앤티앤티앨리어싱

Anti-Antialiasing

정주현, Juhyeon Jeong*, 문성호, Sungho Moon**, 이성길, Sungkil Lee***

요약 rasterization 기반의 컴퓨터 그래픽스 이미지들은 이미지 해상도보다 높은 frequency를 가지는 sub-pixel detail은 undersampling으로 인해 표현되지 못한다. 이러한 문제는 비트맵(bitmap) 방식의 이미지에서 색상이 불연속 되는 지점 즉, 가장자리(Edge)가 톱니모양으로 표현되는 계단현상 (앨리어싱; Aliasing)이 생긴다. 이런 앨리어싱을 가지지 않고 부드러운 이미지의 계를 얻기 위한 방법론을 앤티앨리어싱(Antialiasing)이라 하며, 렌더링에서는 필수적인 기법이다. 그러나, 앤티앨리어싱은 그 기본 개념으로 인해 픽셀 내에 여러개의 샘플링된 색을 포함하여 혼합된 픽셀을 생성하고, 이는 객체들 사이의 경계선을 모호하게 만든다. 본 논문은 앤티앨리어싱이 적용되었거나 광학 시스템으로부터 생성된 이미지들에서 앤티앨리어싱을 제거하는 방법을 제안한다. 이 방법은 여러 개의 샘플을 통한 필터링(spatial filtering)과정을 모델링한 후, 그 역의 과정을 통해 필터링되기 전의 픽셀 색을 찾는 것이다. 하지만 필터링되기 전의 원본 색을 알 기 어렵기 때문에 정확히 필터링을 모델링할 수 없다. 이에, 본 논문은 필터링된 주변 색상을 근사적으로 필터링 모델에 대입하여 앤티앨리어싱을 제거한다. 이로 인해 향상된 세그멘테이션(segmentation)은 이미지 편집 프로그램이나 레이어 기반의 렌더링 알고리즘에 유용히 사용될 수 있다.

Abstract This is an example of ABSTRACT format. The heading, 'Abstract', should be made 'bold' after the style sheet [abstract] has been applied. Abstract should be typed in Yoon Myungjo 120, 9 points in size, horizontally scaled to 95%, with tracking of -5, line spaced in 120%, justified, 30 points on left & right margin. A blank line should be inserted after 요약문, where the abstract starts.

색심어: Antialiasing

본 논문은 2012년도 정부(교육과학기술부)의 재원으로 한국연구재단의 기초연구, 중견연구자, 〈실감교류 인체감응솔루션〉 글로벌프론티어 사업의 지원으로 수행되었음(2011-00014015, 2012R1A2A2A01045719, 2012M3A6A3055695).

*주저자 : 성균관대학교 컴퓨터공학과 학사과정 e-mail: jhjeong10@skku.edu

**공동저자 : 성균관대학교 컴퓨터공학과 학사과정 e-mail: tellel@skku.edu

***교신저자 : 성균관대학교 컴퓨터공학과 교수; e-mail: sungkil@skku.edu

1. 서론

실세계의 광학 시스템은 센서에 빛이 기록될 때 자연스럽게 주변의 빛이 누적되어 낮은 해상도에도 자연스러운 이미지가 생성이 되지만, rasterization 기반의 컴퓨터 그래픽스 이미지들은 이미지 해상도보다 높은 frequency를 가지는 sub-pixel detail은 undersampling으로 인해 표현되지 못한다. 이러한 문제는 비트맵(bitmap) 방식의 이미지에서 색상이 불연속 되는 지점 즉, 가장자리(Edge)가 톱니모양으로 표현되는 계단현상 (앨리어싱; Aliasing)이 생긴다. 이런 앨리어싱을 없이 부드러운 이미지를 얻기 위한 방법론을 앤티앨리어싱(Antialiasing)이라 하며, 렌더링에서는 필수적인 기법이다 (관련 연구 참조).

최근 컴퓨터 그래픽스에서는 rasterization으로 생성된 이미지나 실사 이미지를 분할(segmentation)하거나, 여러 개의 layer를 추출하여 특수 효과를 위해 후처리 (postprocessing) 기법을 많이 이용한다.

그러나, 앤티앨리어싱은 그 기본 원리로 인해 픽셀 내에 여러 개의 샘플링된 색을 포함하여 혼합된 픽셀을 생성하고, 이는 객체들 사이의 경계선을 모호하게 만든다. 이에 분할(segmentation)이나 클러스터링(clustering)을 하여 이미지를 자르거나 나눌 때 가장자리 주변의 두 색상의 강도가 크게 차이가 나지 않는다면 이미지를 깨끗하게 잘라내기 어렵다. 또한, 실제 광학 시스템을 이용하여 생성한 이미지들도, 자연스럽게 많은 detail이 포함되어 있으므로, 유사한 문제가 발생한다.

이런 문제를 극복하기 위해 정확한 가장자리를 찾기 위한 효과적인 가장자리 검출(edge detection) 방법들이 제안 되었다. 그러나 이런 방법들은 노이즈에 민감하며 이미지에 따라 경계선 인식 성능이 달라질 뿐 아니라, 경계 역시 하나의 영역이나 layer에 포함되어야 하므로 여전히 모호함을 내포한다.

본 논문은 앤티앨리어싱이 적용되었거나 광학시스템으로부터 생성된 이미지들에서 앤티앨리어싱을 제거하는 방법을 제안한다. 앤티-앤티앨리어싱의 기본 아이디어는 여러 개의 샘플을 통한 필터링(spatial filtering)과정을 모델링한 후, 그 역의 과정을 통해 필터링되기 전의 픽셀 색을 찾는 것이다. 본 접근 방식의 어려움은 필터링되기 전의 원본 색을 알기 어렵기 때문에 정확히 필터링을 모델링할 수 없음에서 나온다. 이에, 본 논문은 필터링된 주변 색상을 근사적으로 필터링 모델에 대입하여 앤티앨리어싱을 제거한다.

이로 인해 향상된 세그멘테이션은 경계선 그리기, 이미지 합성, 특정 이미지나 배경 색깔 변경을 깨끗이 하여 이미지 편집 프로그램이나 레이어 기반의 렌더링 알고리즘에 유용히 사용될 수 있다.

2. 관련연구

앨리어싱은 주로 벡터(vector) 표현을 rasterization할 때 생겨난다. 비트맵 이미지는 픽셀(pixel) 이라는 매우 작은 사각형을 기본 단위로 구성하고 있다. 픽셀의 색을 정할 때 하나의 샘플만을 사용 할 경우, 픽셀의 크기보다 작은 상세함(sub-pixel detail)이 있을 경우 나타나고, 주로 계단현상이 생긴다. 또한, 원본 이미지를 보다 저해상도로 resampling 할 때, nearest-neighbor sampling을 사용할 경우에도 생겨난다. 이런 계단현상은 가장자리를 거칠게 보이게 하며 원래 이미지의 품질을 많이 저하한다. 앨리어싱을 제거하는 기법인 앤티앨리어싱을 적용하면 부드러운 가장자리를 얻을 수 있다.

직관적인 앤티앨리어싱 방법은 이미지의 해상도보다 높은 샘플링 레이트를 사용하는 Super-sampling 기반 앤티앨리어싱이다. 실시간 렌더링에서는 색의 샘플링은 유지한 채, depth buffer만 여러번 sampling 하는 MSAA (Multi-sampling Anti-Aliasing)가 보편적이다. 최근에는 이러한 렌더링 부하를 대폭 줄이기 위한 후처리 기반의 MLAA (Morphological Anti-Aliasing), FXAA(Fast approximate Anti-Aliasing)과 같은 방법 등이 연구되고 있다.

많은 앤티앨리어싱 방법들이 제안되었음에도, 역의 과정인 anti-antialiasing의 과정은 연구된 바가 거의 없기에, 본 논문에서는 이러한 개념과 알고리즘을 제안하며, 레이어나 segmentation 기반의 후처리 알고리즘에 쉽게 쓰일 수 있도록 함을 지향한다.

3. 앤티앤티앨리어싱 알고리즘

앤티앨리어싱을 제거하기 위한 알고리즘은 다음과 같은 식으로 표현된다.

$$I_0 = \frac{I'_0 - \sum \omega_k I'_k (1 - \rho_k)}{\omega_0 + \sum \omega_k \rho_k} \quad (1)$$

I_0는 앤티앨리어싱이 적용 전의 픽셀 색이며 식 (1)을 통하여 찾는 본 알고리즘의 결과이다. I'_0는 앤티앨리어싱이 적용된 픽셀이다. I'_k은 I'_0을 둘러싸는 주변 여덟 픽셀이다. ω_0와 ω_k는 각 픽셀의 Gaussian weight이다 (그림 1).

1	2	1
2	4	2
1	2	1

그림 1. 3×3 가우시안 weight.

식 (1)은 다음과 같이 유도된다. 앤티앨리어싱 적용시 I'_0를 생성하는 필터링(예, 3×3)은 식 (2)와 같이 쓸 수 있다.

$$I'_0 = \sum_{k\in\Omega} \omega_k I_k + \omega_0 I_0 \quad (2)$$

I_k는 I_0를 둘러싸고 있는 주변 픽셀이다. I_0를 중심으로 식 (2)를 다시쓰면 다음과 같다.

$$I_0 = \frac{1}{\omega_0}(I'_0 - \sum_{k\in\Omega} \omega_k I_k) \quad (3)$$

I'_0, ω_0, ω_k는 알고 있는 값이지만, I_k는 미지수이므로 I'_k를 이용하여 근사한다. 그러나 I'_k의 값이 I'_0 와 비슷한 경우에는 필터링을 거쳐 값이 많이 변형되었을 가능성이 높으므로, 이러한 값은 근사에서 제외시켜야 한다. 이러한 변형 가능성은 I'_k와의 유사함으로 생각하여 아래의 과정을 통해 선형보간으로 모델링한다. I_k를 근사한 값을 $\hat{I}_k$ 라 하면, 다음과 같이 쓸 수 있다.

$$\hat{I}_k \simeq I_0 \cdot \rho_k + I'_k \cdot (1 - \rho_k) \quad (4)$$

식 (4)에서 ρ_k는 I'_0, I'_k가 얼마나 다른지 측정하는 weight이고, 다음과 같이 Gaussian weight로 모델링한다.

$$\rho_k = e^{-(\frac{\delta(I'_0, I'_k)}{\sigma})^2} \quad (5)$$

$\delta(I'_0, I'_k)$ 는 Euclidean color difference이다.

식 (4)를 식 (3)에 대입하여 I_0를 중심으로 정리하면 앞서 보인 식 (1)과 같다. 식 (1)은 I'_0와 다른 픽셀 값들의 합 $\Sigma\omega_k I'_k(1-\rho_k)$을 I'_0 로부터 제거하여 I'_0를 이루고 다른 색상들을 분리하는 과정을 의미한다.

유의할 점은 I'_0가 그림 2와 같이 I'_k을 이루는 두 색상의 비율이 1:1이 되어 $\Sigma\omega_k I'_k$와 거의 비슷한 경우이다.

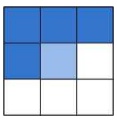

그림 2. I'0의 색이 나머지 주변 픽셀의 평균값을 가질 경우

즉, I'_0 가 나머지 주변 픽셀 I'_k 의 평균값일 경우 $\delta(I'_0, I'_k)$ 값이 모두 같아 I'_0로부터 색상이 분리되지 않는 문제가 생긴다. 이를 해결하기 위해 평균값에 가까운 I'_0에 오프셋을 더하여 어느 한쪽으로 색이 이동할 수 있도록 한다. 오프셋은 다음과 같이 정의된다.

$$O = sign(I'_0 - I'_c) \cdot O_{max} \cdot e^{-(\frac{\delta(I'_0, I'_c)}{\sigma_o})^2} \quad (6)$$

I'_c는 I'_k의 평균값이고, I'_0–I'_c의 부호는 벡터 내적으로 구하며, 오프셋 최대값 O_{max}는 $\delta(I'_0, I'_c)$를 이용하여 정한다.

4. 결과

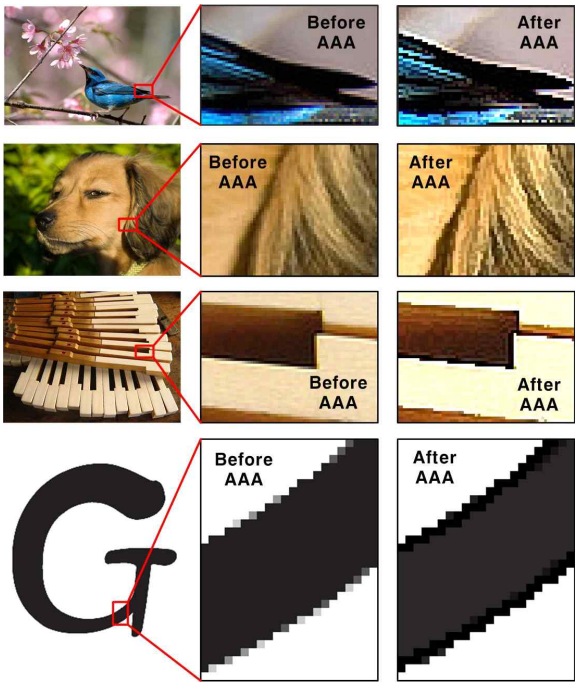

그림 3. I'0의 색이 나머지 주변 픽셀의 평균값을 가질 경우

본 알고리즘은 Intel i7-3770, NVIDIA Geforce GTX 680, DX10 API를 사용하여 구현되었다. 그림 3은 품질 측정을 위해 사용된 3가지 사진과 텍스트를 보인다. 앤티앨리어싱된 이미지에 앤티앨리어싱 제거를 적용한 후 비교한 모습을 보여준다. 일반적으로 앤티앨리어싱 제거를 적용하였을 때 가장자리에 존재하는 대부분의 픽셀이 두 가지 색 중 한 가지 색에 가까워져 앤티앨리어싱이 제거된다. 앤티앨리어싱 제거로 인해 부드럽게 보이던 가장자리에 계단 현상이 다시 발생하였다. 이러한 효과는 뚜렷한 경계선을 만들어 세그멘테이션을 적용할 시 정확성을 높인다.

본 연구의 결과는 antialiasing이 제거된 후 색이 다소 과장되어 밝아지거나 어두워지는 문제가 있다. 추후 연구는 이러한 문제에 대한 해결을 포함한다.

참고문헌

[1] AKELEY, K. 1993. Reality engine graphics. In Proc. SIGGRAPH, 109-116.

[2] RESHETOV, A. 2009. Morphological antialiasing. In Proc. HPC 2009, ACM, 109-116.

[3] NVIDIA. Fast Approximate Antialiasing, http://developer.download.nvidia.com/assets/gamedev/files/sdk/11/FXAA_WhitePaper.pdf