 배분하는 위치를 결정한다. 만약 Excess값의 MSB(most significant bit, 위 절차에서는 (pixel_depth값 -1)인 8)가 1이라면 10진값으로 2^8=256이고 이는 256개를 균등하게 배분할 수 있음을 의미한다. 그리고 Excess값의 LSB(least significant bit)가 1이라면 10진값으로 2^0=1이고 이는 1개를 균등하게 배분할 수 있음을 의미한다. 그런데 이 배분신호는 9bit(pixel depth) 카운터를 설계했을 때, 카운터의 각 bit위치에서 발생하는 펄스의 rising edge가 위 배분위치와 동일함을 알 수 있고, Excess값에 따라 9bit 카운터에서 발생하는 신호를 사용해서 배분위치를 균등하게 나누게 된다.

예를 들면 Fig. 7에서 보는 바와 같이 Excess값 재분배 두 번째 연산과정에서 Excess값이 366(binary => 0101101110)이 남았고, 366은 $2^8+2^6+2^5+2^3+2^2+2^1$ 로 구성된다. 366을 균등하게 배분하기 위해서 2^8개수만큼 배분신호를 생성하여 배분한다. 이는 9bit 카운터에서 8번째 bit위치에서 발생하는 신호의 rising edge마다 1을 배분한다. 2^6개수만큼의 배분신호는 9bit 카운터에서 6번째 bit위치에서 발생하는 신호의 rising edge마다 1을 배분한다. 즉 Excess값에서 1의 값을 가지는 각 bit위치에 따라 배분신호가 생성되고 있으며, Excess값에 의해서 균등하게 배분된다.

Fig. 7. Distribution waveform signals generated in the redistribution step 2

Fig. 8은 Excess값 재분배 연산과정을 FPGA로 구현한 개략도이다. CLAHE의 첫 번째 단계와 마찬가지로 DPRAM을 중심으로 설계되어 있으며, DPRAM에 히스토그램값을 저장한다. 하지만 DPRAM의 주소는 Excess값이 순차적으로 배분되어야 하기 때문에 counter값으로 입력한다. 그리고 제어신호 생성부에서는 제일 먼저 Excess값의 상위값(히스토그램 배분값)이 0보다 큰지를 조사하여 Excess값 재분배 연산 단계를 정의한다. 그리고 저장된 히스토그램값이 ClipLimit값보다 큰지와 히스토그램값이 (ClipLimit - 히스토그램 배분값)보다 큰지도 조사하여 Excess값과 히스토그램값을 경우에 맞게 재연산한다. 그리고 만약 Excess값의 상위값이 0이라면 Excess값 재분배 연산 단계가 2단계로 변경되며, DPRAM의 Write enable 신호가 배분신호에 의해서 특정 위치의 히스토그램 bin만 1씩 더해진다.

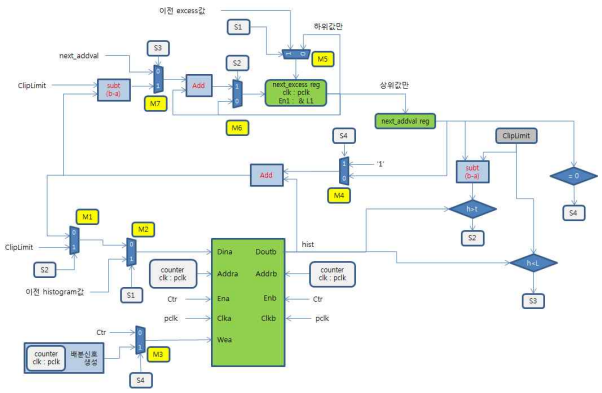


Fig. 8.Simplified schematic of the histogram distribution logic

2.3. The 3rd to 5th stage of CLAHE

세 번째 단계는 Excess값이 균등하게 배분된 히스토그램값에서 Cdf함수를 구하는 단계로서 히스토그램에서 순차적으로 읽으면서 누적하여 Cdf 함수를 구하게 되며 비교적 구현이 간단하다.

네 번째와 다섯 번째 단계는 rayleigh분포를 적용하기 위해서 앞서 구한 Cdf 함수에 rayleigh 역함수를 적용하는 단계로서 FPGA의 floating point IP를 사용하여 구현하였다. rayleigh역함수의 식은 다음과 같다.

$$\sqrt{-2\alpha^2 \times \ln\left[1-\left\{1-\exp\left(\frac{-1}{2\alpha^2}\right)\right\}\times\frac{Cdf}{N}\right]} \qquad (4)$$

α는 rayleigh 분포의 형태를 정의하는 파라미터값으로 본 논문에서는 0.4로 설정하였으며, α값이 작을수록 어두운 영역이 많이 표현되게 되며, α값이 클수록 밝은 영역이 많이 표현되게 된다. N은 블록 내 픽셀수를 나타낸다. 위 수식을 계산하기 위해서 먼저 fixed point값의 Cdf값을 floating point로 바꿔서 수식 (4)를 6개의 단계별 연산으로 나눠서 단계별 연산을 순차적으로 수행하고 최종 계산된 floating point값은 fixed point로 변환되어 변환함수 메모리에 저장한다.

2.4. The 6th stage of CLAHE

마지막 6단계는 이전 프레임에서 변환함수를 구한 상태에서 새로 입력되는 픽셀값을 인접한 4개의 블록별 변환함수를 사용해서 밝기값을 변환하고 이를 이중 선형 보간하여 최종 변환된 밝기값을 구하는 단계이다.

Fig. 9는 선형보간을 수행하기 위한 FPGA 로직 블록도로서 영상입력에 따라 변환함수가 변경되어야 하기 때문에 Mux로 구현되어 있으며, 픽셀 위치값에 따른 가중치값을 곱해서 인접 4개의 결과값을 더하는 이중 선형보간이 구현되어 있다.

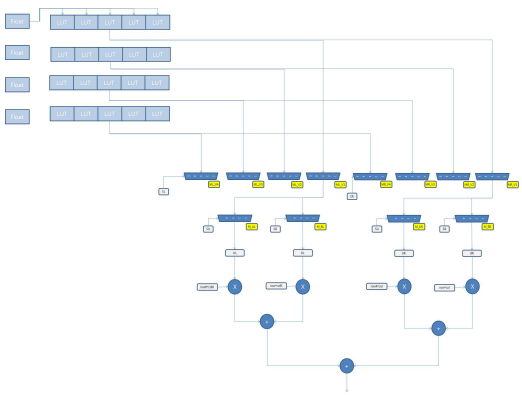


Fig. 9. Simplified schematic of the Bilinear interpolation logic

3. Experimental Results

FPGA 설계 검증은 Fig. 10과 같이 수행하였다. 먼저 시험영상을 Matlab의 “adapthisteq” 함수로 rayleigh CLAHE의 결과값을 먼저 확인하였다. 그리고 FPGA 시뮬레이션을 수행하기 위해서 시험영상을 Matlab에서 text 시험 데이터로 생성하여 Xilinx의 Vivado 툴에서 입력으로 사용하여 function simulation을 수행하였다. 그리고 FPGA 시뮬레이션 출력값인 text 파일를 Matlab으로 읽어와서 모니터에 영상으로 출력하게끔 하였다. 또한 FPGA 출력영상이 대조비 개선이 이뤄졌는지와 출력값이 Matlab 함수 결과값과 유사한지를 확인하였다. 그리고 최종적으로 Post implementation simulation을 수행하여 결과값이 같은지를 확인하였다.

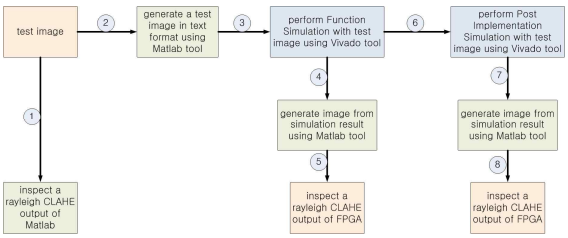


Fig. 10. FPGA design verification procedures

Fig. 11은 Vivado 툴에서 시뮬레이션한 결과 파형으로 영상 첫 번째 행의 5개 블록이 2프레임 동안 CLAHE가 수행되면서 출력되는 파형이다. 2프레임의 시간 동안에 첫 번째 프레임에서는 영상 입력값에 의해 변환함수를 계산하고, 두 번째 프레임에서는 새롭게 입력되는 영상값이 이전 프레임 영상값으로 계산된 변환함수에 의해서 새로운 밝기값으로 변환되게 된다.

Fig. 11의 A영역은 첫 번째 행의 5개 블록에 대해 ClipLimit 값으로 제한된 히스토그램을 구하는 과정이며 B영역은 Excess 값을 재분배하는 과정이다. C영역은 rayleigh 역함수를 계산하는 과정이며, D영역은 2번째 프레임의 영상이 입력되면 각 인접 블록의 변환함수에 의해서 밝기 변환된 값들이 이중 선형보간을 수행하여 최종 CLAHE 변환된 값이 출력되는 과정이다.

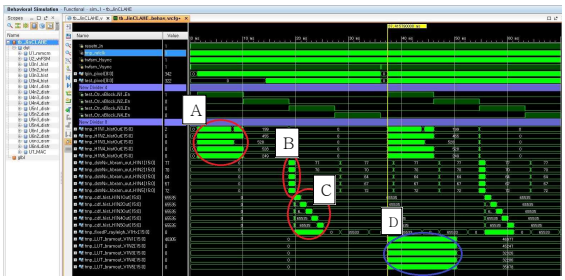
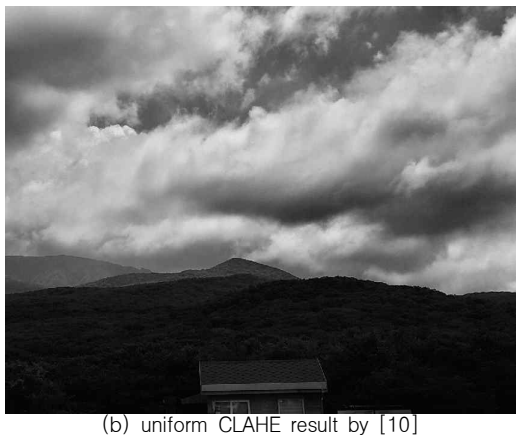


Fig. 11. CLAHE timing diagram in the image block of the first row

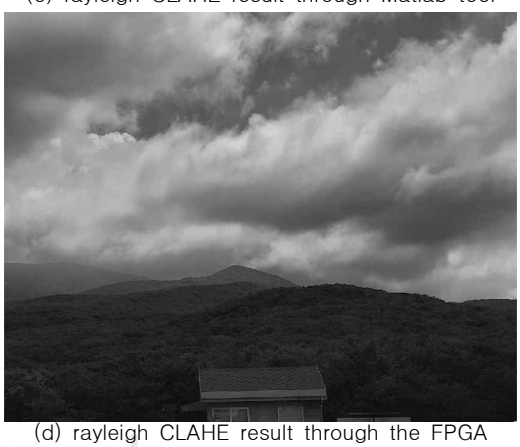
Fig. 12는 밝은 밝기값을 가지는 영역과 어두운 밝기값을 가지는 영역이 동시에 있는 영상에 대해서 명암대비 향상 기법을 적용하여 비교한 결과이다. Fig. 12(a)는 시험영상으로 밝은 밝기값을 가지는 하늘과 어두운 밝기값을 가지는 숲이 동시에 있으며, 숲이 전체적으로 어둡게만 보이고 숲의 윤곽을 확인할 수 없는 상태이다. Fig. 12(b)는 Matlab으로 [10]의 uniform CLAHE를 수행한 결과로서 시험영상에 비해서 명암대비가 향상되어 숲의 윤곽과 건물이 뚜렷하게 구분되어 보이는 것을 확인할 수 있다. 하지만 시험영상에 비해 하늘이 많이 강조되어 너무 밝게 보이는 것을 확인할 수 있다. Fig. 12(c)는 Matlab으로 rayleigh CLAHE를 수행한 결과로서 uniform CLAHE에 비해서 구름은 적게 강조되었지만 숲은 더 밝게 보이는 것을 확인할 수 있다. Fig. 12(d)는 FPGA에서 rayleigh CLAHE를 수행한 결과로서 Matlab의 rayleigh CLAHE와 유사한 결과를 보이며 다소 더 선명하게 보이는 것을 확인할 수 있다.

(a) test image



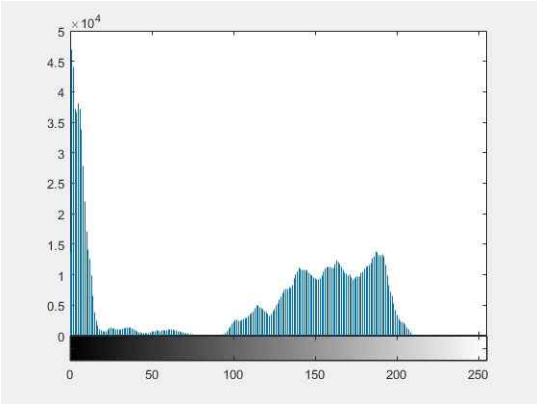
(b) uniform CLAHE result by [10]

(c) rayleigh CLAHE result through Matlab tool

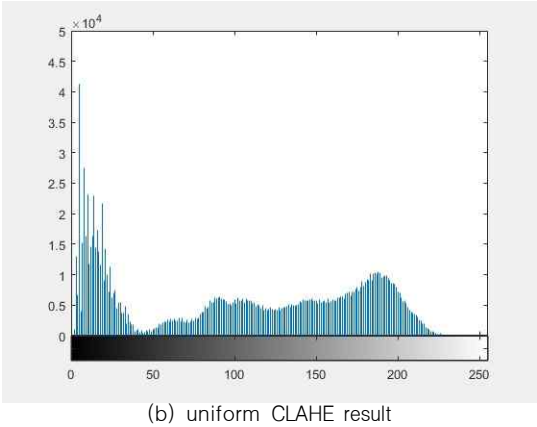


(d) rayleigh CLAHE result through the FPGA

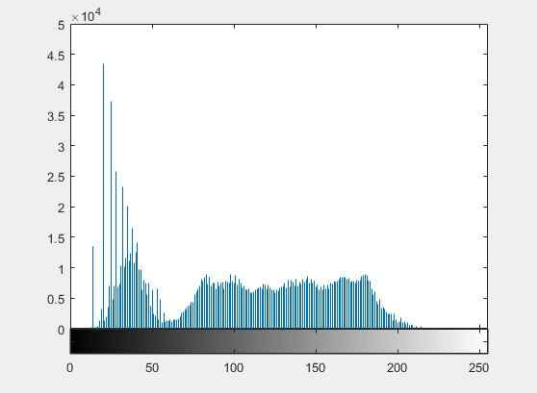
Fig. 12. Comparison of contrast enhancement techniques in low-contrast images



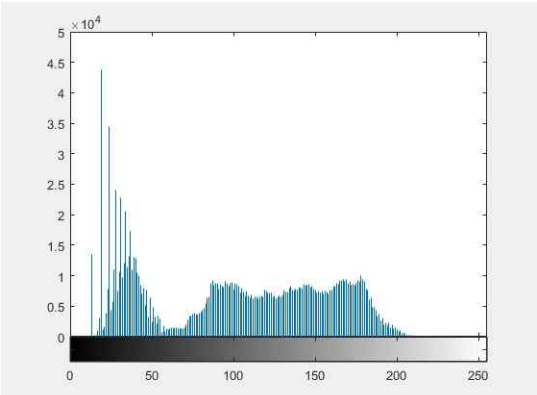
(a) test image



(b) uniform CLAHE result



(c) rayleigh CLAHE result through Matlab tool



(d) rayleigh CLAHE result through the FPGA implementation

Fig. 13. Histogram of result image

Fig. 13은 Fig. 12의 각각의 결과 영상에 대해 히스토그램(밝기값 분포)을 보여주고 있다. Fig. 13(a)는 시험영상의 히스토그램이다. Fig. 13(b)는 Matlab으로 [10]의 uniform CLAHE를 수행한 영상의 히스토그램으로 시험영상의 히스토그램에 비해서 전반적으로 균등하게 분포 되었음을 알 수 있고 상대적으로 밝은 밝기값 분포가 많아졌음을 볼 수 있다. Fig. 13(c)와 Fig. 13(d)는 rayleigh CLAHE를 수행한 영상의 히스토그램으로 밝은 밝기값 분포가 Fig. 13(b)처럼 많이 강조되지 않아서 밝은 값을 가지는 소형 대공 표적탐지에 더 적합함을 예상할 수 있다.

Table 4. Comparison of FPGA resource usage

		Proposed	[10]
Image resolution		1280x1024	640x480
Num. of block		20 (4x5)	64 (8x8)
Device		Kintex-7	Virtex-5
Logic utilization	Num. of Slice	4664	3691
	Num. of BRAM	40	32
	Num. of DSP48	44	18
Max frequnecy		68.0MHz	101.3MHz

Table 1은 [10]와 본 논문에서 CLAHE를 하드웨어 구현하는데 사용한 FPGA 로직 사용량과 최대 처리속도이다. 본 논문에서는 [10]보다 더 높은 영상 해상도를 가지기 때문에 FPGA 구현에 더 열악한 조건이지만, 분할 블록수는 [10]가 더 많아서 FPGA 구현에 더 열악한 조건이다. 그리고 [10]는 uinform CLAHE를 구현하였지만 본 논문에서는 rayleigh CLAHE를 구현하여 히스토그램 정합 알고리즘이 추가로 더 구현되어 BRAM과 DSP48이 더 사용되고 있음을 알 수 있다. 하지만 [10]와 본 논문에서 사용한 FPGA 로직량이 큰 차이를 보이지 않고 Fig. 12를 통해 본 논문이 소형 대공 표적탐지에 더 유리함을 알 수 있다.

IV. Conclusions

열상장비에서는 반드시 명암대비 향상 기법이 적용되어 사용되어야 하는데, 그동안 HE를 기반으로 하는 다양한 변형된 방법들이 적용되어 왔다. 본 논문에서는 명암대비 향상 기법중에서 회귀적 연산으로 인해 FPGA 구현에 제약이 있었던 CLAHE를 어떤 가정이나 제약없이 ClipLimit값을 균등하게 배분될 수 있는 FPGA 설계 방안을 제시하였다. 또한 열영상 관측 장비에서 좀 더 좋은 화질을 제공하기 위해서 rayleigh 분포를 가지도록 히스토그램 정합을 추가하였다. rayleigh 분포의 히스토그램 정합이 추가되었지만 30fps의 1280x1024 해상도 영상에 대해서 실시간 처리가 가능함을 확인하였으며 FPGA 로직 사용량이 내부 모듈별로 5~9%만 사용하게끔 설계되어, 추가적으로 탐지/추적 알고리즘이 FPGA내에 추가 구현될 수

있는 여지를 두었다. 그리고 시험영상으로 FPGA 시뮬레이션한 결과, 명암대비가 향상되어 원영상에서는 식별이 힘들었던 숲의 윤곽과 건물들이 뚜렷하게 보이는 것을 확인 할 수 있었다.

향후에는 설계된 로직을 실제 보드에 탑재하여 성능 확인할 예정이다. 그리고 현재 관측장비에서는 열상카메라 뿐만 아니라, 칼라영상을 지원하는 EO카메라도 같이 장착하고 있다. 따라서 구현된 FPGA 로직을 칼라영상에도 적용하여 칼라영상에서도 화질을 향상시킬 수 있는 방안에 대해 연구할 계획이다.

REFERENCES

[1] MY. Lee, YJ. Han, HS. Hahn, "Contrast Improvement Technique Using Variable Stretching based on Densities of Brightness," Journal of The Korea Society of Computer and Information, Vol. 15, No. 2, pp. 37-45, October 2010.

[2] SM. Jung, BW. On, "An efficient quality improvement scheme of magnified image by using the information of adjacent pixel values," Journal of The Korea Society of Computer and Information, Vol. 18, No. 2, pp. 49-57, February 2013.

[3] R. Choudhary, and S. Gawade, "Survey on Image Contrast Enhancement Techniques," International Journal of Innovative Studies in Sciences and Engineering Technology, Vol. 2, No. 3, pp. 21-25, March 2016.

[4] S. V. Aher, and S. S. Vasekar, "A Review: Histogram Equalization Algorithms for Image Enhancement using FPGA," International Journal of Advanced Research in Computer and Communication Engineering, Vol. 5, No. 4, pp. 711-714, April 2016.

[5] S. M. Pizer, E. P. Amburn, J. D. Austin, R. Cromartie, A. Geselowits, et al., "Adaptive Histogram Equalization and Its Variations," Computer vision, graphics, and image processing, Vol. 39, No. 3, pp. 355-368, 1987.

[6] D. P. Sharma, "Intensity Transformation using Contrast Limited Adaptive Histogram Equalization," International Journal of Engineering Research, Vol. 2, No. 4, pp. 282-285, August 2013.

[7] P. D. Ferguson, T. Arslan, A. T. Erdogan, and A. Parmley, "Evaluation Of Contrast Limited Adaptive Histogram Equalization (CLAHE) Enhancement on FPGA," In SoCC, pp.119-122, 2008.

[8] H. Cho, and H. Kye, "The Clip Limit Decision of Contrast Limited Adaptive Histogram Equalization for X-ray Images using Fuzzy Logic," Journal of Korea Multimedia Society, Vol. 18, No. 7, pp. 806-817, July 2015.

[9] A. M. Reza, "Realization of the Contrast Limited Adaptive Histogram Equalization (CLAHE) for Real-Time Image Enhancement," Journal of VLSI signal processing systems for signal, image and video technology, Vol. 38, No. 1, pp.35-44, August 2004.

[10] V. Schatz, "Low-latency histogram equalization for infrared image sequences: a hardware implementation," Journal of real-time image processing, Vol. 8, No. 2, pp.193-206, June 2013.

[11] K. Kokufuta, and T. Maruyama, "Real-time processing of contrast limited adaptive histogram equalization on FPGA," 2010 International Conference on Field Programmable Logic and Applications. IEEE, pp.155-158, 2010.

[12] R. C. Gonzalez and R. E. Woods, "*Digital Image Processing*" Prentice-Hall, third edn, pp.150-160, 2002.

Authors

Jin Hyun Jung received the B.S. and M.S. degrees in Electrical Communication Engineering from Kumoh National Institute of Technology, Korea, in 2000 and 2002, respectively

In 2004, he joined Hanwha Systems. Co., Republic of Korea, and he is currently a senior researcher. He is interested in Tracking and computer vision.