Journal of the KNST

2023; Vol.6, No.4; pp. 494-497

ISSN: 2635-4926

https://doi.org/10.31818/JKNST.2023.12.6.4.494

Received: 2023/11/28
Revised: 2023/12/11
Accepted: 2023/12/27
Published: 2023/12/31

*Corresponding Author:
Seon-Jeong Hwang
PGM Seeker Lab, LIG Nex1, Gyunggi-Do, Republic of Korea
333 Pangyo-ro, Bundang-gu, Seongnam-si, Gyeonggi-do, 13488, Republic of Korea
Tel: +82-31-326-9305
Fax: +82-31-326-9007
E-mail: seonjeong.hwang@lignex1.com

Abstract

본 논문에서는 이중선형보간법을 이용하여 실시간 영상의 크기를 축소하였다. 우선, 원 영상과 축소할 영상의 크기의 비율을 계산하여 대응되는 좌표를 구한다. 이후, 계산된 좌표의 인접 좌표 픽셀을 이용하여 좌표의 픽셀 세기를 구한다. 해당 계산 과정은 병렬로 처리하여 연산 시간을 단축하였다. CPU에서도 동일한 연산을 시행하여 FPGA에서 병렬처리한 연산 결과와 비교하였다.

This study reduced real-time image size using bilinear interpolation. First, calculate the ratio of the size of the original image and the image to be reduced to obtain the corresponding coordinates. After, calculate pixel intensity of the coordinates using the adjacent coordinate pixels of the calculated coordinates. Process time of the calculate process is reduced by using parallel-processing. Also, same calculation was processed on the CPU and compared with the results of the parallel processing on the FPGA.

Keywords

병렬처리(Parallel Processing),
이중선형보간법(Bilinear Interpolation),
실시간 영상처리(Real-Time Video Processing),
FPGA

Acknowledgement

이 논문은 2023년도 한국해군과학기술학회 동계학술대회 발표 논문임.

FPGA를 이용한 실시간 영상의 이중선형보간법 고속화 설계

Design of High-speed Bilinear Interpolation for Real Time Video Using FPGA

황선정[1*], 이다빈[1], 박성현[1], 김홍락[2]

[1]LIG넥스원 PGM탐색기연구소 선임연구원
[2]LIG넥스원 PGM탐색기연구소 수석연구원

Seon-Jeong Hwang[1*], Da-Been Lee[1], Sunghyun Park[1], Hong-Rak Kim[2]

[1]Research engineer, PGM RF & IIR Seeker R&D Lab, LIG Nex1
[2]Chief research engineer, PGM RF & IIR Seeker R&D Lab, LIG Nex1

1. 서론

영상처리는 산업의 다양한 분야에서 쓰이고 있다. 방위산업 또한 예외는 아니다. 전자광학망원경을 사용하여 인공위성의 형태·궤적·활동을 감시하거나, 유도무기에 카메라를 장착하여 목표물을 탐지·추적하는 등 다양한 분야에서 사용되고 있다. 이러한 분야에서 영상처리에는 실시간 및 고속 처리가 요구된다. 특히 적 유도탄 또는 항공기를 요격하는 유도무기의 경우, 표적이 매우 빠른 속도로 이동하는 특성이 있어 실시간으로 고속 영상처리가 요구된다.

최근에는 AI, 딥러닝 등 다양한 기술이 방위산업에 접목되어 사용되고 있으며, 카메라의 성능 또한 개선되고 있다. 하지만 획득한 영상을 최초 획득한 상태 그대로 쓰게 된다면, 연산에 많은 자원이 필요하게 되거나 프로세서의 성능이 이를 감당하지 못하여 영상이 실시간으로 처리되지 않고 처리가 지연될 것이다. 또한, 딥러닝의 경우 일부 모델은 특정한 크기의 영상을 요구하기도 한다. 이러한 문제를 해결하기 위해 graysacle, 이진화, 확대/축소, 회전/변환 등의 영상 전처리가 필요하다. 본 연구의 설계에는 이중선형보간법을 사용하여 영상 크기를 축소하는 영상 전처리를 시행한다.

2. 본론

본 설계는 Fig. 1과 같이 크기가 큰 $W \times L$의 영상을 $M \times N$ 크기의

영상으로 축소한다. 영상 축소에는 이중선형보간법을 이용한다. 또한 영상은 Fval, Lval, Dval, VideoData로 이루어진 프로토콜로 입력을 받고, 실시간 영상을 최소한의 지연으로 처리하기 위해 FPGA를 이용하여 설계하였다.

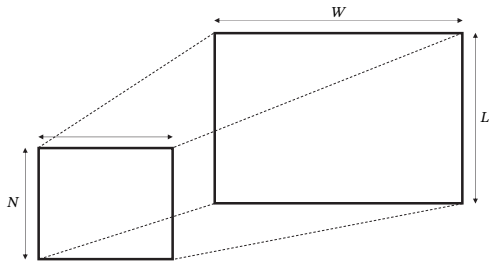

Fig. 1. Image size reduction

2.1 이중선형보간법

이중선형보간법을 설명하기 위해서는 우선 선형보간법에 대한 이해가 필수적이다. 선형보간법은 두 점 (x_1, x_2)과 두 점의 값(y_1, y_2)이 주어졌을 때, 두 점 사이의 임의의 점 x의 값 y를 구하는 방법이다.

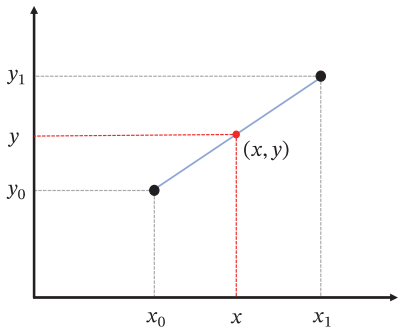

Fig. 2. Linear Interpolation

Fig. 2와 같은 상황이 주어질 때, 좌표 x의 값 y를 구하는 식은 다음과 같다.

$$y = y_0 + (y_1 - y_0)\frac{x - x_0}{x_1 - x_0} \quad (1)$$

Fig. 2와 같은 상황에서 x_0에서 x_1까지의 거리가 1이라고 가정하고, x_0에서 x까지의 거리를 d_1, x_1에서 x까지의 거리를 d_2라 할 때, 식 (1)은 식 (2)와 같이 정리할 수 있다.

$$y = d_2 y_1 + d_1 y_2 \quad (2)$$

위에서 서술한 선형보간법을 1차원 선이 아니라 2차원 평면에 적용한 것이 이중선형보간법이다.

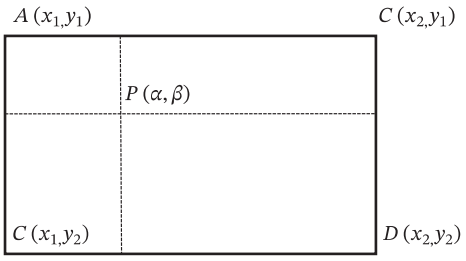

Fig. 3. Bilinear interpolation

Fig. 3에서 dx_1, dx_2, dy_1, dy_2를 각각 식 (3) - 식 (6)으로 정의할 때, 픽셀의 세기인 P의 값은 식 (7)로 구할 수 있다[1].

$$dx_1 = (\alpha - x_1) \quad (3)$$

$$dx_2 = (x_2 - \alpha) \quad (4)$$

$$dy_1 = (\beta - y_1) \quad (5)$$

$$dy_2 = (y_2 - \beta) \quad (6)$$

$$P = Adx_2dy_2 + Bdx_1dy_2 + Cdx_2dy_1 + Ddx_1dy_1 \quad (7)$$

2.2 영상 프로토콜

본 설계의 영상신호 전송에 사용된 영상 프로토콜은 Fval, Lval, Dval, VideoData이다. Fval은 Frame Valid로, High일 때 유효한 라인을 나타내고, Lval은 Line Valid로, 유효한 픽셀일 때 High로 정의된다. Dval은 Data Valid로, High일 때 데이터가 유효하다는 것을 나타낸다. 마지막으로 VideoData는 해당 픽셀의 세기를 나타낸다.

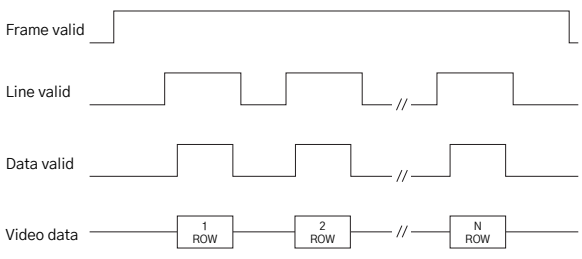

Fig. 4. Image protocol

2.3 FPGA

FPGA는 field programmable gate array의 약자로 LUT(look-up-table)와 FF(flip-flop)으로 디지털 회로를 구성하는 소자이다. FPGA는 CPU 또는 DSP와 다르게, 소스 코드를 작성하면 해당 코드는 위에서부터 아래로 순차적으로 작동하는 것이 아니라, 모든 코드가 클럭에 따라 병렬적으로 동작한다. 단순 클럭 속도는 CPU 또는 DSP보다 느리지만, 병렬로 처리한다는 이점으로 설계에 따라 처리 속도는 더 높다. 또한, task 또는 interrupt service routine을 별도로 처리하지 않고 데이터 송수신 및 처리를 할 수 있기 때문에 실시간 데이터 처리가 안정적이다.

2.4 상세 설계

영상 프로토콜의 Fval이 시작될 때, 영상 프레임에 필요한 값들을 계산한다. 우선, $N \times M$의 좌푯값을 $W \times L$의 영상의 좌푯값으로 변환해야 한다. x 좌표의 경우 W의 길이가 M으로 변환되어야 한다. 이 경우, Fig. 5와 같이 M에서의 좌푯값은 M을 W로 확장했을 때의 좌푯값이다. 따라서, W를 M으로 나눈 값인 R_x가 M의 좌표계를 W의 좌표계로 변환할 때 각 좌표 간의 값이 된다.

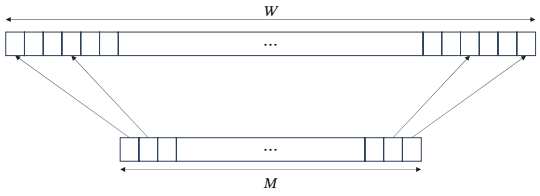

Fig. 5. Resized coordinates

y 좌표 또한 x 좌표와 마찬가지로 L을 N으로 나누는 연산을 시행하여 R_y를 획득한다. 이때, R_x와 R_y가 floating point 형식으로 지정될 경우 곱하기 연산에서 소요되는 자원 및 지연 시간이 증가하므로, floating point 값을 반올림하여 1/8의 정확도까지만 사용하였다.

이후, Dval이 시작될 때 VideoData의 값들을 버퍼에 저장한다. 이때 하나의 라인을 계산하기 위해서는 두 개의 라인이 필요하므로, 최소 2개의 버퍼를 사용하여야 한다. 본 설계에서는 안정성을 높이기 위해 2개의 버퍼를 더블 버퍼링하여 총 4개의 버퍼를 사용하였다. 버퍼링된 VideoData의 값들은 좌푯값과 같이 병렬적으로 계산된다.

첫 번째 단계에서는, 버퍼에 저장된 값을 계산된 좌표를 이용하여 읽어온다. 앞서 y 좌표를 계산하여 필요한 위·아래 라인 두 줄을 저장하는 동작을 서술하였는데, 이때 두 줄이 모두 저장되었다면 해당 정보를 이용하여 $M \times N$ 영상의 한 줄을 계산할 수 있다. 저장된 값을 불러올 때는, 한 줄씩 모두 불러오는 것이 아닌 하나의 픽셀값 P를 계산할 때 필요한 4개의 좌표를 순차적으로 불러온다.

두 번째 단계에서는 4개의 좌표에 따른 픽셀값을 순차적으로 불러오며 좌표에 맞게 식 (1) - 식 (4)를 계산한다. 이 때, 모든 x값과 y값은 정수 형태이고, α와 β만 고정 소수점이기 때문에, 해당 수식을 간략화하여 dx_1, dy_1는 α와 β의 소수값만 분리하여 2의 보수를 취하였고, dx_2, dy_2는 α와 β의 소수값을 그대로 사용하였다.

세 번째 및 네 번째 단계에서는, 식 (5)를 계산하기 위해 여러 곱셈 연산을 시행한다. 우선, 두 번째 단계에서 dx_1, dx_2, dy_1, dy_2의 계산이 완료될 때마다 각각 서로 곱한다. 이때, 해당 신호값은 모두 fixed point 형식으로, 단순 곱셈기를 이용하여 곱셈이 가능하고 지연시간 또한 적다. 이후, 곱셈기에서 출력되는 값과 첫 번째 단계에서 읽어온 좌푯값을 다시 곱한다.

다섯 번째 단계에서는 네 번째 단계에서 계산된 곱셈기의 출력값을 모두 더한다. 이때, 네 항이 모두 계산되고 덧셈을 한 번에 시행하게 되면 덧셈의 지연시간이 한 클럭을 초과할 수 있다. 따라서, 각 항이 계산되어 출력될 때마다 임의의 신호에 덧셈을 시행하여 마지막 네 번째 곱셈이 완료된 다음 클럭에 덧셈이 완료될 수 있도록 설계하였다. Fig. 6는 첫 번째부터 다섯 번째 과정의 계산 병렬 처리의 블록 다이어그램을 나타낸 그림이다.

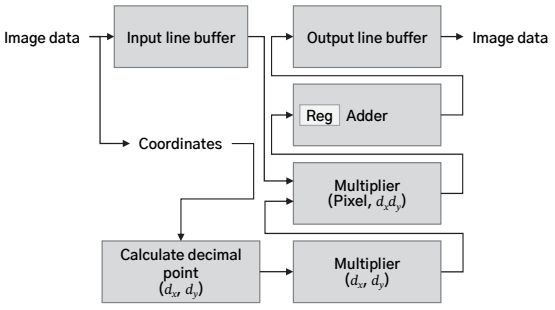

Fig. 6. Pixel calculation

위와 같은 병렬 계산으로 출력된 픽셀값 P는 영상 프로토콜 신호로 변환하기 위하여 FIFO에 저장한다. 하나의 라인이 모두 저장되었다면, 영상 프로토콜 신호 변환 모듈이 FIFO에서 데이터를 출력하며, 해당 데이터 타이밍에 맞게 Fval, Lval, Dval, VideoData를 생성한다.

2.5 연산 시간 측정 결과

해당 설계의 연산 시간을 측정한 FPGA는 Xilinx Artix-7 시리즈 (xc7a200t)이다. 해당 설계에 사용된 FPGA의 주요 자원 사용량은 Table 1과 같다.

Table 1. FPGA resource usage

Resource	Estimation	Available	Utilization (%)
LUT	636	133,800	0.46
LUTRAM	11	46,200	0.02
FF	651	269,200	0.24
BRAM	2.5	365	0.68

지연 시간 측정을 위한 예제 설계에서는 원 영상인 $W\times L$은 400×400, 축소 영상인 $M\times N$은 128×128로 설계하였다. 원 영상의 라인이 종료되는 Dval이 시작하는 시점과 축소된 영상의 Dval이 시작하는 시점의 차이를 라인 지연시간이라고 하면, 영상의 라인 지연시간은 23.175 us이다.

Fig. 7은 FPGA에서 원 영상과 이중선형보간법을 적용한 영상의 타이밍을 나타낸다. 이중선형보간법의 한 라인을 출력하기 위해서는 원 영상이 두 라인을 필요로 하기 때문에, 두 번째 라인이 입력되며 이중선형보간법의 한 라인이 출력된다.

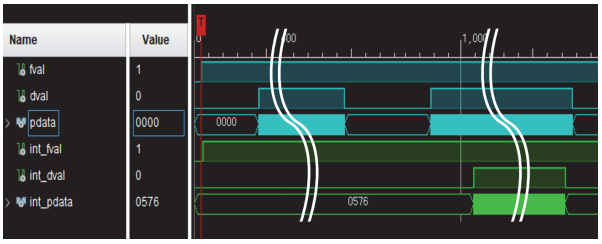

Fig. 7. Calculated timing

또한, 원 영상의 Fval이 끝나는 시점과 축소된 영상의 Fval이 끝나는 시점의 차이를 프레임 지연시간이라고 하면, 프레임 지연시간은 Dval의 지연시간과 동일한 23.175 us이다.

프레임 지연시간이 매우 적은 이유는, 해당 설계는 라인 데이터가 스트리밍되어 입력될 때마다 라인별로 데이터 처리를 하기 때문이다. 따라서, 프레임 지연시간과 라인 지연시간이 동일하게 된다. 해당 설계와 속도를 비교하기 위해 NXP사의 PowerPC 계열 CPU인 T2080에서 동일한 이중선형보간법을 실행하였다. CPU에서의 프레임 지연을 메모리에 업로드된 영상을 축소하는 데 걸리는 시간이라고 하면, 측정된 프레임 지연 시간은 3.91 ms이다.

3. 결론

본 연구는 이중선형보간법을 사용하여 영상 크기를 축소하는 전처리를 설계하였다. FPGA를 이용하여 각 연산 과정을 병렬 처리하여 실시간 영상의 지연 및 연산 시간을 최소화하였다. 연산 시간 측정 결과 FPGA로 이중선형보간법을 실행한 시간이 CPU로 실행한 시간보다 약 168배 빠르다는 것을 알 수 있다.

참고문헌

[1] Bok-Deuk Song et al., “Automatic Drawing Conformity Inspection System Using Image Features Matching and Bilinear Interpolation,” J. KIEEME, Vol. 25, No. 4, 2012, pp. 321-327.