COMS이미지 센서용 효과적인 White Balance 구현

송형돈,이동훈,손승일

한국한신대학교 정보통신학과

Implementation of Efficient White Balance of CMOS Image Sensor

Hyung Don Song, Dong Hun Lee, Seung Il Sonh
Dept. of Information and Communication Hanshin University

e-mail : brothermoney@hs.ac.kr

요 약

상황에 따라 조명의 밝기나 종류 등에 영향으로 CMOS 이미지 카메라 센서로부터 입력 받은 영상의 색상은 원색과 차이가 있다. 이러한 왜곡된 색상을 Red, Green, Blue와 휘도를 이용하여 원래의 색으로 표현하는 과정이 White Balance이다. 사람은 색이 눈으로 입사되는 물리적인 자극 외에 대뇌의 작용으로 광원이 바뀌어도 같은 색으로 인지하는 특징이 있다. 따라서 이러한 과정이 없을 시에는 우리의 눈으로 보는 것과 영상장치를 통해서 모니터에 표시되는 영상의 색상과 차이가 생긴다. 본 논문에서는 RGB와 휘도를 이용하는 방법과 논문에서 제안한 히스토그램을 이용하는 방법에 대해 소프트웨어를 사용하여 각각의 상황에 따라 알고리즘을 적용하여 WB를 수행한 결과에 대하여 PSNR을 구하여 비교 분석한 후 최적화된 알고리즘을 이용하여 하드웨어 설계 언어인 VHDL을 사용하여 구현하고, ModelSim6.0a를 이용하여 데이터를 검증한다.

키워드
AWB, 영역처리, VHDL, YCbCr

Ⅰ. 서 론

우리가 보는 영상과 CMOS 이미지 카메라 센서로부터 입력 받은 영상은 상황에 따라 조명의 밝기나 종류 등에 영향으로 차이가 있다. 영상 장치의 경우는 주어진 색온도의 반사광을 그대로 재현하여 보여주기 때문에, 인간의 눈으로 볼 때는 색의 차이로 인지하게 된다. 이러한 왜곡된 색상을 Red, Green, Blue와 휘도를 이용하여 원래의 색으로 표현하는 과정이 White Balance이다. 사람은 색이 눈으로 입사되는 물리적인 자극 외에 대뇌의 작용으로 광원이 바뀌어도 같은 색으로 인지하는 특징이 있다. 광원이 형광등과 같은 청색광원이거나 태양광과 같은 백색광원, 혹은 백열등과 같은 적색광원으로 이동해도 흰색은 항상 흰색으로 느끼게 된다. 따라서 이러한 과정이 없을 시에는 우리의 눈으로 보는 것과 영상장치를 통해서 모니터에 표시되는 영상의 색상이 차이가 생긴다. 본 논문에서는 RGB와 휘도를 이용하는 방법은 하드웨어 설계 시 연산량이 많아 출력결과도 오래 걸리고 칩의 규모가 커지기 때문에 수행속도의 향상과 저전력 사용을 위해 Cb, Cr값을 가지고 히스토그램을 이용하는 방법을 프로그램으로 검증한 후 최적화된 알고리즘으로 VHDL 하드웨어 설계언어를 이용하여 코딩한 후 ModelSim6.0a를 이용하여 파형을 검증하고 이미지를 비교하여 설계에 대한 결론을 내리고자 한다.[1][2]

Ⅱ. 본 론

2.1 YCbCr의 개념

디지털영상장비에서 사용하는 Color Space는 휘도(Y)성분과 색상(C)성분된 YCbCr이다. Cb는 R성분에서 휘도(G)성분을 뺀 값이고 Cr은 B성분에서 휘도(G) 성분을 뺀 값이며 R-Y, B-Y형태로 쓰이며 색차신호라 한다.

2.2 White Balance의 개념

이미지의 색차신호 R-Y, B-Y를 이용한 평면에서 원점주위는 무채색이다. 흰색, 회색, 검은색 모두 중앙에 모인다. 이상적인 white와 black의 좌표는 (128,128)인 원점중앙이다. Auto White Balance란 그림 1처럼 입사된 광원의 색온도와

상관없이 R-Y, B-Y의 색차성분 백터의 평균치가 화이트라고 가정을 하고, 색 평균치를 화이트(128,128)로 이동시키는 것이다.

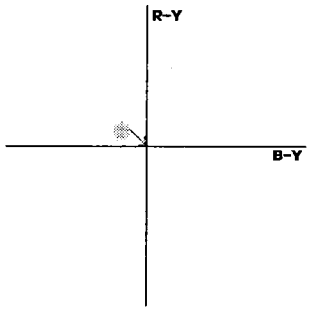

그림 1. B-Y R-Y 평면

2.2 RGB와 휘도를 이용한 AWB 방법

세 가지 색상으로부터 휘도 Y는 그림 2와 같이 계산된다.

$$Y = 0.299R + 0.587G + 0.114B$$

$$E(Y) = WrE(R) + WgE(G) + WbE(B)$$
$$Wr=0.299\ ,\ Wg=0.587\ ,\ Wb=0.114$$

$$ArE(R) = AgE(G) = AbE(B)$$

$$E(Y(A)) = WrArE(R) + WgAgE(G) + WbAbE(B) = E(Y)$$

$$A_r = \frac{E(Y)}{E(R)},\ A_g = \frac{E(Y)}{E(G},\ A_b = \frac{E(Y)}{E(B}$$

그림 2. RGB와 Y의 관계식

위 계산된 계수들은 새로이 입력되는 각각의 색상에 적용되어 보정하는 것이다.[] 계수를 사용하여 AWB 설계 시에 WB블록을 제외한 YCbCr 변환블록에서 8비트 곱셈기가 3개가 더 사용된다.

2.3 Cb,Cr 의 히스토그램을 이용한 WB 방법

AWB는 이상적인 흰색의 Cb, Cr값인 128에 이미지의 Cb, Cr의 평균값을 수렴시키는 것이다. 특정 광원의 영향으로 변한 색차값의 변화를 히스토그램을 사용하여 AWB를 한다. 그림 3은 본 논문에서 제안한 히스토그램을 사용할 때 적용되는 계산식이다.

(수렴시키고자 하는값) / (픽셀들의 평균값) * (구하고자 하는 픽셀값)

그림 3. 히스토그램 계산식

2.4 AWB 영역분할처리

카메라 위치에 따른 색상을 균형있게 조절하기 위해 영상 프레임을 아래 그림 4처럼 5영역으로 분할한다. 본 논문의 테스트 이미지는 256x200 이미지로 사용한다. 또한 각 영역에서 피사체의 중요 위치에 따른 가중치를 줄 수도 있다. 본 논문에서는 AWB를 위해 Cb, Cr의 평균값을 구할 때 Top영역과 Center영역값의 평균값을 더한 후 2로 나누어 사용한다.

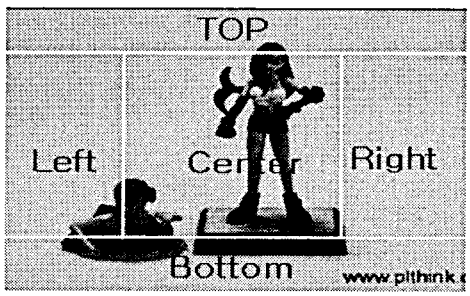

그림 4. 영역처리

III. White Balance 설계

3.1 WB 전체블록도

그림 5는 YCbCr포맷을 가지고 WB를 수행하는 블록도이다. Multi_division image size 블록은 영역처리를 위해 이미지에서 각 영역에 대해 면적을 구하고, WB Processing블록은 Multi_division image size 블록에서 얻어진 면적값을 사용하여 AWB를 수행한다.

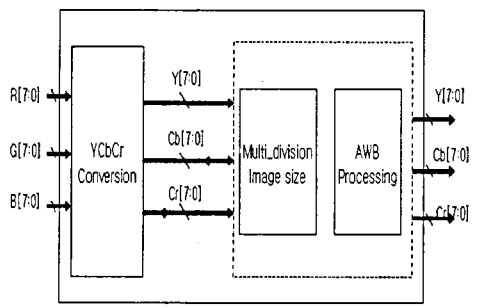

그림 5. WB 블록도

3.2 영역처리 블록도

영역분할의 가운데 영역 위치값은 초기에 레지

스터에 저장된 기본값과 사용자 입력값인 사용자 값이 있다. 두 값은 모드에 의해서 결정된다. 이미지의 전체 가로길이와 세로길이, 가운데 영역의 값을 가지고 각 영역-위쪽, 왼쪽, 가운데쪽, 오른쪽, 아래쪽-의 가로, 세로 길이를 구한 후 곱하여 면적을 구하여 각각 레지스터에 값을 저장한다. 저장된 레지스터값은 AWB Processing 블럭에서 영역별 Cb, Cr의 합산된 값을 나누어 평균값을 구할 때 사용한다.

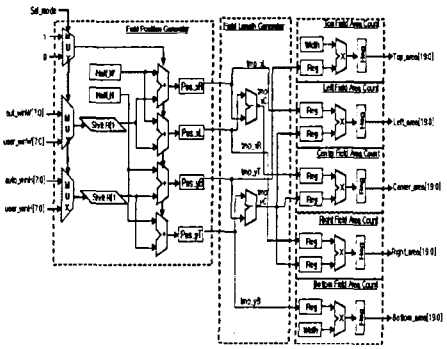

그림 6. 영역처리 블록도

3.3 WB 블록도

그림 7은 각 AWB_en 신호가 활성화되면 입력 영역별 적분을 한 후 평균값을 구한다.

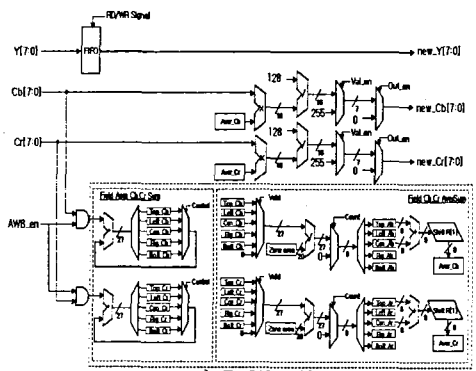

그림 7. WB 블록도

처음 센서를 통하여 입력된 데이터는 일련의 ISP처리 과정을 거친 후 YCbCr 컨버전을 통해서 변환된 데이터들은 Cb, Cr의 계수값을 구하는데 사용된다. 다음 입력되는 이미지의 데이터는 이전 이미지의 Cb, Cr 값에서 얻어진 계수값과의 연산을 통하여 지속적인 AWB가 수행된 수행한 이미지가 Out_en 신호가 활성화 되면 출력된다. 계수와의 연산은 곱셈기와 나눗셈기를 사용한다.

IV. 설계 결과

4.1 설계 결과

본 논문의 수행 결과를 테스트하기 위해 원본 이미지 그림 8에 대하여 그림 9는 RGB와 휘도를 사용하여 AWB를 수행한 결과이고 그림 10은 CbCr에 대해 히스토그램을 사용하여 AWB를 수행한 결과이다. 청색광원이 강조된 원본이미지에 대하여 그림 9와 그림 10 모두 청색광원이 제거된 것을 볼 수가 있다.

그림 8. 원본 이미지

그림 9. RGB와 Gain을 적용한 AWB

그림 10. 색차값과 히스토그램 적용한 AWB

또한 PSNR식에서 도출한 값이 원본 이미지와 두가지 방법으로 처리하여 비교한 결과 PSNR값의 차이가 거의 없었다. YCbCr를 이용한 AWB가 처리속도와 적은 데이터 비트율 가지며 RGB를 사용한 결과와도 차이가 거의 없는 효과적인 AWB를 수행할 수 있었다.

4.2 시뮬레이션 결과

그림 11은 입력이미지(256x200)를 두 번 입력하

여 AWB를 수행한 결과이다. Vaild_en 신호가 활성화되면 AWB가 동작된다. 첫 번째 입력이미지에서 영역별 면적값과 Cb,Cr의 계수값을 구한 후 두 번째 입력이미지에 바로 적용하여 AWB가 수행된 결과가 출력된다. 연산 수행 시 덧셈, 뺄셈에 대한 처리는 최소 지연이 발생한다. 나눗셈 연산 시 결과를 얻기 위한 딜레이는 30클럭 데이터 지연을 소요한다. 지연 요소 결과는 프레임 단위 간 지연시간을 고려한다면 연산 시 크게 문제가 발생하지 않으므로 실시간 데이터 처리가 가능하다고 생각한다[4][5].

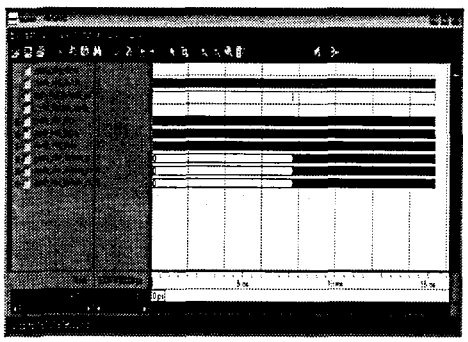

그림 11. AWB 시뮬레이션 파형 결과

제안한 AWB 알고리즘은 주위 환경에 따른 색차값을 고려하기위한 방법이다. AWB를 수행할 때 YCbCr 컨버젼에서 불필요한 연산을 하지 않아 처리속도가 향상된다.

표1 회로 합성 결과

Block	No. of Gates	Timing	Target Device
AWB(Auto White Balance)	73,208	Minimum period: 12.528ns (Maximum Frequency: 79.821MHz)	XCV1000e -hq240

V. 결 론

CMOS 센서는 CCD센서에 비교하여 화질의 단점을 개선하기 위한 방법이 중 우선적인 필수 고려 요인이다. ISP전처리 과정을 거쳐 RGB보다 적은 연산과 적은 비트율을 사용하는 AWB를 설계하였으며 영역 분할처리로 이미지의 필드를 나누고 주변 환경에 따라 변화하는 출력이미지에 대하여 효과적인 알고리즘을 적용하여 처리했다. 제안한 아키텍처에서는 연산 수행 시 지연 발생을 최소화하여 덧셈, 뺄셈 연산 수행과 파이프라인으로 처리하였고 분할영역처리에서 사용되는 가중치에 대한 연산을 설계에 반영하지 않았다. 본 논문은 카메라 이미지 센서용 ISP처리 적용칩에서 사용될 AWB를 구현하였으며, 이를 효과적으로 처리 할 수 있을 것으로 사료된다.[2][3]

참 고 문 헌

[1] 김병수, 이준석, 정유영, 고성제, "자동 초점 자동 노출 및 자동 화이트 밸런스 비디오 카메라의 설계 및 구현", 대한전자공학회 하계종합학술대회 논문집 제 22권 제1호(C), 2001.

[2] Orly Yadid-Pecht and Alexander Belenky "In-Pixel Autoexposure CMOS APS", IEEE JOURNAL OF SOLID-STATE CIRCUITS, VOL. 38, NO. 8, 2003

[3] S. Battiato, A. Bosco, A. Castorina, G. Messina, "Automatic Global Image Enhancement by Skin Dependent Exposure Correction" NSIP, 2003.

[4] Rongzheng ZHOU, Xuefeng CHEN, Feng LIU, Jie HE, Tiankang LIAO, Yanfeng SU, Jinghua YE, Yajie QIN, Xiaofeng YI, Zhiliang HONG, " System-on-Chip for Mega-Pixel Digital Camera Processor with Auto Control Functions", IEEE. Trans 2003.

[5] June-Sok Lee You-Young Jung, Byung-Soo Kim, Sung-Jea Ko, "An Advanced Video Camera System with Robust AF, AE, and AWB CONTROL", IEEE, Trans. vol, 47. No 3, 8. 2001.

[6] Yung-Cheng Liu, Wen-Hsin Chan and Ye-Quang Chen, "Automatic White Balance For Digital Still Camera", IEEE. Trans, Vol 41, No 3, 8 1995.