http://dx.doi.org/10.3745/KTSDE.2013.2.11.809

Linear Regression-Based Precision Enhancement of Summed Area Table

Juhyeon Jeong† · Sungkil Lee††

ABSTRACT

Summed area table (SAT) is a data structure in which the sum of pixel values in an arbitrary rectangular area can be represented by the linear combination of four pixel values. Since SAT serially accumulates the pixel values from an image corner to the other corner, a high-resolution image can yield overflow in a floating-point representation. In this paper, we present a new SAT construction technique, which accumulates only the residuals from the linearly-regressed representation of an image and thereby significantly reduces the accumulation errors. Also, we propose a method to find the integral of the linear regression in constant time using double integral. We performed experiments on the image reconstruction, and the results showed that our approach more reduces the accumulation errors than the conventional fixed-offset SAT.

Keywords : Summed Area Table, Regression

선형 회귀분석 기반 합산영역테이블 정밀도 향상 기법

정 주 현† · 이 성 길††

요 약

합산영역테이블은 이미지 픽셀 주변 임의의 사각 영역 내 픽셀 값의 합을 4개 픽셀의 합차로 표현할 수 있는 자료구조이다. 그러나 합산영역테이블은 픽셀의 값을 한쪽 모서리에서 다른 쪽 모서리로 순차 누적하므로, 이미지의 크기가 큰 경우에 부동소수점 방식의 표현 범위를 초과하는 문제가 일어날 수 있다. 이를 해결하기 위해 본 논문은 선형 회귀분석을 이용하여 이미지를 근사하고, 회귀분석식과의 차이만을 누적하여 정밀도 누적 오차를 감소시킬 수 있는 제안한다. 또한, 이미지의 복원 시 회귀분석식의 합을 2중 적분을 이용하여 상수시간에 구할 수 있는 방법을 함께 제안 한다. 이미지의 복원에 대한 실험을 수행하였고, 결과는 제안하는 방식이 일반적인 고정오프셋 방식보다 누적 오차를 감소시킴을 보였다.

키워드 : 합산영역테이블, 회귀분석

1. 서 론

합산영역테이블(Summed Area Table; 이하 SAT)은 이미지 픽셀 주변 임의의 사각 영역 내 픽셀 값의 합을 영역의 넓이(즉, 샘플링할 픽셀 수)에 관계없이 동일한 수의 픽셀을 사용하여 상수 시간에 구할 수 있는 2차원 테이블형의 자료구조이다[1]. SAT는 각 성분이 현재 위치와 아래 왼쪽 모서리(혹은 위 오른쪽 모서리)로 정의된 사각 영역을 이루는 모든 성분의 합을 저장한다. 이를 통해 현재 픽셀 주변 값의 합(혹은 평균)을 영역 내 모든 픽셀 값이 없이 단 4개의 픽셀 값들의 합과 차로 표시할 수 있어 사각 영역의 크기가 증가할 경우에도 픽셀 샘플링으로 인한 성능저하가 일어나지 않는다. 이러한 SAT는 고성능 사각 필터링(box filtering)을 필요로 하는 이미지 처리나 대용량 데이터 처리에 널리 사용되고 있다[2-5].

SAT는 좌측 아래(혹은 우측 위) 모서리에서 시작하여 그 반대 방향으로 사각영역의 합을 재귀적으로 합산하여 생성된다. 따라서 색상의 강도가 높거나 고해상도 이미지에 SAT를 적용할 경우와 같이 많은 픽셀의 값이 누적되어야 하는 경우, 일반적인 IEEE-754 방식 32비트 부동소수점 표현의 표현 범위를 넘어 정밀도가 떨어질 수 있는 문제가 있어, 다소 제한적으로 사용되어 왔다.

이러한 정밀도 오차를 줄이기 위한 방법으로 일반적으로 고정 오프셋(offset)을 적용하여 그 오프셋과의 차이만을 누적하여 SAT를 생성하는 방법이 사용된다[2]. 예를 들어, 정

※ 이 논문은 한국정보처리학회 제39회 춘계학술발표대회에서 '선형 회귀분석을 이용한 합산 영역 테이블의 정밀도 향상'의 제목으로 발표된 논문을 확장한 것이며, 2012년도 정부(미래창조과학부)의 재원으로 한국연구재단의 기초연구, 중견연구자, <실감교류 인체감응솔루션> 글로벌프론티어 사업의 지원으로 수행되었음(2011-00014015, 2012R1A2A2A01045719, 2012M3A6A3055695).
† 준 회 원 : 성균관대학교 전자전기컴퓨터공학과 석사과정
†† 종신회원 : 성균관대학교 컴퓨터공학과 조교수
논문접수 : 2013년 6월 28일
수 정 일 : 1차 2013년 8월 6일
심사완료 : 2013년 8월 6일
* Corresponding Author : Sungkil Lee(sungkil@skku.edu)

규화(normalization)된 이미지에 0.5부터의 offset을 저장하는 방법이 있다. 그러나 모든 픽셀에 대해 동일한 오프셋을 사용하기 때문에 여전히 고해상도 이미지나 색상 분포에 따른 문제점을 완전히 해결하지 못한다.

본 논문은 SAT의 정밀도를 보다 효과적으로 향상시키기 위해 픽셀별 오프셋을 사용하는 방법을 제안한다. 픽셀별 오프셋은 선형 회귀분석(linear regression)을 이용하여 구하고, SAT 생성 단계에서 회귀분석식과의 차이(residual)만을 누적함으로써, 일반적인 고정 오프셋 방식보다 누적오차를 줄여, 정밀도를 현격히 향상시킬 수 있다. 이러한 회귀분석 기반의 오프셋은 고정 오프셋 방식과 달리 각 픽셀 별로 그 값이 변하므로, 이미지 복원 시 각 픽셀에서 회귀분석된 오프셋을 따로 구하여 전 영역에 걸쳐 모두 더해야 하는 부하가 추가로 발생한다. 이에, 본 논문에서는 이러한 회귀분석 오프셋의 합을 수치적으로 적분하여 이미지의 복원을 상수 시간에 가능하게 하여, 이미지 복원 성능 또한 비약적으로 향상시키는 방법도 함께 제안한다.

2. 관련 연구

SAT는 Crow[1]에 의해 처음으로 제안되었다. 텍스쳐 맵핑(texture mapping)의 앨리어싱(aliasing)을 제거하기 위해 일반적으로 사용되는 밉맵(mipmap)보다 pre-filtering의 정확도를 향상시키기 위해 제안되었으나, 테이블의 생성 시간을 고려하지 않아 그 성능이 느린 문제점이 있었다.

Hensley 등[2]은 SAT를 보다 빠르게 생성하기 위해 recursive doubling 알고리즘을 이용한 병렬 생성 방식을 제안하였다(3.2 참조). 또한 정밀도 문제에 관련하여 이미지에 고정 오프셋을 적용하여 테이블을 생성할 것을 제시하였다. 이러한 방법을 통해 픽셀 값의 범위를 [0.0 1.0]으로부터 [-0.5 0.5]로 이동시켜 단조 증가로 인해 생길 효과를 상쇄시킬 수 있었다. 그러나 고정 오프셋 방식은 여전히 이미지의 해상도와 색상 분포에 따라 음수나 양수 방향으로 정밀도가 떨어지게 될 잠재적인 문제점을 안고 있다. 따라서 본 논문에서 제안하는 방법은 픽셀 그리고 채널 별로 다른 오프셋을 적용하여 정밀도를 더욱 향상한다. 이와 함께 SAT를 이용한 translucency 렌더링, glossy environmental reflection, depth of field 렌더링을 소개하였으며[2-3] 추가적인 연구를 통해 high-dynamic range environment map을 이용한 image-based lighting 기법을 소개하였다[4].

Sengupta 등[6]은 recursive doubling의 불필요한 계산 요소를 줄이기 위한 balanced trees 기반 work-efficient parallel scan 알고리즘을 소개하였으며, 최근에는 테이블 생성 시 memory bandwidth를 줄이는 방법으로 shared memory를 이용하는 기법이 연구되고 있다. Harris 등[7]은 CUDA를 이용한 work-efficient parallel scan 알고리즘을 소개하였으며, Nehab 등[8]은 overlapping 알고리즘을 소개하였다. 수직 방향 계산을 위해 transpose를 이용, 수평 방향으로 계산을 한다. shared memory 사용으로 단순 병렬 알고리즘에 비해 그 성능을 향상시켰다. 이러한 연구들은 테이블 생성의 성능을 향상시키지만 정밀도에 대해서는 동일한 문제를 내포하고 있다.

Shen 등[5]은 soft shadow mapping에 사용할 pre-filtering을 위해 SAT를 사용하였다. 정밀도 문제를 해결하기 위해 SAT tile grid를 이용하였다. 이미지를 grid로 나누어 각 grid별로 SAT를 생성하였으며, 필요한 grid 수를 계산하는 방법을 소개하였다.

3. 알고리즘

3.1 선형 회귀분석을 통한 오프셋 정의

SAT의 정밀도를 높이기 위해 본 논문에서는 선형 회귀분석을 이용하여 각 픽셀의 오프셋을 계산한다. Fig. 1은 1차원 배열의 경우에 대해 같은 고정 오프셋을 적용하는 경우와 선형 회귀분석을 통해 오프셋을 적용하는 경우를 비교하여 예시한다.

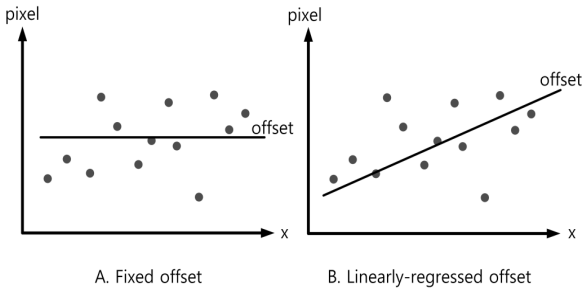

Fig. 1. Comparison of residuals for 1-D array

Fig. 1A는 고정 오프셋을 사용한 경우를 보여주며 Fig. 1B는 선형 회귀분석을 이용한 오프셋을 사용한 경우를 보여준다. Fig. 1B를 보게 되면 각 픽셀별로 회귀분석된 선으로부터 수직 방향으로의 거리를 오프셋으로 적용하였다. 선형 회귀분석 오프셋을 적용할 경우 모든 픽셀에 대해 동일한 값을 적용하는 방식보다 더 많은 픽셀이 0에 가깝게 수렴하게 되는 것을 알 수 있다. 이로 인해, SAT를 생성할 때 총합이 부동소수점 표현 방식의 한계 범위를 초과하지 않는 효과를 기대할 수 있다. 또한, 본 방식을 이용하여 픽셀마다 각각 다른 오프셋을 적용하면, 기저값이 되는 오프셋을 선형 회귀식으로부터 쉽게 구할 수 있어, 별도의 이미지(혹은 텍스쳐)를 생성할 필요가 없이 원본 픽셀을 빠르게 복원할 수 있다.

각 픽셀에 대해 적용되는 오프셋은 식 (1)과 같다. x와 y는 각각 이미지의 x축, y축 좌표이다. 회귀분석 결과는 RGB 채널별로 다를 수 있으므로, RGB 채널별로 별도의 회귀분석을 수행하여, 각 채널별로 별도의 회귀분석 오프셋을 이용한다.

$$\begin{aligned} r(x,y) &= a_r x + b_r y + c_r \\ g(x,y) &= a_g x + b_g y + c_g \\ b(x,y) &= a_b x + b_b y + c_b \end{aligned} \tag{1}$$

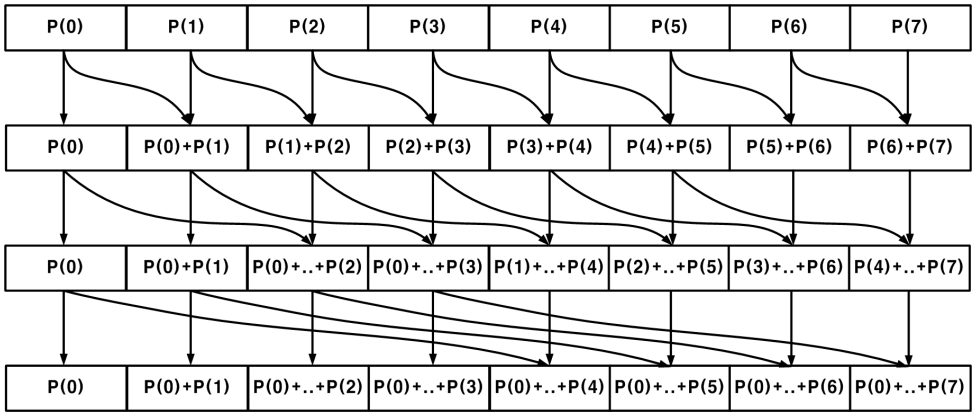

Fig. 2. Recursive doubling algorithm [2]

3.2 Recursive doubling을 이용한 SAT 생성

본 논문에서는 SAT를 생성하기 위해 보편적인 recursive doubling 알고리즘을 사용한다[2, 9]. Fig. 2는 1차원 배열에 대한 recursive doubling 알고리즘의 예를 보여준다. recursive doubling은 병렬화가 가능한 방법으로 일반적으로 GPU에서 구현된다. GPU 프로그램의 성능은 텍스쳐 샘플링의 수에 선형적으로 비례하므로, recursive doubling 방법은 텍스쳐 샘플링의 효율성을 위해 수평 단계와 수직 단계를 연달아 수행하는 2-pass를 연달아 수행하는 방식이다. 수평 단계에선 각 행마다 수평 방향으로 프리픽스 섬(prefix sum)을 하여 저장한 후, 수직 단계에서 각 열마다 수직 방향으로 프리픽스 섬을 실행한다. 현재 수행하는 이미지 혹은 2D 배열의 너비와 높이를 w, h라고 할 때, 각 단계의 패스 수는 수평 단계에서 $\log_2(w)$, 수직 단계에서는 $\log_2(h)$이다. 이에 프리픽스 섬을 수평, 수직 방향에 대해 반복적으로 수행하여 2n 크기를 포함하는 테이블을 생성할 수 있다.

Recursive doubling 알고리즘은 보편적인 shader 기반 프로그래밍과 GPGPU 방식의 프로그래밍에 공통적으로 적용 가능하며, 본 연구에서는 pixel shader 기반의 프로그래밍 방법을 사용한다. 본 논문에서 제안하는 회귀분석 기반 방법은 SAT의 생성 방법과는 의존성이 없으므로, 최근 방식인 GPU의 shared memory 기반의 보다 효율적인 방법들[7-8]을 이용하여 성능을 향상시킬 수 있다.

3.3 SAT 필터링을 이용한 이미지 복원

앞서 말한 바와 같이, SAT는 임의의 사각 영역 내 모든 픽셀의 합을 구하는데 사용되며, 이는 다음 식과 같이 4개 픽셀의 선형 합으로 표현될 수 있다.

$$average = \frac{UR - UL - LR + LL}{w * h} \tag{2}$$

GPU 구현에서는 4번의 텍스쳐 룩업이 필요하며, 텍스쳐로 생각하면 Fig. 3과 같이 보일 수 있다.

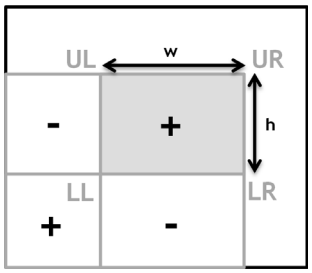

Fig. 3. Reconstruction of a pixel sum for a rectangular area in the summed area table

SAT에 식 (2)를 적용하여 복원한 이미지는 오프셋이 적용된 이미지이다. 따라서 선형 회귀 오프셋을 적용하기 전의 선형회귀 오프셋 값을 더하는 보정을 요구한다. 하지만 고정 오프셋과는 다른, 선형 회귀분석을 통한 오프셋 정의로 인해 각 픽셀 마다 다른 오프셋을 가진다. 그러므로 모든 픽셀의 오프셋 값을 선형 회귀식을 통하여 구한 값과 더하면 원본을 복원할 수 있다.

그러나, 이러한 방법은 큰 영역에 대하여 무시하지 못할 계산 부하를 동반하므로 본 논문에서는 아래와 같이 선형 회귀식을 x와 y에 대해 2중 적분한 식을 이용하여 간단히 상수 시간에 구할 수 있는 방법을 제안한다.

식 (1)의 적분은 아래 식 (3)과 같이 표현할 수 있다.

$$\begin{aligned}
\int_{y_0}^{y_1}\int_{x_0}^{x_1} r(x,y)dxdy &= \frac{1}{2}wh(a_r(x_1+x_0)+b_r(y_1+y_0)+2c_r)\\
\int_{y_0}^{y_1}\int_{x_0}^{x_1} g(x,y)dxdy &= \frac{1}{2}wh(a_g(x_1+x_0)+b_g(y_1+y_0)+2c_g)\\
\int_{y_0}^{y_1}\int_{x_0}^{x_1} b(x,y)dxdy &= \frac{1}{2}wh(a_b(x_1+x_0)+b_b(y_1+y_0)+2c_b)
\end{aligned} \tag{3}$$

x_1, x_0, y_1, y_0는 사각 영역을 이루는 x, y 좌표이며, LL, LR, UL를 이용해 얻을 수 있다. w, h는 사각 영역의 너비(x_1-x_0)와 높이(y_1-y_0)이다. 본 적분을 이용하여 사각영역 내의 회귀값의 합을 마찬가지로 상수시간에 구할 수 있어, 복원의 부하도 무시할 정도로 작아지게 되어, 품질의 향상에 따르는 부하를 최소화할 수 있다.

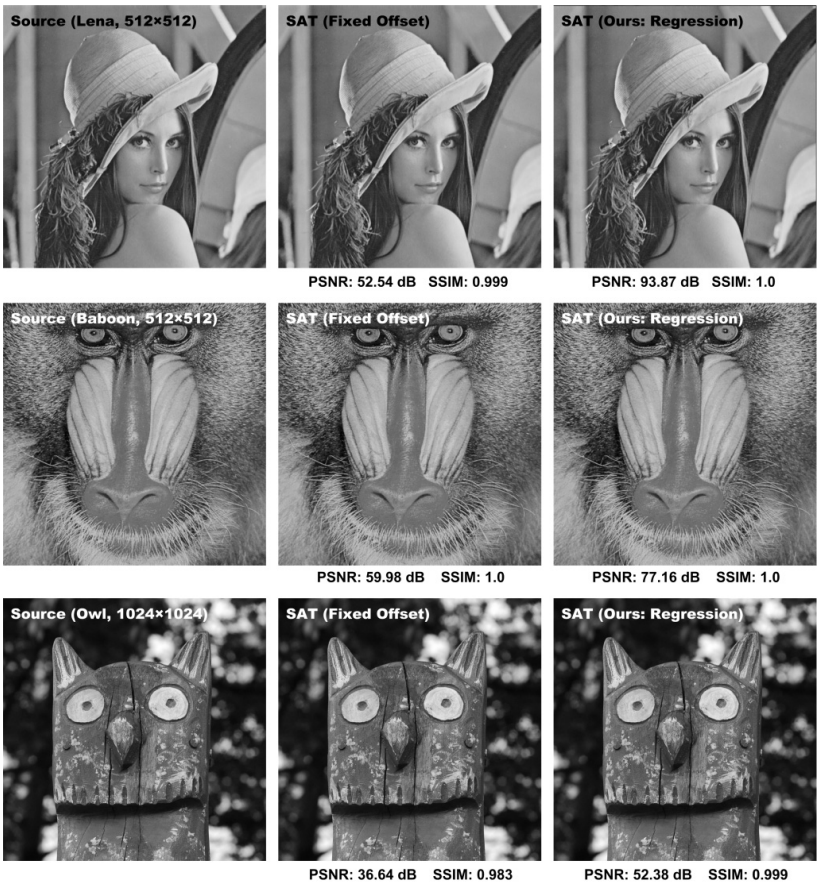

Fig. 4. Comparison of results reconstructed from the summed area table using a fixed offset and our method (linear regression) against three source images

4. 결과 및 토론

4.1 이미지 복원 품질

정밀도 오차를 측정하기 위해 원본 이미지와 SAT를 이용하여 복원한 이미지를 비교하였다(Fig. 4). 이는 사각필터링을 적용하지 않고, 원본 그대로의 이미지와 비교를 함으로써, 누적 자체의 오차를 보기 위함이다. 보편적인 고정오프셋 방식과 본 논문에서 제안하는 방법이 사용되었다.

Lena, Baboon, Owl의 3가지 이미지(각 해상도 512×512, 512×512, 1024×1024)가 사용되었으며, 원본 이미지와의 차이는 PSNR, SSIM[10]의 두 대표적 방법을 이용하여 측정되었다. PSNR은 세 이미지에 대해 본 논문의 회귀분석 기반 방법이 고정 오프셋 방식에 비해 오차가 현격히 줄였음을 알 수 있다. 해상도가 높은 이미지인 경우 오차가 더 많이 누적되어 상대적으로 낮은 복원품질을 보이나, 그 역시 50 dB 이상이 되어, 만족스런 복원이 가능함을 알 수 있다. 이러한 복원 오차는 이미지의 해상도가 커질수록 그 차이가 커질 것을 예상할 수 있다. 한편, SSIM은 세 이미지 모두 1에 매우 가까워 복원된 이미지의 구조적 유사도가 매우 높은 것으로 나타나고 있고, 여전히 본 논문의 방법이 고정 오프셋 방식보다 나은 품질을 보임을 알 수 있다.

4.2 수행성능

본 논문에서 제안하는 SAT는 OpenGL을 이용하여 구현되었으며, 성능 측정을 위해 Intel Core i7 3770, 16GB RAM, NVIDIA Geforce GTX680의 사용 환경에서 테스트되었다. 고정 오프셋을 사용할 경우 모든 픽셀에 같은 오프셋이 적용되어 복원 시, SAT로부터 복원된 결과에 단 하나의 상수 오프셋을 합하여 빠르게 복원 할 수 있다. 그러나 회귀분석식을 이용한 오프셋을 사용할 경우 각 픽셀에서 다른 오프셋 값을 이용하므로, 임의의 사각영역 내 모든 픽셀에 대하여 오프셋을 계산한 후 그 합을 SAT의 복원결과에 더해주어야 한다. 예를 들어 3×3 크기의 사각 영역의 경우 모두 9개의 오프셋 계산이 필요하며, 마찬가지로 5×5 크기의 사각 영역은 25개, 7×7 크기의 사각 영역은 49개의 오프셋을 계산해야한다. 사각 영역의 크기가 커질수록 계산해야 할 오프셋의 수가 크게 증가하며, 이는 곧 성능저하의 원인이 된다.

본 논문에서는 이러한 성능저하를 매우 효과적으로 막기 위한 오프셋 적분 방식을 제안하였다. 이를 통해 문제점이었던 회귀분석 오프셋의 합을 적분을 이용하여 상수시간에 구할 수 있으며, 또한 이러한 적분은 추가적인 성능 부하가 매우 적어 일반적인 SAT와 거의 동일한 성능을 낼 수 있다.

오프셋 보정 방법에 따른 성능 차이를 관찰하기 위해 9×9 사각 필터링에 대한 실험을 실시하였다. 비교를 위해 총 3가지 방법, 고정 오프셋, 선형적인 오프셋(선형) 합산, 오프셋 적분을 이용하였으며, 성능은 초당 렌더링 프레임수를 이용하여 비교되었다(Table 1). 고정 오프셋을 사용할 경우 오프셋 복원을 위해 1개의 고정 값을 합하여 평균을 계산할 수 있으며, 선형적인 오프셋 합산의 경우 회귀분석식을 통해 81번의 오프셋 계산을 통해 합을 구한다. 오프셋 적분을 사용할 경우 한 번의 2중 적분을 통해 오프셋 합을 구한다.

Table 1. Comparison of SAT reconstruction performance

Resolution	Frames per second (fps)		
	Fixed offset	Ours (Linear sum)	Ours (Integral)
512×512	11184	3565	11200
1024×1024	3995	1044	3781

Table 1에서 볼 수 있듯이, 선형적인 오프셋 합산을 이용할 경우, 고정 오프셋 방식에 비해 모든 픽셀(81개)에 대해 회귀분석식을 계산하고 더함으로써 성능이 1/3 정도까지 감소하는 것을 볼 수 있지만, 오프셋 적분을 적용한 방식은 사각 영역의 크기에 관계없이 한 번의 계산을 통해 사각 영역의 오프셋 합을 계산하기에 그 성능은 고정오프셋 방식과 거의 차이가 없음을 알 수 있다. 이러한 실험결과를 통해 본 논문에서 제안하는 방법은 고정 오프셋 방식에서 성능 저하 없이 품질을 향상시킬 수 있음을 알 수 있다.

4.3 토론 및 향후 연구

본 연구에선 1차 다항식으로 회귀분석을 통해 오프셋을 적용하여 이미지 색상분포에 따라 R^2가 다소 낮게 나오는 경우가 생긴다. 이는 색상분포가 선형으로 균일한 이미지에서는 큰 성능향상을 보이지만, 매우 불규칙적인 이미지에서는 성능향상이 다소 감소할 수 있음을 의미한다. 이러한 제약은 본 논문에서 전역 회귀분석을 이용하므로, 국소 영역별 상세도를 적절히 반영할 수 없기 때문이다.

이에 본 저자들은 전역 회귀분석의 단점을 극복하기 위해 국소적 회귀분석을 적용하여 정밀도 오차를 보다 감소시킬 수 있는 방법을 향후 연구에 계획하고 있다. 본 논문에서 제안하는 선형 회귀분석을 넘어, 다중 차수의 회귀 분석이나 이산 자료 보간(Scattered data interpolation)과 같은 방법이 사용될 수 있으리라 예상하며, 이에 대한 연구를 진행할 예정이다.

5. 결 론

본 논문은 선형 회귀분석을 통해 이미지를 채널별로 근사하고, 회귀분석식과의 오차만을 누적하여 SAT의 정밀도를 향상시키기 위한 방법을 제안하였다. 또한 이미지 복원 시 오프셋을 복원함에 있어 상수시간에 계산 가능한 적분을 사용하여 그 성능을 비약적으로 향상시켜, 성능저하 없이 고정 오프셋 방식을 대체하여 품질을 향상시킬 수 있음을 보였다.

참 고 문 헌

[1] Crow, Franklin C. “Summed-area tables for texture mapping.” ACM SIGGRAPH Computer Graphics. Vol.18, No.3, ACM, 1984.

[2] Hensley, Justin, et al. “Fast Summed Area Table Generation and its Applications.” Computer Graphics Forum. Vol.24, No.3, Blackwell Publishing, Inc., 2005.

[3] Hensley, Justin, et al. “Interactive summed-area table generation for glossy environmental reflections.” ACM SIGGRAPH 2005 Sketches. ACM, 2005.

[4] Hensley, Justin, et al. “Fast hdr image-based lighting using summed-area tables.” Symposium on Interactive 3D Graphics and Games (Poster). 2007.

[5] Shen, Li, Jieqing Feng, and Baoguang Yang. “Exponential Soft Shadow Mapping.” Computer Graphics Forum. Vol.32, No.4, Blackwell Publishing Ltd, 2013.

[6] Sengupta, Shubhabrata, Aaron E. Lefohn, and John D. Owens. “A work-efficient step-efficient prefix sum algorithm.” Proceedings of the Workshop on Edge Computing Using New Commodity Architectures. 2006.

[7] Harris, Mark, Shubhabrata Sengupta, and John D. Owens. “Parallel prefix sum (scan) with CUDA.” GPU gems 3.39 (2007): 851-876.

[8] Nehab, Diego, et al. “GPU-efficient recursive filtering and summed-area tables.” ACM Transactions on Graphics (TOG) 30.6 (2011): 176.

[9] Horn, Daniel. “Stream reduction operations for GPGPU applications.” Gpu gems 2 (2005): 573-589.

[10] Wang, Zhou, et al. “Image quality assessment: From error visibility to structural similarity.” Image Processing, IEEE Transactions on 13.4 (2004): 600-612.

정 주 현

e-mail : jhjeong10@skku.edu

2013년 성균관대학교 컴퓨터공학과(학사)

2013년～현 재 성균관대학교 전자전기컴퓨터공학과 석사과정

관심분야 : GPGPU, Real-time rendering

이 성 길

e-mail : sungkil@skku.edu

2002년 포항공대 재료공학과(학사)

2009년 포항공대 컴퓨터공학과 (Ph.D., 석박사통합)

2011년~현 재 성균관대학교 컴퓨터공학과 조교수

관심분야 : Real-time rendering, Perception-based rendering and visualization, Image-based ray tracing, High-Performance GPU Computing, HCI