논문 2010-47SD-6-4

영상 스케일러의 저전력 VLSI 구조 설계 및 계수 최적화

(Low-power VLSI Architecture Design for Image Scaler and Coefficients Optimization)

한 재 영*, 이 성 원**

(Jae-Young Han and Seong-Won Lee)

요 약

기존의 영상 스케일러(scaler)들은 연산량과 하드웨어 복잡도를 줄이기 위해 선형 보간과 같은 간단한 보간을 적용함으로써 화질을 희생시키거나, 고품질 영상을 얻기 위하여 복잡한 보간 기법을 적용함으로써 전력소모와 크기가 큰 하드웨어 구조를 적용하여 왔다. 그러나 영상기기들의 소형화와 고화질 영상에 대한 사용자들의 욕구 증대로 소형, 저전력이면서 결과 영상의 화질 또한 우수한 스케일러의 개발이 중요시되고 있다. 따라서 본 논문은 실시간, 고화질, 소형, 저전력의 목표를 모두 달성할 수 있는 래스터 스캔(raster scan) 방식의 스케일러 하드웨어 구조를 제안한다. 본 논문에서 제안하는 스케일러는 기존의 3차 보간(cubic interpolation) 기법과 룩업테이블(look-up table) 구조를 개선하여 저전력화와 소형화를 달성하였다. 제안하는 스케일러 구조의 특징은 기존의 실시간 스케일러가 포함하던 버퍼를 라인메모리로 대체하여 메모리 접근 횟수를 줄임으로써 저전력을 달성할 수 있도록 했다는 것이며, 또한 기존의 룩업테이블 구조에서 사용하던 3차 보간 수식을 재정리하여 곱셈기 수와 룩업테이블의 크기를 줄임으로써 하드웨어를 소형화하는 방법을 제안하였다. 마지막으로 사용되는 계수의 크기에 따른 결과를 분석하여 영상의 화질과 하드웨어 크기 간의 최적의 타협점을 제시하였다.

Abstract

Existing image scalers generally adopt simple interpolation methods such as bilinear method to take cost-benefit, or highly complex architectures to achieve high quality resulting images. However, demands for a low power, low cost, and high performance image scaler become more important because of emerging high quality mobile contents. In this paper we propose the novel low power hardware architecture for a high quality raster scan image scaler. The proposed scaler architecture enhances the existing cubic interpolation look-up table architecture by reducing and optimizing memory access and hardware components. The input data buffer of existing image scaler is replaced with line memories to reduce the number of memory access that is critical to power consumption. The cubic interpolation formula used in existing look-up table architecture is also rearranged to reduce the number of the multipliers and look-up table size. Finally we analyze the optimized parameter sets of look-up table, which is a trade-off between quality of result image and hardware size.

Keywords: cubic interpolation, scaler, hardware implementation, LUT, look-up table

Ⅰ. 서 론

다양한 디지털 영상 기기들의 등장으로 영상의 해상도 변환 및 회전, 기하학적 왜곡의 보정 등과 같은 다양한 영상 처리를 고속으로 처리하는 시스템에 대한 요구가 증가하고 있다. 특히, 해상도 변환은 이종기기간의 호환성을 높이기 위해서 매우 중요한 기술이라고 할 수 있는데, 이것은 영상 포맷의 종류가 다양하게 존재하고 수많은 영상 기기들이 다양한 크기의 디스플레이 장치를 갖고 있기 때문이다. 따라서 디지털 TV, DVD 플레이어, 셋탑박스, 프로젝터 등 다양한 디지털 영상 기기들은 영상의 해상도 변환을 담당하는 스케일러 칩

* 학생회원, ** 정회원, 광운대학교 컴퓨터공학과
(Dept. of Computer Eng., Kwangwoon Univ.)

※ 본 논문은 교육과학기술부의 재원으로 한국연구재단의 지원(2010-0015441, 2009-0088064)과, 2008년 광운대학교 교내 학술연구비 지원에 의해 연구되었음.

접수일자: 2010년1월30일, 수정완료일: 2010년4월29일

(scaler chip)을 내장하고 있으며, 전용의 스케일러 장치들 또한 개발되고 있다.

스케일러는 임의의 영상 좌표에 해당하는 화소의 값을 추정하기 위하여 보간 기법을 이용하는 장치이다. 스케일러의 입출력 영상은 일반적으로 래스터 방식으로 이루어지며, 수직 방향 보간을 위해 버퍼와 라인메모리를 사용하고 있다[2, 4~12]. 따라서 스케일러에서 사용하는 보간 기법은 결과 영상의 화질과 전체 하드웨어 크기에 큰 영향을 주므로, 보간 기법의 선택이 매우 중요하다. 실시간 처리를 요하는 스케일러에서는 양선형 보간(bilinear)이나 인접 화소 보간(nearest neighbor)과 같이 비교적 연산량이 적은 보간 기법들이 많이 이용되고 있으나[1~2], 이런 방법들은 결과 영상의 화질이 다소 떨어진다는 단점이 있다. 이런 이유로 최근에는 평탄한 영역과 경계 영역 모두에서 우수한 품질을 보여주는 3차 보간(cubic interpolation) 기법의 적용이 고려되고 있으며, 3차 식 계산에 의한 많은 연산량을 해결하고 실시간 스케일러에 적용하기 위한 연구들이 진행되고 있다[3~8].

현재까지 제안되어오고 있는 3차 보간을 적용한 스케일러로는, 첫 번째로 보간 수식을 덧셈기와 곱셈기 등을 이용해 직접적으로 구현하되, 병렬처리 및 파이프라인(pipeline) 기술을 이용해 실시간 해상도 변환이 가능하도록 한 구조[3~4]가 있으며, 두 번째로 보간 수식 및 스케일러 내부에서 필요한 복잡한 계산 중 일부를 미리 계산하여 저장한 룩업테이블을 이용하는 구조[5]가 있다. 세 번째로 3차 수식을 1차 수식으로 근사화해 3차 보간을 적용하는 기법[6~7]이 있다. 첫 번째 구조는 기본적으로 곱셈기의 수가 많이 필요하며, 병렬 성을 크게 할수록 필요한 곱셈기의 수도 증가해 하드웨어의 크기가 크게 증가한다는 단점이 있다. 두 번째 구조는 3차 보간뿐만 아니라 여러 가지 보간의 적용[1, 9~10]을 위해 빈번하게 사용되는 방법이지만, 호환하는 해상도의 종류와 정밀도가 제한되며, 사용하는 정확도에 따라 룩업테이블의 크기가 커지는 단점이 있다. 세 번째 구조는 정밀도를 높이기 위해 근사에 사용하는 일차 함수의 구간을 세분화 할수록 곱셈기 및 덧셈기의 사용이 많아져 하드웨어 복잡도가 높아진다는 단점이 있다. 또한, 계수의 정밀도와 결과영상의 화질에 대한 분석도 이루어지고 있지 않다. 1차 식 근사화 방법과 룩업테이블을 접목한 구조[7]의 경우도 계수를 단순히 룩업테이블로 구현한 구조에 비하여 더 많은 곱셈기를 사용하게 된다.

본 논문에서는 여러 가지 표준 영상들 사이에서 해상도 변환이 가능한 호환성 높은 실시간 래스터 스캔 방식의 스케일러 구조를 제안한다. 고려하는 최대 해상도는 QXGA(2048x1536)로 초당 약 1억 화소(30frames/sec)에 대한 처리가 가능한 구조를 목표로 하였으며, 해상도의 축소는 확대에 비하여 화질의 열화가 적게 나타날 뿐만 아니라, 영상의 확대와 축소는 모두 보간을 이용하는 같은 원리로 동작하기 때문에, 본 논문에서는 스케일러의 확대 기능에 대해서만 다룬다. 보간 알고리즘으로는 이미 잘 알려진 3차 보간 기법을 적용하였으며, 하드웨어 복잡도 및 연산량 해결을 위해서 룩업테이블 구조를 기반으로 성능을 개선하였다. 기본적인 스케일러 구조에서 메모리 접근에 의한 소비전력의 감축을 위하여 버퍼 대신 라인메모리를 추가로 사용하는 메모리 부의 구조를 제안 및 적용하였으며, 3차 보간 수식의 재정리를 통해 보간 부에서 사용되는 곱셈기의 수와 룩업테이블의 크기를 최소화하는 방법 및 구조에 대하여 제안하였다. 또한, 사용되는 계수의 양자화 레벨과 정확도에 따른 화질과 하드웨어의 크기를 실험적으로 분석하였고, 이를 통해 스케일러 결과 영상의 품질과 하드웨어 비용 사이의 타협점을 찾음으로써, 최적화된 룩업테이블의 계수 구성을 소개한다.

본 논문의 구성은 Ⅱ장의 서두에서 기존 연구에 대해 기술하고, 저전력화 달성을 위한 새로운 메모리 부 및 하드웨어 소형화를 위한 새로운 보간 부의 구조를 제안한다. 또한 제안하는 구조에 따른 최적화된 룩업테이블의 구성을 제안하고, 룩업테이블 최적화를 위한 분석을 제시한다. Ⅲ장에서는 실험결과 및 비교 대상이 되는 구조와의 비교를 보이며, 최종적으로 Ⅳ장에서는 결론을 맺는다.

Ⅱ. 본 론

1. 배경지식

보간 함수의 표현 방법에 있어서 많은 보간 방법들을 (1)과 같이 콘볼루션(convolution)과 유사한 형태의 수식으로 나타낼 수 있다[13].

$$g(x) = \sum_k c_k u\left(\frac{x - x_k}{h}\right) \quad (1)$$

식 (1)에서 x는 보간을 수행할 점의 좌표, x_k는 보간에 이용하는 점의 좌표, h는 샘플링(sampling) 증분, c_k

는 x_k에서의 값, u는 보간 커널(kernel), $f(x)$는 x 좌표에서의 보간 값을 나타낸다. 식 (1)에서 보간 커널 u를 조정하여 여러 가지 보간 방법을 적용할 수 있다. 3차 보간을 위한 보간 커널 u는 식 (2)와 같다[13].

$$u(s)=\begin{cases}(a+2)|s|^3-(a+3)|s|^2+1, & 0\le|s|<1\\ a|s|^3-5a|s|^2+8a|s|, & 1\le|s|<2\\ 0, & elsewhere\end{cases} \tag{2}$$

식 (2)에서 상수 a의 값에 따라서 3차 보간의 특성이 바뀌게 된다. 최적의 a값을 결정하기 위한 많은 방법들이 제안되어 왔으며, a=-1/2인 커널을 이용하면 영상의 특성에 독립적으로 안정적인 결과를 얻을 수 있다[14]. 식 (1), (2)와 a=-1/2을 이용하여 3차 보간 수식을 정리한 후, 보간에 이용하는 네 개의 점과 그 점에 곱해지는 계수에 대해 정리하면 식 (3)과 같다. 룩업테이블을 이용하는 일반적인 스케일러에서는, 식 (3)에서 $f(x_n)$과 곱해지는 s항으로 이루어진 네 가지 수식의 결과를 룩업테이블에 저장한다. 해당 방법에 대해서는 2.3절의 '가' 항에서 보다 자세히 설명한다.

$$\begin{aligned}f(x)=&\left[-\frac{1}{2}s^3+s^2-\frac{1}{2}\right]f(x_{k-1})\\&+\left[\frac{3}{2}s^3-\frac{5}{2}s^2+1\right]f(x_k)\\&+\left[-\frac{3}{2}s^3+2s^2+\frac{1}{2}\right]f(x_{k+1})\\&+\left[\frac{1}{2}s^3-\frac{1}{2}s^2\right]f(x_{k+2})\end{aligned} \tag{3}$$

스케일러에서 식 (3)과 같은 3차 보간을 덧셈기 및 곱셈기를 이용해 구현할 경우, 연산량이 매우 많고 하드웨어 복잡도가 매우 크다는 단점이 있다. 따라서 연산량과 하드웨어 복잡도를 해결하기 위한 다양한 스케일러 구조가 제안되어 오고 있다. 영상을 여러 개의 블록으로 나누고, 각각의 블록에 대해서 파이프라인을 이용해 병렬적으로 보간을 수행하는 스케일러 구조[3~4]는 연산 속도를 높일 수 있지만, 파이프라인의 적용과 여러 개의 보간 부를 사용함으로써 하드웨어 복잡도가 높다는 단점이 있다. 그 예로, Arias Estrada가 제안한 스케일러는 최소 20개의 곱셈기와 12개의 덧셈기를 필요로 한다. 영상의 스케일 비율에 따른 좌표 변환 정보를 저장하고 있는 룩업테이블을 이용한 구조[5]는, 스케일링 과정에서 좌표 계산 시에 발생하는 연산량과 양자화 오류를 낮추지만, 호환하는 해상도의 종류 수에 따라 룩업테이블의 크기가 매우 커져, 호환성 높은 스케일러에 적용하기에는 무리가 있다. 계수 정보를 룩업테이블로 저장하는 구조는 연산량과 하드웨어 복잡도를 어느 정도 해결할 수 있으나, 계수와 화소 값의 곱을 위해 최소 4개의 곱셈기를 사용하므로, 여전히 하드웨어 복잡도에 대한 문제가 남아 있다[7].

일반적으로 스케일러의 하드웨어 구현 시, 영상을 저장하기 위한 프레임메모리를 이용하거나, 몇 개의 라인메모리와 정보의 손실을 방지하기 위한 입력 버퍼를 이용한다. 프레임메모리를 이용하면 구현이 쉽다는 장점이 있지만, 많은 메모리 공간을 필요로 하기 때문에, 대부분의 스케일러들은 프레임메모리 보다 적은 공간의 입력 버퍼와 라인메모리를 이용하는 구조[2, 6~7, 9, 11~12]로 되어 있다. 이 때, 라인메모리는 수직 방향 보간을 위해 사용되므로, 보간에 사용하는 필터의 수직 방향 크기와 동일한 수의 라인메모리를 사용하게 된다. 따라서 3차 보간을 적용하는 스케일러는 일반적으로 4개의 라인메모리와 1개의 입력 버퍼를 포함하는 구조로 구성된다. 라인메모리를 이용하면 수직 방향 보간의 적용이 단순해진다는 장점이 있지만, 라인메모리에 저장된 모든 입력 화소의 데이터가 다른 라인메모리로 복사되는 연산이 필요하게 된다. 즉, 3차 보간의 적용 시, 라인메모리에서는 모든 입력 화소에 대하여 총 4번의 쓰기 연산이 수행되어야 한다. 이에 따라, 메모리에서 발생하는 쓰기 연산이 많아지고, 전체 하드웨어의 전력 소모가 커진다는 단점이 있다.

2. 기존의 스케일러 구조와 제안하는 스케일러의 메모리 부 하드웨어 구조

본 절에서는 일반적인 룩업테이블을 사용하는 스케일러의 전체 구조 및 메모리 부의 구조에 대해 설명하

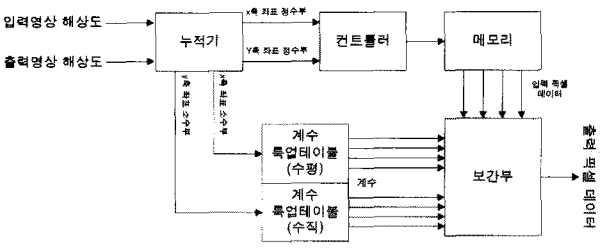


그림 1. 룩업테이블을 이용하는 스케일러의 일반적 구성

Fig. 1. An architecture for a scaler using look-up tables.

고, 저전력 스케일러 설계를 위한 메모리 부의 새로운 구조를 제안한다.

<그림 1>은 본 논문에서도 적용하고 있는 일반적인 룩업테이블을 이용하는 스케일러의 구조이다. 입력 영상의 해상도와 출력 영상의 해상도 비에 따라 계산한 출력영상에서의 각 화소간 거리는 x축 및 y축 방향으로 누적기를 통해 누적되고, 이를 통해 보간을 수행해야 할 화소의 좌표가 계산되어 누적기로부터 출력된다. 좌표는 정수부와 소수부로 나누어지고, 정수부는 컨트롤러로 입력되며 소수부는 룩업테이블로 입력된다. 컨트롤러는 입력 받은 좌표의 정수부를 이용하여 라인 메모리에 컨트롤 신호를 내보내고, 이에 따라 보간에 이용할 입력 영상의 화소 정보들이 보간 부로 입력된다. 룩업테이블에서는 x축과 y축 좌표의 소수부를 주소로 입력 받아 수평 및 수직 보간 시 이용되는 계수를 보간 부로 출력한다. 마지막으로, 보간 부에서는 입력된 계수와 입력 영상의 정보를 이용하여 보간을 수행한 최종 결과를 출력한다.

가. 기존의 버퍼를 구성하는 메모리 부의 구조

래스터 방식으로 영상을 입력 받는 스케일러의 메모리 부는 구현의 편의를 위해 프레임메모리를 이용하는 경우[3~5, 8, 10]도 있지만, 일반적으로는 메모리 공간의 절약을 위해 보다 적은 크기의 입력 버퍼와 입력 화소 데이터를 차례대로 출력하는 라인메모리로 구성된다. 라인메모리의 수는 수직 방향 보간 수행 시 필요한 화소 수와 같으며, 3차 보간의 경우 네 개의 화소 데이터를 이용해 보간을 수행하므로, <그림 2>와 같이 네 개의 라인메모리가 필요하다.

<그림 2>와 같은 구조를 이용하면, 입력 받은 후 가장 오래된 데이터가 가장 하단의 라인메모리에 위치하게 되어 수직 방향으로 보간을 수행하는 구조의 설계가 용이하게 되는 장점이 있다. 그러나 매 클록 사이클(clock cycle) 마다 모든 라인메모리 및 버퍼에서 쓰기 연산이 수행되어, 많은 수의 쓰기 연산에 의해 전력 소모가 커지는 단점이 있다.

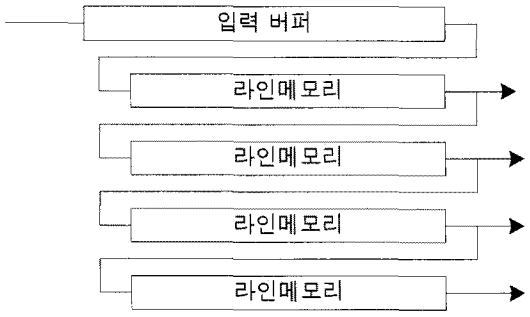


그림 2. 버퍼를 사용하는 일반적인 메모리 부 구성
Fig. 2. Conventional memory block having an input buffer.

나. 메모리 접근 수를 줄인 제안하는 메모리 부의 구조

<그림 3>은 제안하는 스케일러의 메모리 부 구조를 나타낸다. 제안하는 구조에서는 <그림 2>에서 라인메모리와는 별도로 구성하였던 버퍼를 사용하지 않고, 버퍼의 기능을 대신하기 위하여 추가적인 두 개의 라인메모리를 사용한다. <그림 2>에서는 각 라인메모리의 출력이 다른 라인메모리의 입력으로 연결되어 있는 반면, 제안하는 구조에서는 모든 라인메모리가 동일한 입력 데이터와 연결되어 있고, 해당 데이터를 저장해야 할 라인메모리에서만 쓰기 연산이 수행되도록 되어 있다. 입력 영상의 수평선 한 줄을 하나의 라인메모리에 채우고 나면, 두 번째 수평선 한 줄을 다음 라인메모리에 채우도록 되어 있어서, 각 라인메모리로부터 출력되는 데이터를 이용해 수직 방향으로 보간을 수행할 수 있다. 이 때, 여섯 개의 라인메모리로부터 출력되는 데이터 중 보간에 필요한 네 개의 데이터만을 선택하기 위하여 네 개의 2-to-1 먹스(MUX)가 이용되며, 먹스의 선택 신호는 전체 시스템을 제어하고 있는 컨트롤러로부터 입력 받는다.

제안하는 구조의 메모리 부는 매 클록 사이클 마다 한 개의 라인메모리에서만 쓰기 연산이 수행되도록 설계되어 있어서, 각 메모리에서 발생하는 쓰기 연산의 총 횟수가 <그림 2>의 구조에 비하여 1/4배로 줄어들게 된다. 따라서 제안하는 구조의 메모리 부는 기존의

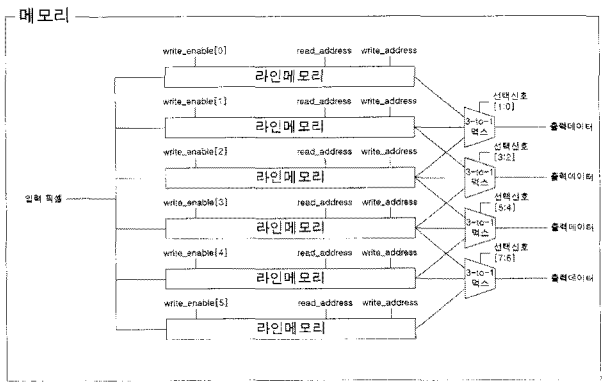


그림 3. 제안하는 메모리 부의 구조
Fig. 3. The architecture of the proposed memory block.

방법에 비하여 전력 소모의 측면에서 이점이 있다. 또한, 제안하는 메모리 부는 각각의 라인 메모리에서 쓰기 연산과 읽기 연산이 동시에 일어나지 않아, 싱글 포트(single port) 메모리를 이용한 라인메모리 구성이 가능하며, 싱글 포트 메모리 이용 시 전력 소모를 줄일 수 있다는 장점이 있다.

<그림 2> 구조의 메모리 부로부터 출력되는 네 개의 데이터는 스케일러에 입력된 시간의 순서로 정렬되는 반면, <그림 3>의 구조로부터 출력되는 데이터는 정렬되지 않아 기존의 일반적인 보간 구조로는 보간이 불가능해진다. 따라서 보간 부에는 입력 받은 데이터를 재정렬하는 회로가 포함되어야 하며, 이에 대해서는 2.1절의 '가' 항에서 다룬다.

3. 제안된 룩업테이블 구성 방법과 이를 적용한 보간 부의 구조

본 절에서는 식 (3)의 3차 보간 기법을 적용한 스케일러에서 일반적으로 사용되는 계수 룩업테이블의 구성을 보이고, 식 (3)의 재정리를 통해 크기가 더 작은 룩업테이블을 구성하는 방법과 새롭게 정의된 룩업테이블을 사용하기 위한 새로운 보간 부의 구조를 제안한다. 또한, 일반적인 스케일러가 적용하는 룩업테이블을 양자화 오류에 대하여 분석하여 룩업테이블이 갖는 양자화 레벨의 적절한 타협점을 제시하고, 이에 따라 최적화된 룩업테이블의 구성을 제안한다.

가. 기존의 룩업테이블의 구성과 보간 부의 구조

일반적으로 룩업테이블에는 식 (3)에서 s항으로 정리되는 계수를 계산한 값이 저장된다. 즉, s 값에 따라 변하는 4개의 계수가 계산되어 저장된다. 따라서 s 값을 입력으로 받아 네 개의 계수를 출력으로 내보내는 룩업테이블을 구성해야 하며, 룩업테이블에 저장되는 각각의 계수를 계산하기 위한 수식은 식 (4)와 같다.

$$
\begin{aligned}
coeff_{-1}(s) &= -\frac{1}{2}s^3 + s^2 - \frac{1}{2}s \\
coeff_0(s) &= \frac{3}{2}s^3 - \frac{5}{2}s^2 + 1 \\
coeff_1(s) &= -\frac{3}{2}s^3 + 2s^2 + \frac{1}{2}s \\
coeff_2(s) &= \frac{1}{2}s^3 - \frac{1}{2}s^2
\end{aligned} \tag{4}
$$

식 (4)에서 각 계수의 합은 1이지만, 계수를 룩업테이블에 저장할 때 발생하는 양자화 오류로 인해, 실제 룩업테이블에 저장되는 계수의 합은 1이 되는 조건을 만족하지 못한다. 따라서 계수 중 $coeff_1(s)$의 값은 식 (5)를 이용해 저장하여, 계수의 합이 1이 되도록 조정한다.

$$
\begin{aligned}
coeff_1(s) &= 1 - \{coeff_{-1}(s) \\
&\quad + coeff_0(s) + coeff_2(s)\} \qquad (5) \\
coeff_{-1}(s) &= \left(-\frac{1}{2}s^3 + s^2 - \frac{1}{2}s\right) \times 2^n \\
coeff_0(s) &= \left(\frac{3}{2}s^3 - \frac{5}{2}s^2 + 1\right) \times 2^n \\
coeff_1(s) &= 2^n - \{coeff_{-1}(s) \\
&\quad + coeff_0(s) + coeff_2(s)\} \\
coeff_2(s) &= \left(\frac{1}{2}s^3 - \frac{1}{2}s^2\right) \times 2^n \qquad (6)
\end{aligned}
$$

보간 부에서의 계산은 고정소수점 방식을 이용하므로, 룩업테이블에 저장되는 계수도 고정소수점 방식으로 표현된다. 이 때 각 계수를 저장하기 위해 사용하는 비트 수를 n이라고 한다면 실제로 룩업테이블에 저장되는 값은 식 (6)과 같이 계산된다.

<그림 4>는 계수 룩업테이블을 이용하여 보간을 수행하는 기존의 보간 부[8]에 대하여, <그림 3>과 같은 제안하는 메모리 부의 적용을 위해, 필요한 회로를 추가한 것이다. 왼쪽 네 개의 4-to-1 먹스는 제안하는 메모리 부로부터 입력 받은 데이터를 재 정렬하기 위해 사용되며, 먹스의 선택 신호는 컨트롤러로부터 입력 받는다. 먹스에 의해 정렬된 데이터는 룩업테이블로부터 입력 받은 계수와 함께 수직 방향 보간에 이용되며, 수평 방향으로 다시 한 번 보간을 수행하고 컨버터를 거치고 나면 최종적으로 보간이 수행된 출력 값을 얻을 수 있다. 컨버터에서는 오버플로(overflow) 에러 수정과 출력 비트 수 조정이 수행된다. <그림 4>의 2-to-1 먹스는 특정 위치의 화소 값을 바이패스(bypass)하여, 그대로 보간 결과로 사용하기 위해 사용된다.

<표 1>은 본 논문에서 제안하는 타협점을 적용한 계수 룩업테이블의 구성을 보여준다. 본 논문에서는 룩업테이블의 양자화 레벨에 대해서는 32단계, 각 요소의 정확도에 대해서는 $coeff_0(s)$와 $coeff_1(s)$를 위해 7-비트, $coeff_{-1}(s)$와 $coeff_2(s)$를 위해 4-비트를 사용하여, 총 22-비트의 정확도를 사용할 것을 타협점으로 제시한다. 타협점의 근거는 2.3절의 '라' 항과 3장 '실험'을 통해 확인할 수 있다. 특이한 것은 s가 0일 때의 $coeff_0(s)$의 값이

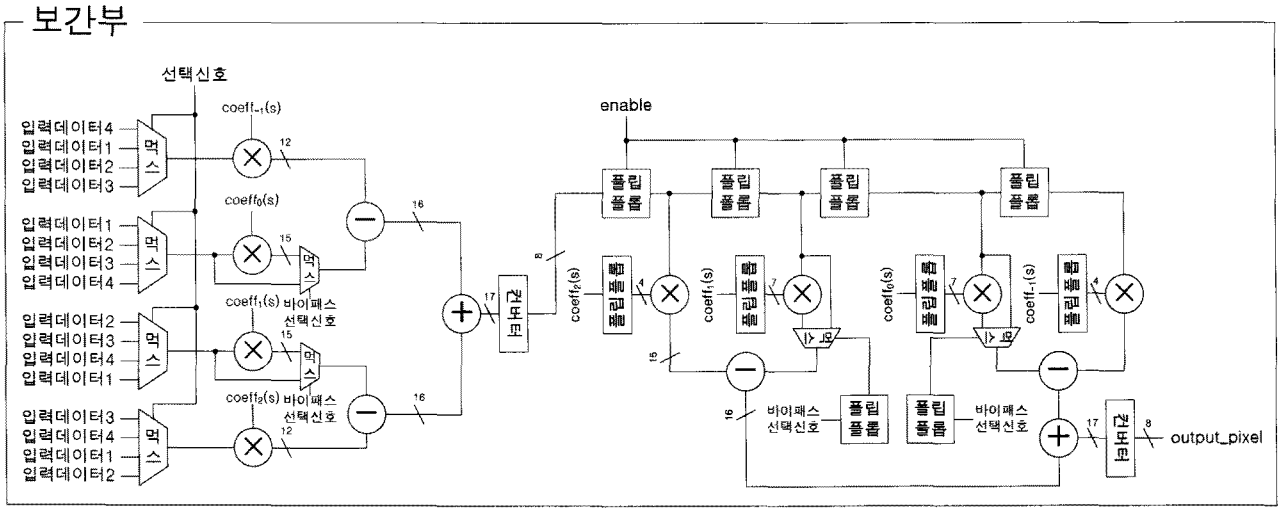


그림 4. 기존의 보간 부의 필터구조
Fig. 4. The conventional interpolation filter block.

표 1. 최적화된 계수 룩업테이블
Table 1. Optimized coefficients in the look-up table.

s	4-비트 $coeff_{-1}(s)$	7-비트 $coeff_0(s)$	7-비트 $coeff_1(s)$	4-비트 $coeff_2(s)$
0	0	0	0	0
1	-1	127	2	0
2	-3	126	5	0
3	-4	125	7	0
⋮	⋮			
28	0	11	123	-6
29	0	8	124	-4
30	0	4	127	-3
31	0	2	127	-1

128이 아닌 0이라는 사실인데, 이것은 128을 저장하여 1-비트를 추가적으로 사용하는 것 대신, 보간 부에서 바이패스를 이용함으로써, 룩업테이블의 크기가 늘어나는 것을 막기 위함이다.

나. 곱셈기의 수를 줄이기 위한 룩업테이블의 구성

본 항에서는 식 (3)을 수정하고, 룩업테이블의 구성을 바꿔 곱셈기의 수를 줄이는 방법을 제안한다.

식 (3)은 화소의 값과 계수 항에 대하여 정리되어 있는데, 식을 전개하여 다시 정리하면, 식 (7)과 같이 s 항에 대하여 나타낼 수 있다.

$$f(x) = [1/2\{-f(x_{k-1}) + f(x_{k+2})\} + 3/2\{f(x_k) - f(x_{k+1})\}]s^3 + [f(x_{k-1}) - 2\{f(x_k) - f(x_{k+1})\} - 1/2\{f(x_k) + f(x_{k+2})\}]s^2 + [-1/2\{f(x_{k-1}) - f(x_{k+1})\}]s + f(x_k) \quad (7)$$

식 (7)을 이용하여 보간 과정을 구현하기 위해서는 s^2, s^3을 미리 계산하여 저장하고 있는 룩업테이블을 이용할 수 있으며, 룩업테이블 내의 각 요소를 위한 정확도로는 Ⅲ장 '실험'을 통해 6-비트를 사용할 것을 제안한다.

<표 2>는 제안하는 룩업테이블의 실제 구성 값을 나타낸다. <표 1>의 룩업테이블은 88 Bytes의 크기를 갖는 반면 <표 2>와 같이 제안하는 방법을 통해 룩업테이블을 구성하면 48 Bytes의 작은 크기로 줄일 수 있다.

표 2. 제안하는 s^2, s^3 항을 위한 룩업테이블
Table 2. Proposed look-up table for s^2, s^3 terms.

s	6-비트 s^2	6-비트 s^3	s	6-비트 s^2	6-비트 s^3
0	0	0	⋮	⋮	
1	0	0			
2	0	0	28	49	42
3	0	0	29	52	47
⋮	⋮		30	56	52
			31	60	58

다. 곱셈기를 줄인 보간 부의 구조

<그림 5>는 2.3절의 '나' 항에서 다룬 내용을 하드웨어로 적용했을 때의 보간 부의 일부분으로서, 수직 방향 보간을 위한 회로도의 구성을 나타낸 것이다. 수평 방향 보간에 대한 구조는 수직 방향 보간에 대한 구조와 동일하므로 생략하였다.

<그림 5>를 통해 제안하는 구조는 수직 및 수평의 한쪽 방향 보간에 있어서 <그림 4>의 구조에 비하여 1개의 곱셈기를 줄이고, 8개의 덧셈기 및 뺄셈기를 추가적으로 사용하는 것을 알 수 있다. 또한, <표 2>의 0번째 양자화 레벨의 출력인 '0' 값은 <표 1>에서와는 다르게, 모두 실제로 이용되는 값이므로 바이패스 결정을 위해 필요했던 컨트롤에 대한 하드웨어 자원을 줄일 수 있는 장점이 있다.

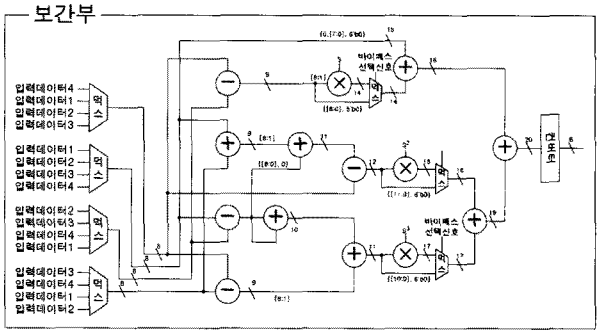


그림 5. 제안하는 곱셈기를 줄인 보간 부의 구조
Fig. 5. The proposed interpolation block with decreased multipliers.

라. 스케일러 내부의 양자화 오류와 룩업테이블의 양자화 레벨에 대한 분석

스케일러 내부에서 해상도 변환 과정을 수행할 때, 각 수치를 표현하기 위한 비트 수는 유한하게 한정되므로, 여러 번의 양자화 오류가 발생하게 된다. 첫 번째로 입력 및 출력 영상의 해상도를 입력 받아 출력 영상의 화소 간 거리 및 좌표를 구하는 과정에서, 좌표를 표현하기 위해 사용하는 비트 수에 따라 양자화 오류가 발생하게 된다. 두 번째로 룩업테이블의 양자화 레벨에 따라 좌표의 소수부에 대한 양자화 오류가 발생한다. 마지막으로 룩업테이블 각 요소의 정확도에 따라, 룩업테이블의 계수 값을 이용하는 보간 부에서 양자화 오류가 발생하게 된다.

본 논문에서는 양자화 오류가 발생하는 단계의 적절한 비트 수를 결정하기 위해, 컴퓨터에서 사용되는 표준 해상도인 QVGA~QXGA에 대해서만 고려하였다.

표 3. 소수부의 정확도에 따른 좌표 계산 시의 누적 양자화 오류(QXGA)
Table 3. Accumulated quantization error on coordination calculation(QXGA).

소수부를 위한 비트 수	누적 양자화 오류
11-비트	$1-2^{-11}$
12-비트	$0.5-2^{-11}$
13-비트	$0.25-2^{-11}$
14-비트	$0.125-2^{-11}$
15-비트	$0.0625-2^{-11}$

QXGA 보다 더 큰 해상도 표준이 존재하지만, 현재는 잘 사용되고 있지 않아 배제하였으며, TV에서 사용하는 표준 해상도 등 다른 종류의 해상도가 존재하지만, 결과 영상의 화질은 입출력 영상의 종류 보다는 입력 영상을 확대하는 비율에 높게 영향을 받으므로, 컴퓨터 표준 해상도에 대해서만 고려하였다.

<표 3>은 임의의 영상에서 QXGA 영상으로 확대 시, 소수부에 사용하는 정확도에 따른 누적 양자화 오류의 최댓값을 보여주며, 각각의 값은 사용하는 정확도에 따라 소수부의 양자화 오류가 정수부의 값에 영향을 미칠 확률이기도 하다. 따라서 본 논문에서는 좌표 값의 정수부에 오류가 나타날 확률을 7% 미만으로 떨어뜨리는 15-비트의 정확도를 사용할 것을 타협점으로 제시한다.

좌표 계산 시 사용하는 정수부의 비트 수는 입력 영상의 크기를 표현할 수 있으면 충분하다. QXGA의 경우 11-비트를 이용하면 충분하지만, 미래에 빈번하게 사용될 더 큰 영상들인 4K, HXGA, HUXGA 등에 대한 호환성을 위해, 본 논문에서는 13-비트를 타협점으로 제시한다. 최종적으로 출력 영상의 좌표를 표현하기 위해 정수와 소수부를 합쳐 28-비트 정도를 사용하는 것이 타당하다.

<표 4>는 영상의 해상도 변환 비율에 따라 변하는 출력영상의 두 화소간 거리를 일부 경우에 대해서 정리한 것이다. 본 논문에서는 영상의 확대만을 고려하므로 좌측 하단 부분은 비어있다. <표 4>의 값들 대부분이 m과 n을 자연수로 하는 $m/2^n$의 형태로 표현될 수 있으며, QVGA~UXGA를 모두 고려했을 때, 두 화소간 거리와 룩업테이블의 양자화 레벨을 $128(=2^7)$개로 나누면, 룩업테이블을 사용함으로써 발생하는 대부분의 양자화 오류를 제거할 수 있다. 또한 실험을 통해서는 QXGA를 최대 해상도로 고려했을 때, 룩업테이블의 양자화

표 4. 각 영상의 해상도 변환에 따른 출력영상의 두 화소간 거리
Table 4. Distance between two output pixels according to scale factors.

	Width / Height							
	320	1/2	2/5	2/5	5/16	1/4	1/5	5/32
	240	1/2	1/2	2/5	5/16	15/64	1/5	5/32
	640		4/5	4/5	5/8	1/2	2/5	5/16
	480		1	4/5	5/8	15/32	2/5	5/16
	800			1	25/32	5/8	1/2	25/64
	480			4/5	5/8	15/32	2/5	5/16

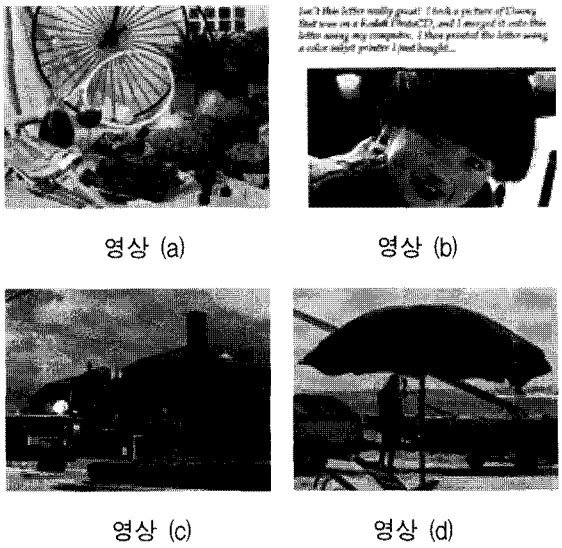
레벨을 32(=2^5)개 정도로 나누어도, 그 이상으로 나누었을 때에 비하여 화질의 차이가 눈에 띄지 않는 것을 확인하였다. 룩업테이블의 양자화 레벨 수 및 양자화 오류에 따른 실험 결과는 3.2절의 <표 6>과 <그림 7>을 통해 확인할 수 있다.

룩업테이블의 양자화 레벨 수를 많이 나누면 나눌수록 양자화 오류가 줄어들어 결과 영상의 화질이 더 좋아지는 장점이 있다. 그러나 룩업테이블의 양자화 레벨을 많이 나누면, 룩업테이블을 저장하기 위한 메모리 공간이 더 많이 필요하다는 단점이 있다. 사용하는 양자화 레벨 수와 정확도에 따른 룩업테이블의 크기 변화를 3.3절에 표시하였다. 확대하는 비율이 높을수록 결과 영상의 화질 열화가 눈에 띄게 나타나기 때문에 룩업테이블의 양자화 레벨 수도 늘릴 필요가 있다. 3.2절의 실험을 통해 본 논문에서 고려하는 해상도에 대해서는 32단계 양자화 레벨의 룩업테이블을 사용하면 더 많은 단계를 사용했을 때와 눈에 띄는 차이가 나타나지 않는 결과 영상을 획득할 수 있음을 확인하였다.

III. 실 험

1. 실험 방법 및 실험영상

실험에 이용한 원본 영상은 <그림 6>과 같이 모양과 크기가 다른 네 장의 영상이다. 실험을 위해 원본 영상을 특정 크기의 영상으로 축소시켜 스케일러의 입력으로 사용할 영상을 생성하였고, 스케일러를 통해 입력 영상을 다시 원본 영상 크기로 확대하여 출력 영상과 원본 영상 사이의 화질을 비교해 보았다. 각 원본 영상



영상 (a) 영상 (b)

영상 (c) 영상 (d)

그림 6. 실험에 이용한 원본 영상
Fig. 6. The original test scenes.

표 5. 실험한 해상도 변환
Table 5. Resolution conversion pairs in experiments.

영상 (a)	QVGA(320x240) → QXGA(2048x1536)	영상 (b)	VGA(640x480) → SVGA(800x600)
영상 (c)	VGA(640x480) → UXGA(1600x1200)	영상 (d)	XGA(1024x768) → UXGA(1600x1200)

에 대해서 실험한 해상도 변환은 <표 5>와 같다.

실험의 구체적인 과정은 다음과 같다. 입력 영상을 생성할 때 영상의 축소 시 발생하는 에일리어싱(aliasing)을 최소화하기 위하여, 원본 영상에 가우시안(gaussian)을 이용한 저역 통과 필터 처리를 하였다. 저역 통과 필터 처리는 주파수 영역에서 수행하였고, 이용한 가우시안 수식은 (8)과 같다.

$$H(u,v) = e^{-D^2(u,v)/2D_0^2} \tag{8}$$

필터 처리 후에는 영상을 다시 공간 영역으로 변환한 후 서브샘플링(subsampling)을 통해 영상을 축소시켰으며, 축소 영상의 좌표에 소수점이 포함되어 값을 알 수 없는 화소에 대해서는 3차 보간을 통해 그 값을 추정하였다.

2. 실험결과

가. 계산의 정확도에 따른 결과

2.3절에서 설명하였던 룩업테이블의 양자화 레벨 단계와 정확도에 따른 결과를 실험하였다. 누적기로부터

표 6. 양자화 오류와 관련된 각 실험에 따른 PSNR
Table 6. PSNR of result images according to LUT organizations.

구분		영상 (a) QVGA⇒ QXGA	영상 (b) VGA⇒ SVGA	영상 (c) VGA⇒ UXGA	영상 (d) XGA⇒ UXGA
양자화 레벨	16단계	19.70	24.13	29.45	37.51
	32단계	19.73	24.07	29.37	37.21
정확도 (계수)	14-비트 (2+5+5+2)	19.66	23.86	29.19	37.09
	18-비트 (3+6+6+3)	19.70	23.98	29.30	37.26
	22-비트 (4+7+7+4)	19.71	24.05	29.37	37.33
정확도 (s^2, s^3)	10(5+5)	19.75	24.11	29.39	37.79
	12(6+6)	19.74	24.11	29.44	37.54

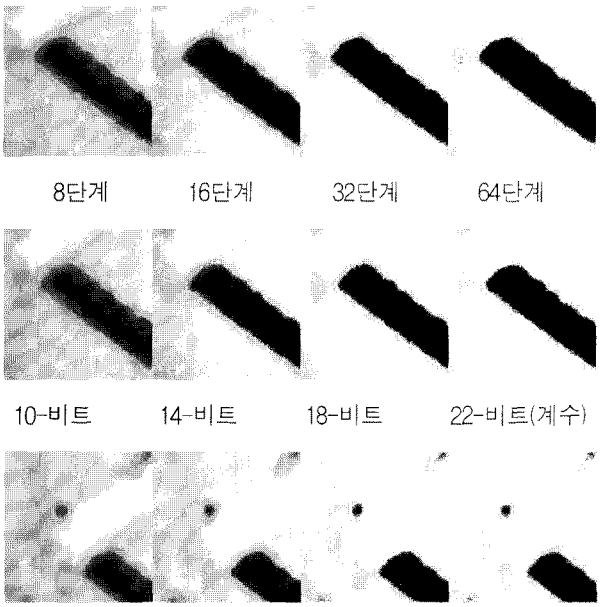


그림 7. 룩업테이블의 구성과 정밀도에 따른 결과 영상
Fig. 7. Result images according to various LUT organizations and precisions.

출력되는 좌표는 논문에서 제안한 28-비트를 이용하였으며, 룩업테이블의 양자화 레벨 수에 대한 실험에서는 좌표의 소수부에 대한 양자화 오류가 결과에 미치는 영향만을 보기 위하여, 계수 및 보간과 관련된 계산에 대해 배정도수 부동소수점(double precision floating point)을 사용하였다. 룩업테이블의 정확도에 대한 실험은 계수 룩업테이블과 s^2, s^3 항을 위한 룩업테이블에 대해서 각각 따로 진행하였으며, 룩업테이블의 양자화 레벨 수는 논문에서 제안하는 32단계로 나누어 실험을 진행하였다. <그림 6>의 영상을 이용하여 각 실험에 대한 PSNR을 정리한 것이 <표 6>이다.

<표 6>을 보면 좌표의 양자화 레벨 수가 PSNR에 큰 영향을 미치지 않음을 알 수 있다. 따라서 주관적 화질 측정을 위하여, 양자화 레벨 수와 정확도에 따라 눈에 띄는 차이가 나타나는 영상을 <그림 7>에 나타내었다. <그림 7>의 영상은 실험에서 가장 확대 비율이 높은 <그림 6>의 영상 (a)에 대한 결과이다.

<그림 7>의 결과를 통해 룩업테이블의 양자화 레벨 단계로는 16단계, 계수 항을 위한 룩업테이블의 정확도로는 18-비트, s^2과 s^3항을 위한 룩업테이블의 정확도로는 10-비트 이상을 사용하면 결과 영상의 화질에 큰 차이가 없음을 알 수 있다. 그러나 논문에서 고려하지 않은 더 큰 해상도를 갖는 영상에 대한 고려와 출력 영상의 안정적인 품질을 위하여, 32단계의 양자화 레벨과 22-비트 정확도를 갖는 계수 룩업테이블, 또는 12-비트 정확도를 갖는 s^2, s^3 룩업테이블을 사용할 것을 제안한다.

나. 사용하는 보간 부의 하드웨어 구조에 따른 결과 비교

<그림 8>은 <그림 4>와 같이 일반적인 보간 부 구조를 갖는 스케일러의 시뮬레이션 결과를 보여주며, <그림 9>는 <그림 5>와 같이 제안하는 보간 부 구조를 갖는 스케일러의 시뮬레이션 결과를 보여준다.

각각의 시뮬레이션은 gate level에서 수행하였으며, VGA(640×480) 영상을 XGA(1024×768)으로 확대하는

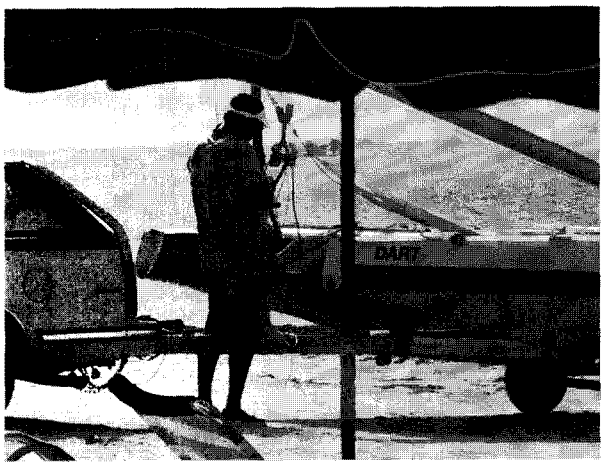


그림 8. 기존의 구조의 시뮬레이션 결과
Fig. 8. The simulation result of existing architecture.

그림 9. 제안하는 구조의 시뮬레이션 결과
Fig. 9. The simulation result of the proposed architecture.

경우에 대하여 실험하였다. <그림 8>과 <그림 9>를 통해 제안하는 보간 부의 구조를 이용하면, 하드웨어 자원을 절약하면서 결과 영상의 화질이 떨어지지 않는 스케일러를 구현할 수 있음을 알 수 있다.

3. 룩업테이블 크기 계산

룩업테이블의 양자화 레벨을 2^m으로 나누고 룩업테이블 각 요소의 정확도로 사용하는 비트 수를 n-비트라고 한다면, 룩업테이블의 총 크기는 (2^m*n-비트)가 된다. 여기서 룩업테이블의 양자화 레벨 수를 m이 아닌 2^m으로 가정한 이유는, 모든 계산에는 연산 시간을 고려하여 고정소수점 방식이 이용되고 두 화소간 거리는 m-비트를 이용해 표현되기 때문이다. 먼저 <표 1>과 같은 계수를 저장하는 룩업테이블의 정확한 크기를 계산해 보기 위해서는 $coeff_0(s)$의 비트 수에 따라 변하는 $coeff_{-1}(s)$~$coeff_2(s)$의 전체 비트 수를 계산해야 한다. <표 7>은 $coeff_0(s)$의 비트 수에 따른 계수 전체 비트 수의 총 합을 나타낸다. <표 7>을 이용해 m 단계로 나누어지는 계수 룩업테이블의 전체 크기를 계산해 <표 8>에 나타내었다.

표 7. $coeff_0(s)$에 사용되는 비트 수에 따라 변하는 전체 계수의 비트 수
Table 7. The size of the whole coefficient bits according to the precision of $coeff_0(s)$

$coeff_0(s)$	6-비트	7-비트	8-비트	9-비트
전체 계수	18-비트	22-비트	26-비트	30-비트

표 8. 계수 룩업테이블의 양자화 레벨과 정확도에 따른 룩업테이블의 전체 크기
Table 8. Coefficient LUT size according to quantization levels and precisions.

	18-비트	22-비트	26-비트	30-비트
2^4=16 단계	36 Bytes	44 Bytes	52 Bytes	60 Bytes
2^5=32 단계	72 Bytes	88 Bytes	104 Bytes	120 Bytes
2^6=64 단계	144 Bytes	176 Bytes	208 Bytes	240 Bytes
2^7=128 단계	288 Bytes	352 Byes	416 Bytes	480 Bytes

표 9. s^2, s^3 룩업테이블의 양자화 레벨과 정확도에 따른 룩업테이블의 전체 크기
Table 9. LUT size for s^2, s^3 according to quantization levels and precisions.

	10-비트	12-비트	14-비트	16-비트
2^4=16 단계	20 Bytes	24 Bytes	28 Bytes	32 Bytes
2^5=32 단계	40 Bytes	48 Bytes	56 Bytes	64 Bytes
2^6=64 단계	80 Bytes	96 Bytes	112 Bytes	128 Bytes
2^7=128 단계	160 Bytes	192 Bytes	224 Bytes	256 Bytes

<표 8>을 보면 알 수 있듯이, 룩업테이블의 전체 크기는 정확도보다는 나누어지는 양자화 레벨 수에 더 크게 영향을 받는다. 3.2절 '실험결과'와 <표 8>을 통해, 2.3절의 '가' 항에서는 크기가 88 Bytes인 32단계 양자화 레벨과 22-비트 정확도를 갖는 계수 룩업테이블을 사용할 것을 타협점으로 제안하였다. 본 논문에서 최종적으로 제안하는, <표 2>와 같은 구조의 s^2, s^3 항을 위한 룩업테이블의 전체 크기를 사용하는 양자화 레벨 수와 정확도에 따라 정리하여 <표 9>에 나타내었다.

3.2절 '실험결과'의 <표 6>과 <그림 7>을 통해, 32단계의 룩업테이블을 사용할 때 12-비트 정확도를 갖는 s^2, s^3 룩업테이블을 사용하면, 22-비트 정확도를 갖는 계수 룩업테이블을 사용할 때의 결과에 비해서 PSNR 및 주관적 화질 측면에서 그 이상의 성능을 보여줌을 알 수 있다. 따라서 2.3절의 '나' 항에서는 32단계의 양자화 레벨로 나누어지며, 12-비트 정확도를 갖는 s^2, s^3 룩업테이블을 사용할 것을 타협점으로 제안하였다. 이 때의 룩업테이블 크기는 48 Bytes로, 제안하는 s^2, s^3 룩업테이블을 사용하면, 계수 룩업테이블을 사용할 때에 비하여 40 Bytes의 공간을 줄일 수 있는 장점이 있다.

4. 기존 방법과의 비교

본 절에서는 본 논문에서 제안하는 룩업테이블, 메모리 부, 보간 부를 사용하여 하드웨어를 구성했을 때와 기존에 제안된 방법으로 하드웨어를 구성했을 때의 차이를 자원과 소모 전력 측면에서 비교/분석하였다. 본 논문에서 제안하는 스케일러는 실시간 래스터 방식으로 스캔하는 구조이므로, 고정된 데이터 율로 결과 영상이 출력된다. 따라서 기존 구조와 수행시간 측면에 대해서는 비교하지 않았다.

<표 10>은 3차 보간을 이용하는 스케일러 각각의 구조에 대하여 보간 부에서 소요되는 주요 하드웨어 자원을 정리하여 나타낸 것이다. <표 10>의 (a)는 식 (3)의 3차 보간 수식을 단순히 덧셈기와 곱셈기를 이용해 구현했을 때, 보간 부와 계수발생기에서 필요한 하드웨어 자원을 나타낸다. (b)는 H. K. Chiang가 제안한 방법으로, 수평 및 수직 방향에 대해 하나의 계수 발생기를 이용하여 자원을 절약한 방법에 대해 정리한 것이다. (c)는 2.3절의 '가' 항에서 설명한 네 개의 계수에 대하여 룩업테이블을 이용할 때의 필요한 자원을 정리한 것이다. (d)는 1차식 근사화 방법을 이용할 때의 필요 하드웨어 자원을 정리한 것이다. 1차식 방법의 근사화는 세분화하는 구간의 수에 따라 하드웨어 복잡도가 달라지므로, 최명렬이 제안한 방법인, 3차식을 8구간의 다른 1차식으로 근사화 했을 때의 방법에 대하여 정리하였다. (e)는 본 논문에서 제안하는 재정리된 3차식과 s^2, s^3 룩업테이블을 사용하는 구조에 대하여 정리한 것이다. 제안하는 구조는 다른 구조에 비하여 적은 수의 곱셈기를 사용하며, 룩업테이블의 사용에 있어서도 메모리 공간을 절약할 수 있음을 알 수 있다.

표 10. 스케일러 구조 별 보간 부 및 계수 발생기에서 필요한 주요 하드웨어 자원

Table 10. Hardware resources needed in interpolation block.

	(a) 단순 3차식 구현	(b) 축소된 계수 발생기 이용[8]	(c) 계수 룩업 테이블 이용	(d) 1차식 근사화 [6]	(e) 제안하는 구조
곱셈기	12	10	8	8	6
덧셈기	20	12	6	16	22
룩업테이블 (Bytes)	0	0	160	0	96

표 11. 메모리 블록의 구조에 따른 라인메모리 접근 횟수

Table 11. The number of accesses to the line memories.

	연산	일반적인 버퍼를 사용하는 구조	제안하는 구조
영상 (a)	읽기	12,582,912	12,582,912
	쓰기	307,200	76,800
영상 (b)	읽기	1,920,000	1,920,000
	쓰기	1,228,800	307,200
영상 (c)	읽기	7,680,000	7,680,000
	쓰기	1,228,800	307,200
영상 (d)	읽기	7,680,000	7,680,000
	쓰기	3,145,728	786,432

<표 10>을 통해 제안하는 방법을 이용하여 스케일러를 구성하면 48 Bytes의 룩업테이블 만으로 곱셈기의 수를 대폭 줄일 수 있어, 하드웨어의 크기를 작게 할 수 있음을 알 수 있다.

하드웨어가 소모하는 전력에 대해서 비교해 보기 위하여, 제안하는 구조와 일반적인 라인메모리를 사용하는 구조에 대하여 라인 메모리에 대한 접근 횟수를 비교해 보았다. 2.2절의 '가' 항에서 설명한 것과 같이 일반적인 라인메모리를 사용하는 구조[4~8]에서는, 모든 화소에 대하여 모든 라인메모리에서 한 번씩 쓰기 연산을 수행하게 된다. 3차 보간을 위해서는 네 개의 라인메모리를 이용하므로, 일반적인 라인메모리를 사용하는 구조에서는 제안하는 구조에 비해서 동일한 횟수의 읽기 연산과 4배에 해당하는 쓰기 연산이 라인메모리에서 발생하게 된다.

<표 11>은 <그림 6>과 <표 5>에 나타낸 해상도 변환 경우에 대하여, 한 프레임 계산 시 라인메모리에서 발생하는 읽기/쓰기 연산을 비교하여 나타낸 것이다. <표 11>로부터, 제안하는 구조를 이용하면 라인메모리에서 발생하는 쓰기 연산을 대폭 줄일 수 있어, 소모되는 전력 또한 크게 줄일 수 있다는 것을 알 수 있다.

Ⅳ. 결 론

본 논문에서는 기존의 3차 보간과 룩업테이블을 적용한 스케일러에 대해서 저전력 및 소형화를 달성하기

위하여 새로운 메모리 부 및 보간 부의 구조와 최적화된 룩업테이블의 구성을 제안하였다. 메모리 부에 대해서는 기존의 버퍼를 이용하는 구조 대신 두 개의 라인 메모리를 추가적으로 사용하여 쓰기 연산 횟수를 1/4회로 줄였으며, 싱글 포트(single port) 메모리의 적용 가능성을 통해 소비 전력을 줄일 수 있는 하드웨어 구조를 제안하였다. 또한, 소형화를 위해 3차 보간 수식을 재정리하여 하드웨어 구현 시 복잡도를 늘리는 곱셈 연산에 대한 항을 줄였으며, 그 결과 6개의 곱셈기와 22개의 덧셈기만으로 구성 가능한 보간 부의 하드웨어 구조를 제안하였다. 마지막으로 결과 영상의 화질을 유지하면서 룩업테이블의 크기를 줄이기 위하여 룩업테이블의 정확도 및 양자화 레벨에 대한 타협점을 이론 및 실험적으로 분석하였고, 결과적으로 32단계의 양자화 레벨과 12-비트의 정확도를 갖는 룩업테이블을 사용할 것을 제안하였다. 따라서 본 논문에서 제안하는 메모리 부와 보간 부 및 최적화된 룩업테이블을 이용하면 기존의 방법과 동일한 성능을 보이면서 하드웨어 크기와 소모 전력은 작은 스케일러를 구성할 수 있다. 또한, 앞으로의 과제로서, 다운 스케일 시에 발생하는 에일리어싱을 최소화함으로써 영상 스케일러의 성능을 개선하기 위하여, 해상도 축소 비율에 따라 변하는 가변 저역 통과 필터를 함께 적용하는 방법 등의 연구를 진행하고 있다.

참 고 문 헌

[1] D. Soo, M. Long, "System and method for image scaling interpolation," United States Patent, Patent No. US6903749B2, June 2005.

[2] C. W. Malinowski, S. R. Zepp, "Video scaling method and device," United States Patent, Patent No. US5574572A, November 1996.

[3] E. Aho, J. Vanne, T. D. Hämäläinen and K. Kuusilinna, "Block-level parallel processing for scaling evenly divisible images," IEEE Trans. on Circuits and Systems, Vol. 52, No. 12, pp. 2717 - 2725, December 2005.

[4] M. A. Nuno-Maganda, M. O. Arias-Estrada, "Real-time FPGA-based architecture for bicubic interpolation: an application for digital image scaling," in Proc. of IEEE Conf. on Reconfigurable Computing and FPGAs, pp. 1-8, Puebla, Maxico, September 2005.

[5] B. G. Lee, Y. M. Kwon, C. G. Oh, Y. M. Chung, "Design of a scan converter using cubic splines," IEEE Trans. on Consumer Electronics, Vol. 47, No. 1, pp. 6-9, February 2001.

[6] 전영현, 윤종호, 박진성, 최명렬, "저연산을 위한 수정된 3차 회선 스케일러 구현," 멀티미디어학회 논문지, 제10권, 제7호, 838-845쪽, 2007년 7월.

[7] 심우성, 최성규, "디지털영상 처리 장치 및 방법," 대한민국특허청, 등록번호 10-0423503, 2004년 3월.

[8] C. C. Lin, Z. C. Wu, W. K. Tsai, M. H. Sheu, H. K. Chiang, "The efficient VLSI design of BI-CUBIC convolution interpolation for digital image processing," in Proc. of IEEE Symp. on Circuits and Systems, pp. 480-483, Seattle, Washington, USA, May 2008.

[9] J. J. Lee, Y. M. Chung, C. G. Oh, J. G. Kim, C. W. Hong, "Design of a scan format converter using the bisigmoidal interpolation," IEEE Trans. on Consumer Electronics, Vol. 44, No. 3, pp. 115-1121, June 1998.

[10] 김재진, 박남서, 이범근, "LCOS(Liquid Crystal On Silicon)를 위한 바이시그모이드 보간을 적용한 스케일러와 컬러 콘트롤 드라이버 설계," 전자공학회 논문지, 제40권, TE편, 제4호, 1-7쪽, 2003년 12월.

[11] 성시문, 이진언, 김춘호, 김이섭, "면적 점유비를 이용한 영상 스케일러의 설계," 전자공학회 논문지, 제40권, SD편, 제3호, 43-53쪽, 2003년 3월.

[12] I. J. Hwang, B. S. Kang, J. Gerard, "High-resolution image scaler using interpolation filter for multimedia video applications," IEEE Trans. on Consumer Electronics, Vol. 43, No. 3, pp. 813-818, August 1997.

[13] E. G. Keys, "Cubic convolution interpolation for digital image processing," IEEE Trans. on Acoustics, Speech, and Signal Processing, Vol.29, No.6, pp.1153 - 1160, December 1981.

[14] T. M. Lehmann, C. Gonner, K. Spitzer, "Survey: Interpolation methods in medical image processing," IEEE Trans. on Medical Imaging, Vol. 18, No. 11, pp. 1049-1075, November 1999.

저 자 소 개

한 재 영(학생회원)
2008년 광운대학교 컴퓨터공학과 졸업.
2010년 광운대학교 컴퓨터공학과 석사졸업.
<주관심분야 : 컴퓨터구조, 영상신호처리>

이 성 원(정회원)
1988년 서울대학교 제어계측공학과 졸업(공학사).
1990년 서울대학교 제어계측공학과 석사졸업.
2003년 University of Southern California 전기공학과 박사졸업.
2010년 현재 광운대학교 전자정보공과대학 컴퓨터공학과 교수.
<주관심분야 : 미디어프로세서 및 SoC 설계, 영상신호처리, Power-Aware Computing>