Journal of Digital Contents Society Vol. 8 No. 1 Mar. 2007(pp. 85-91)

# 포인트 샘플링으로 표현된 3차원 객체를 위한 하이브리드 앤티앨리어싱 방법

김학란*, 박화진**


## 요 약

본 논문에서는 포인트 샘플링을 이용한 음함수 곡면에서 나타나는 앨리어싱을 줄이기 위한 새로운 하이브리드 앤티앨리어싱 방법을 제안한다. 하이브리드 앤티앨리어싱 방법은 하나의 음함수 곡면에 대해 Z-버퍼에서 나타나는 픽셀 값들의 차이를 구하여 세가지 형태의 Z-버퍼를 각각 사용하는 방법이다. 차이의 수준을 정하여 차이가 심할 때는 멀티 Z-버퍼를 사용하고 차이가 중간 정도일 때는 더블 Z-버퍼를 사용하며 차이가 경미할 때는 원래의 Z-버퍼만을 사용하여 3차원 객체를 표현하는 방법이다. 기존에 앤티앨리어싱 효과를 높이기 위하여 전체적으로 멀티 Z-버퍼를 사용하였는데 사용되는 Z-버퍼의 개수를 줄이면서 멀티 Z-버퍼를 사용한 것과 비슷한 효과를 보이는 효율적인 방법이다


# Hybrid Anti-aliasing Method for 3D Object represented by Point Sampling

Hakran Kim*, Hwajin Park**


## Abstract

This paper proposes a new hybrid anti-aliasing method for reducing aliasing appearing on an implicit surface using the point sampling. The hybrid anti-aliasing method is a method that finds differences in the values of pixel shown in Z-buffers for an implicit surface and thereby uses each of three types of Z-buffer. After determining the level of differences, it expresses a 3 dimensional object by using a multi Z-buffer if the level is severe, a double Z-buffer for a middle level, and only the original Z-buffer for a negligible difference. In comparison with the existing method in which multi Z-buffers have been entirely used for enhancing the anti-aliasing effect, the hybrid anti-aliasing method is an efficient method demonstrating an effect similar to the one using a multi Z-buffer while reducing the number of Z-buffers to be used.


Key words : point sampling, anti-aliasing, implicit surface

## 1. 서론

실시간의 빠른 렌더링을 위한 하나의 방법인 포인트 샘플링을 이용한 3차원 그래픽 객체는 속도 면에서 빠른 장점을 지니고 있지만 앨리어싱이라는 피할 수 없는 단점을 나타낸다. 포인트 샘플링에 의한 앤티앨리어싱 방법은 기존에 유니폼 샘플링 방식 중에서 슈퍼 샘플링이나[8] non 유니폼방식인 Stochastic 샘플링을 사용하였다[6]. 하지만 고성능 그래픽 시스템에서 앨리어싱 문제를 해결하기 위해 수행되는 슈퍼샘플링은 보통 최종 생성되는 이미지보다 높은 해상도에서 이미지를 얻어낸 다음 평균다운이라는 필터링 과정을 통해 저해상도의 이미지를 생성하는 방식으로 이 방법의 문제점은 필요한 메모리의 요구량이 많다는 것이다.

이러한 문제점을 해결하기 위하여 기존에 포인트 샘플링을 사용하여 렌더링한 음함수 곡면을 예제로 사용하여 슈퍼 샘플링이나 Stochastic 샘플링을 사용한 앤티 앨리어싱이 아닌 시프트 Z-버퍼를 사용한 효율적인 앤티 앨리어싱 방법

※ 제일저자(First Author) : 김학란
접수일자:2007년01월25일, 심사완료:2007년02월24일
* 숙명여자대학교 컴퓨터과학전공
imhera@sookmyung.ac.kr
** 숙명여자대학교 멀티미디어과학전공

을 제안하였다. 이 방법은 사용하는 시프트된 버퍼의 개수를 늘리게 되면 앤티 앨리어싱 효과가 매우 크게 나타나지만 그만큼 계산시간이나 메모리 효율성이 떨어지게 되는 단점이 있다.

따라서 본 논문에서는 하나의 음함수 곡면에 대해 사용하는 Z-버퍼의 개수를 정하여 앤티앨리어싱을 구현하는 기존의 방법을 사용하지 않고 하나의 음함수 곡면에 대해 Z-버퍼에서 나타나는 주변 픽셀 값들의 차이를 구하여 세가지 형태의 Z-버퍼를 각각 사용하는 하이브리드 앤티앨리어싱 방법을 제안한다. 차이의 수준을 정하여 차이가 심한 픽셀의 경우 멀티 Z-버퍼를 사용하고 차이가 중간 정도의 픽셀에 대해서는 더블 Z-버퍼를 사용하며, 차이가 경미할 때는 시프트된 새로운 Z-버퍼를 사용하지 않고 원래의 Z-버퍼만을 사용하여 3차원 객체를 표현하는 방법이다. 기존에 앤티앨리어싱 효과를 높이기 위하여 전체적으로 멀티 Z-버퍼를 사용하였는데 사용되는 Z-버퍼의 개수를 줄이면서 멀티 Z-버퍼를 사용한 것과 비슷한 효과를 보이는 효율적인 방법이다.

## 2. 관련연구

포인트 샘플링을 이용한 저 해상도의 음함수 곡면에서 나타나는 앨리어싱을 줄이기 위해 기존에 제안한 앤티앨리어싱 방법들은 다음과 같다.

### 2.1. 더블 Z-버퍼 앤티앨리어싱

시프트 더블 Z-버퍼 앤티 앨리어싱 방법은 특정한 음함수를 통하여 얻어진 3차원 객체를 표현하기 위해서 프레임 버퍼를 사용하는 대신에 Z-버퍼를 사용한 프로그램에 시프트된 더블 Z-버퍼를 더 추가하여서 모든 Z-버퍼의 픽셀 값을 평활화하여 앨리어싱을 줄이는 방법이다. 이런 처리 절차는 대비가 심한 두 픽셀 사의의 중간 값을 생성시켜서 픽셀간 색상대비를 완화시키는 효과로 앨리어싱을 줄인 부드러운 객체로 표현된다.

아래의 수식(1), (2), (3)은 예제로 사용한 음함수들이다. 이 함수들은 [3]에서 예제로 사용된 함수 중에서 비교적 앨리어싱이 많이 나타나는 예제로 skeleton으로 표현되었다. 이 함수로 표현된 곡면들은 앤티앨리어싱이 적용되지 않은 곡면들로 앨리어싱으로 인하여 매우 거칠어 보인다. 향후과제중의 하나로 앤티앨리어싱의 적용이 제안 되었었으며 저해상도의 음함수를 위한 앤티앨리어싱 방법이 필요한 예제들이었다. (그림 1)은 예제로 사용한 식(1)과 식(2), 식(3)의 함수로 표현된 음함수 곡면을 각각 나타낸 이미지들이다.

$$F = \min(G, TempF) \tag{1}$$

where,

$$G = |f_i| + |f_o|$$

$$f_i = F - 0.15 \sin\left(\frac{x}{0.1}\right) \sin\left(\frac{y}{0.1}\right) + 0.3 \sin\left(\frac{x}{0.038}\right) \sin\left(\frac{y}{0.06}\right) \sin\left(\frac{z}{0.06}\right)$$

$$f_o = F + TempF$$

$$where \quad , F = x^2 + 2.0\,y^2 + 2.0\,z^2 - 2.0$$

$$TempF = x^2 + 2.0\,y^2 + 2.0\,z^2 - 2.0 + 0.2$$

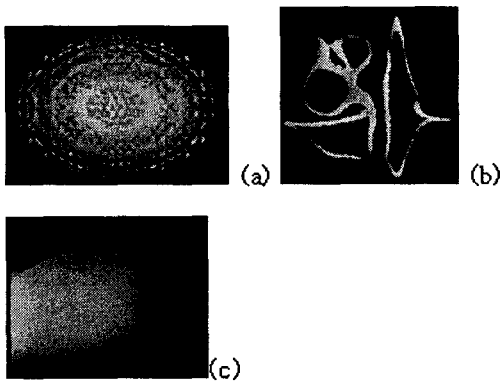
$$miro = |F| + |F1| \tag{2}$$

where ,

$$F = 3.2 - \frac{0.3}{\sqrt{Y - 0.6 + \sqrt{Z} + 0.1}} - \frac{0.3}{\sqrt{X - 0.5 + \sqrt{Z} + 0.1}} - \frac{0.3}{0.2 * \sqrt{x + 0.8 + \sqrt{y + 0.8 + \sqrt{z - 0.7 + 0.1}}}} - \frac{0.1}{\sqrt{x - 0.9 + \sqrt{y + \sqrt{z + 0.1}}}} - \frac{0.4}{\sqrt{x + 0.8 + \sqrt{y + \sqrt{z + 0.1}}}} - \frac{0.3}{\sqrt{x}\sqrt{y + 0.6 + \sqrt{z - 0.7 + 0.1}}} - \frac{0.4}{\sqrt{\sqrt{x + 0.3 + \sqrt{\sqrt{y - 0.6 + \sqrt{z - 0.4 + 0.1}}}}}}$$

$$F1 = 3.2 - \frac{0.3}{\sqrt{Y - 0.6 + \sqrt{Z} + 0.1}} - \frac{0.3}{\sqrt{x - 0.6 + \sqrt{z - 0.3 + 0.1}}} - \frac{0.3}{\sqrt{x + 0.7 + 2 * \sqrt{y + 0.8 + \sqrt{z - 0.7 + 0.1}}}} - \frac{0.1}{\sqrt{x - 1.2 + )y + 0.3 + \sqrt{z + 0.1}}} - \frac{0.4}{\sqrt{x + 0.8 + \sqrt{y + \sqrt{z - 0.4 + 0.1}}}} - \frac{0.3}{\sqrt{x + 0.15 + \sqrt{y + 0.7 + \sqrt{z - 0.9 + 0.1}}}} - \frac{0.4}{\sqrt{\sqrt{x + 0.5 + \sqrt{\sqrt{y - 0.7 + \sqrt{z - 0.5 + 0.1}}}}}}$$

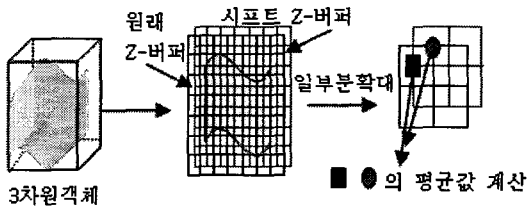
$$F = \sqrt{\sqrt{(x + y^3) + z^2}} \tag{3}$$




(그림 1) 사용된 음함수 곡면 예제들 :
(a) 수식1로 표현한 음함수 곡면 이미지,
(b) 수식2로 표현한 음함수 곡면 이미지,
(c) 수식3으로 표현한 음함수 곡면 이미지

더블 Z-버퍼 앤티앨리어싱 방법의 처리 절차는 다음과 같다.

먼저, 원래의 함수 값에 의한 Z-버퍼와 x,y,z 축으로 각각 0.5픽셀씩 움직여준 값에 의해 생성된 함수에 의한 또 다른 Z-버퍼를 만든다. 이렇게 만들어진 Z-버퍼와 원래의 Z-버퍼의 픽셀 값을 더한 후에 평균을 구하여 그 값을 새로운 Z-버퍼 값으로 출력한다. 이런 처리과정은 원래의 픽셀 사이의 값 차이를 줄여서 중간 색상의 픽셀을 추가하게 되므로 특히 경계면에서 나타나는 앨리어싱에 효과적이다. 더블 ㅋ-버퍼 앤티앨리어싱 방법을 그림으로 표현하면 다음 (그림 2)와 같다.



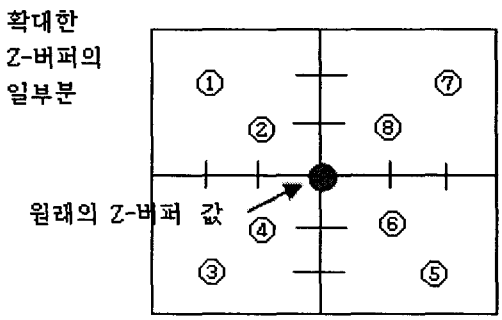

(그림 2) 더블 Z-버퍼 개념도

## 2.2 멀티 Z-버퍼 앤티앨리어싱

더블 Z-버퍼를 적용한 후에 앨리어싱이 줄어들었지만 만족할 만한 수준이 아닌 경우가 존재한다. 이 경우에 더블 Z-버퍼의 개념을 확대하여 즉, 시프트의 범위를 축소시키고 이에 따라서 Z-버퍼의 사용개수는 늘어난 멀티 Z-버퍼 앤티앨리어싱 방법을 제안한다. 시프트 Z-버퍼를 사용한 앤티앨리어싱 방법이 Z-버퍼의 개수를 늘릴 경우 앤티앨리어싱 효과가 증대 되는 방법임을 일반화시키기 위해 여러 개의 Z-버퍼를 사용하여 적용한 구현결과를 보인다.

시프트 멀티 Z-버퍼 앤티앨리어싱 방법은, 원래의 함수 값에서 각각 값에 1/3을 시프트한 값을 더한 여덟 개의 함수 값을 계산해 준다. 그 후 그 함수 값에 해당하는 Z-버퍼를 만든 후 원래의 Z-버퍼와 시프트된 여덟 개의 Z-버퍼를 합하여 모두 아홉 개의 Z-버퍼의 평균값을 구하여 표현한다. 여덟 개로 시프트한 Z-버퍼를 사용하였다는 것은 결국 여덟 개의 포인트로부터 값을 샘플링하여 렌더링 하는 방법이다. 이는 하나의 포인트 샘플링이 아닌 주변 값을 같이 샘플링하여 평활화하는 방법으로 샘플링하는 포인트를 많이 할수록 앤티앨리어싱 효과는 더 높은 것으로 나타난다.

원래의 프로그램처럼 하나의 포인트를 대푯값으로 하여 렌더링을 한 경우보다는 더블 Z-버퍼를 사용하여 두 개의 포인트 값을 고려하여 렌더링을 한 경우가 더 정확한 이미지로 표현 되었으며, 멀티 Z-버퍼를 사용하여 아홉 개의 포인트를 참조한 본 논문에서는 표현된 이미지가 매우 부드러워 보일 뿐 아니라 원래의 이미지에서는 표현되지 않았던 세부 정보를 더 많이 표현하고 있음을 알 수 있다




① ② ③ ④ ⑤ ⑥ ⑦ ⑧
: 시프트한 새로운 Z-버퍼 픽셀 값들

(그림 3) 멀티 Z-버퍼 개념도

## 3. 하이브리드 앤티 앨리어싱

포인트 샘플링에 의한 앨리어싱을 줄이기 위해서 제안한 멀티 Z-버퍼 앤티앨리어싱 방법은 시프트 범위를 세

분화 하여 여러 개의 버퍼를 사용할수록 앤티앨리어싱 효과가 증대되는 것이 사실이다. 또한 예제로 사용한 음함수 곡면의 경우 하나의 포인트로부터 샘플링하여 렌더링하는 경우보다 시프트된 Z-버퍼를 많이 사용할수록 주변의 포인트들을 더 많이 샘플링하여 렌더링을 하게 되므로 픽셀의 대표성이 증가하여 앤티앨리어싱이 적용된 이미지가 더 많은 정보를 포함한다고 할 수 있다. 즉, 주변의 포인트를 더 많이 샘플링 할수록 앤티앨리어싱 효과는 더 증가 되며, 더 정확한 3차원 객체를 표현하게 되므로 더 많은 시프트 멀티 Z-버퍼를 사용하는 것이 이미지의 질을 향상 시키는 것은 사실이다.

하지만 더 많은 시프트된 Z-버퍼를 계속 사용할 수 없는 이유는 계산에 따른 표현시간이 너무 많이 걸린다는 단점이 있다. 또한 시프트한 각 함수를 저장하기 위한 메모리 공간과 Z-버퍼를 위한 메모리 공간의 증가도 고려해야 할 사항이다. 따라서 더 많은 시프트된 멀티 Z-버퍼를 사용하여 앤티 앨리어싱 방법을 적용하는 것은 표현시간과 메모리 요구를 고려할 때 사용자의 요구에 따라 최적의 Z-버퍼 개수가 달라질 수 있다.

따라서 멀티 Z-버퍼를 사용한 것과 같은 앤티앨리어싱 효과를 나타내면서 계산시간과 필요로 하는 메모리를 줄일 수 있는 개선된 하이브리드 시프트 Z-버퍼 앤티앨리어싱 방법을 제안한다.

하이브리드 앤티앨리어싱 방법은 사용하는 Z-버퍼의 개수를 Z-버퍼의 픽셀 값에 따라 다르게 적용하는 방법이다. 즉, 1보다 작은 임의의 값 α와 β를 설정하고 각 Z-버퍼 픽셀 값과 이웃하는 픽셀 값의 차를 구하여 그 차이가 임의의 값 α 이상일 경우에는 색상대비에 의한 앨리어싱이 심하게 나타나므로 멀티 Z-버퍼를 사용하며, 차이가 임의의 값 α와 β 사이의 값일 경우에는 더블 Z-버퍼를 적용하며, β보다 작은 값일 경우 원래의 Z-버퍼 값만을 적용한다. 단, 임의의 α와 β의 값은 α > β 이며, 0.0 <|α|, |β| < 1.0사이의 값을 가진다. 또한 음함수 곡면 예제에 따라 달라질 수 있는데 설정한 α와 β의 값에 따라 구현 결과와 성능에 밀접한 영향을 미치게 된다.

하이브리드 앤티앨리어싱 방법의 알고리즘은 다음과 같다.

```
for ( index = start_pixel; index == end_pixel; index++)
{ make a difference  between next_pixel of origin Z-Buffer; }

α= arbitrary constant;
β= arbitrary constant;

for ( index = start_pixel; index == end_pixel; index++)
{
 if (a difference of origin Z-Buffer > α)
        {  multi  Z-Buffer  anti-aliasing Algorithm(); }
 else if (β< a difference of origin Z-Buffer < α)
{ double Z-Buffer anti-aliasing Algorithm(); }
else
{ use only origin Z-Buffer Algorithm(); }
```

## 4.성능 분석 및 구현 결과

이 장에서는 제안하는 하이브리드 앤티 앨리어싱 방법과 기존에 더블 Z-버퍼를 사용한 방법과 멀티 Z-버퍼를 사용한 앤티앨리어싱 방법의 효율성을 분석 비교하고 구현 결과를 보인다.

구현환경은Pentium 4, 2.8GHz, 512MB RAM, 그래픽카드는 GForce MX 4이며, Visual C++을 사용하였다. 사용된 음함수 곡면 예제는 해상도 150 * 150으로 일반적인 음함수 곡면에 비해 저해상도로 표현되었다. 따라서 필요한 메모리나 계산시간은 이 해상도를 기준으로 분석하였다.

먼저, 음함수 곡면을 생성하기 위하여 사용되는 Z-버퍼의 수에 따라 필요한 메모리를 비교해보면, 앤티앨리어싱을 적용하지 않은 경우, 하나의 음함수 곡면 예제에 필요한 메모리는 Z-버퍼를 위해서 더블형 2차원 배열 150*150크기를 요구하게 되므로 8바이트*150*150= 0.18MB가 필요하다. 더블 Z-버퍼를 적용할 경우, 같은 크기의 Z-버퍼를 하나 더 필요하게 되므로 0.36MB,

8개의 Z-버퍼를 적용할 경우 1.44MB가 필요하다. 즉 추가되는 시프트 Z-버퍼의 개수를 n이라고 가정하면 필요한 메모리는 0.18nMB가 된다.

실행시간을 비교해 보면, Z-버퍼를 생성하기 위한 시간과 음함수 곡면을 표현하기 위한 시간을 더한 시간이 전체 계산시간에 가장 큰 영향을 미치는 요소이다. 따라서 이를 적용 앤티애리어싱 방법별로 분석해 보면, 앤티앨리어싱 적용 전에는 Z-버퍼를 처리하기 위한 반복문을 150^2회 만큼 수행하고 음함수 곡면을 생성하기위한 반복 횟수는 150^3만큼 반복하게 된다. 즉, 150^5이라는 프로그램에서의 실행횟수를 보인다. 앤티앨리어싱 적용 전의 기본적인 계산시간을 m이라고 하면, 더블 Z-버퍼를 적용한 방법은 2m만큼의 계산시간을 요구하며, 멀티 Z-버퍼의 경우 8개의 시프트 Z-버퍼를 사용하면 8m만큼의 반복문을 실행하게 된다.

따라서 기존에 제안한 앤티앨리어싱 방법들은 추가하는 Z-버퍼의 수에 따라 메모리나 실행시간이 비례하는 것으로 나타났다. 하지만 구현하고자 하는 음함수 곡면 예제가 다른 경우에도 필요로 하는 메모리나 실행시간에는 차이가 없이 동일하다.

<표 1> 앤티앨리어싱 방법에 따른 메모리 필요량 및 실행시간 비교

| 앤티앨리어싱 방법 | 필요 메모리 | 실행 시간 (반복횟수) |
|---|---|---|
| 적용 전 | 0.18MB | 1505 |
| 더블 Z-버퍼 사용 | 0.36MB | 15010 |
| 멀티(8개) Z-버퍼 사용 | 1.44MB | 15040 |

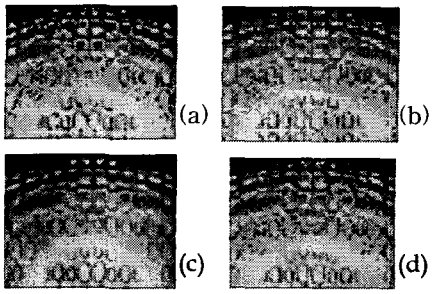
하이브리드 앤티앨리어싱 방법을 적용하면, 음함수 곡면이 다른 경우 필요한 메모리나 실행시간이 각각 다르게 나타나게 된다. 앨리어싱이 많이 나타나는 예제의 경우가, 많은 시프트 Z-버퍼를 필요로 하므로 더 많은 메모리와 실행시간을 요구하게 된다. 본 연구에서 사용한 음함수 예제는 비교적 앨리어싱이 많이 나타나서 상대적으로 앤티앨리어싱 방법을 적용 했을 경우 앨리어싱이 제거되어 가시적으로 부드럽게 표현되는 효과가 두드러지는 예제를 선택하여 구현하였다. 좌우 주변 픽셀과의 차이에 따른 적용 버퍼의 수를 결정하기 위하여 임의의 $|\alpha| = 0.1$, $|\beta| = 0.05$로 설정하였다.

이런 조건에 의하여 먼저 첫번째 예제 이미지의 경우 전체 Z-버퍼 픽셀 수 22,500에서 주변 픽셀과의 차이가 0.05미만이어서 앤티앨리어싱을 적용하지 않은 픽셀 수는 16,505이며 멀티 Z-버퍼를 적용한 픽셀 수는 2784개 ,더블 Z-버퍼를 적용한 픽셀 수는 3211개로 나타났다. 예제 이미지2의 경우 앤티앨리어싱을 적용하지 않은 픽셀 수는 19,981이며 멀티 Z-버퍼를 적용한 픽셀 수는 1018개 ,더블 Z-버퍼를 적용한 픽셀 수는 1501개이고 예제 이미지3의 경우 앤티앨리어싱을 적용하지 않은 픽셀 수는 21,980이며 멀티 Z-버퍼를 적용한 픽셀 수는 95개 ,더블 Z-버퍼를 적용한 픽셀 수는 425개이다. 실험 결과 앨리어싱이 많이 나타나는 예제 이미지1의 경우 멀티 Z-버퍼나 더블 Z-버퍼를 사용한 픽셀 수가 앨리어싱이 적은 예제 이미지3보다 상대적으로 많은 것으로 나타났다.

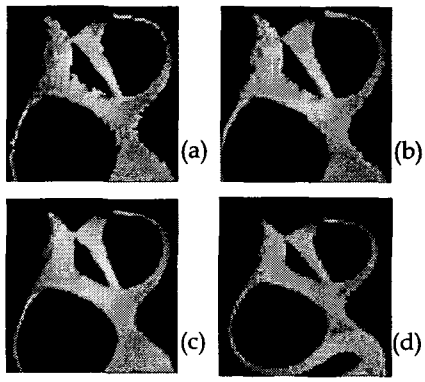
이를 바탕으로 사용한 세가지 음함수 곡면 예제별로 필요한 메모리와 실행시간은 다음 <표 2>와 같이 분석 할 수 있다.

<표 2> 하이브리드 앤티앨리어싱 적용 시 음함수 곡면 예제 별 필요한 메모리 및 계산시간

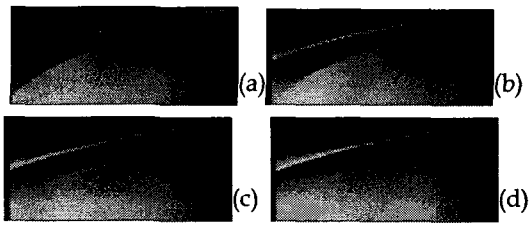
| 음함수 곡면 | 필요한 메모리 | 실행시간 (반복횟수) |
|---|---|---|
| 예제 이미지 1 | 멀티 Z-버퍼 수= 2784<br>더블 Z-버퍼 수= 3211<br>총 메모리 = 약 0.362MB | 45199 |
| 예제 이미지 2 | 멀티 Z-버퍼 수= 1018,<br>더블 Z-버퍼 수 = 1501<br>총 메모리 = 약 0.25MB | 31127 |
| 예제 이미지 3 | 멀티 Z-버퍼 수=95,<br>더블 Z-버퍼 수 = 425<br>총 메모리 = 약 0.188MB | 23590 |



(그림 4) 음함수 곡면 예제 이미지1 : (a) 원래 이미지, (b) 더블 Z-버퍼 사용 이미지, (c) 멀티 Z-버퍼 사용 이미지, (d) 하이브리드 방식의 앤티앨리어싱 방법 사용 이미지



(그림5) 음함수 곡면 예제 이미지2 : (a) 원래 이미지, (b) 더블 Z-버퍼 사용 이미지, (c) 멀티 Z-버퍼 사용 이미지, (d) 하이브리드 방식의 앤티앨리어싱 방법 사용 이미지



(그림 6) 음함수 곡면 예제 이미지3 : (a) 원래 이미지, (b) 더블 Z-버퍼 사용 이미지, (c) 멀티 Z-버퍼 사용 이미지, (c) 하이브리드 방식의 앤티앨리어싱 방법 사용 이미지

<표 1>에서 보여준 기존의 방법을 적용한 경우와 <표 2>의 하이브리드 앤티앨리어싱 방법의 메모리 필요량과 실행시간(반복 횟수)을 분석해 보면 가장 앨리어싱이 많은 예제 이미지 1의 경우에 조차도 멀티 Z-버퍼를 적용하였을 때 필요한 메모리 0.44MB에 못 미치는 약 0.362MB를 필요로 하며, 예제 이미지3의 경우에는 앤티앨리어싱 적용 전에 필요한 메모리와 비교해도 8KB 정도의 추가 메모리 만을 요구하게 된다.

또한 제안한 방법은 경우에 따라서 이미지의 질과 실행시간사이의 관계를 임의의 수치인 α와 β의 값을 사용하여 조절하여 더블이나 멀티 Z-버퍼의 개수를 유동성있게 적용 할 수 있는 융통성을 지닌다.

(그림 4) ~ (그림 6)은 예제로 사용한 세가지 음함수를 각각 원래의 이미지와 더블 Z-버퍼를 사용하여 앤티 앨리어싱을 적용한 이미지, 멀티 Z-버퍼를 사용하여 앤티앨리어싱을 구현한 이미지, 그리고 하이브리드 앤티앨리어싱을 적용하여 구현한 이미지를 확대 비교하였다

## 5. 결론

3차원 만화 캐릭터나 물의 곡면과 같은 유연한 곡면을 표현하기 위하여 Blinn에 의해 처음 만들어진 음함수 곡면은, 분자 안의 일정한 에너지 곡면을 모델하기 위한 것이었으나[4] 현재 건축물 시뮬레이션이나 애니메이션, 의료 영상 시뮬레이션 등 다양한 분야에서 활용되고 있는 추세이다. 음함수 곡면이 앨리어싱이 없는 부드러운 곡면으로 표현되기 위해서는 고해상도를 필요로 하지만[7] 본 논문에서는 저해상도의 포인트 샘플링으로 표현된 음함수로 만들어진 곡면에 나타나는 앨리어싱을 줄이기 위해서 시프트 Z-버퍼 앤티앨리어싱 방법을 확장한 하이브리드 앤티앨리어싱 방법을 제안하였다.

기존에, 복잡한 함수를 사용하여 연산시간이 오래 걸리는 예제의 경우 간단한 앤티앨리어싱 방법으로 시프트 더블 Z-버퍼를 적용하거나 멀티 Z-버퍼를 프로그램에 적용하면 효과적으로 앨리어싱을 줄여 부드러운 이미지를 생성하는 것으로 나타났다. 구현 결과를 보면 곡면에 픽셀 사이의 대비가 심하게 나타나서 색상 값으로 인한 보기 싫은 앨리어싱을 나타내는 함수에 중간 정도의 픽셀 값으로 표현함으로써 픽셀 간 대비를 완화시키는 효과가 있음이 밝혀졌다.

하지만 더 많은 시프트된 Z-버퍼를 계속 사용

할 수 없는 이유는 계산에 따른 표현시간이 너무 많이 걸린다는 단점이 있다. 또한 시프트한 각 함수를 저장하기 위한 메모리 공간과 Z-버퍼를 위한 메모리 공간의 증가도 고려해야 할 사항이다.

따라서 멀티 Z-버퍼를 사용한 것과 비슷한 앤티앨리어싱 효과를 나타내면서 계산시간과 필요로 하는 메모리를 줄일 수 있는 개선된 하이브리드 시프트 Z-버퍼 앤티앨리어싱 방법을 제안하였다. 하이브리드 앤티앨리어싱 방법은 기존 방법과 비교하여 메모리 사용량이나 실행시간에 있어서 매우 효율적이다

## 참고문헌


[1] 김학란, 박화진, "복셀로 표현된 임플리시트 곡면을 위한 시프트(shifted) 더블 Z-버퍼 앤티 앨리어싱," 멀티미디어학회 논문지, 7권,1호, 44-53, 2004.

[2] 김학란, 박화진, "3차원 임플리시트 곡면 렌더링을 위한 시프트(shifted) 멀티 Z-버퍼 앤티 앨리어싱 연구," 멀티미디어학회 논문지, 8권,2호, 249-257, 2005.

[3] Satoshi Tanaka, Tomoharu Nakamura, Miharu Ueda, Hiroaki Yamamoto, Kisou Shino, "Ap-plication of the stochastic sampling method to various implicit surfaces," Computers & Graphics 25, 441-448, 2001.

[4] Matthew Jondrow. "A Survey of animation related Implicit surfaces Papers", AT in Computer Graphic s, 227-237, 2000

[5] Bloomenthal, Bajaj, Blinn, Cani-Gaxcuel, Rockwood, Brian Wyill, Geoff Wyvill, Intro-duction to Implicit Surfaces, MORGAN KAUFMANN PUBLISHERS, INC., 1997.

[6] Satoshi Tanaka, Tomoharu Nakamura, Miharu Ueda, Hiroaki Yamamoto, Kisou Shino, "Ap-plication of the stochastic sampling method to various implicit surfaces", Computers & Graphics 25, 441-448, 2001

[7] Nilo Stolte Homepage http://nilo.stolte.free. fr/graphi cs.html

[8] 김병욱, 박우찬, 양성봉, 한탁돈, "RUF 버퍼를 이용한 간단하고 효율적인 안티알리아싱 기법," 정보과학회 논문지, 제30권 제3·4호, 205-212, 2003.



**김 학 란**

1983년: 숙명여자대학교 전산학과(학사)
2003년: 숙명여자대학교 정보통신대학원(석사)
2004년 - 현재 : 숙명여자대학교 대학원컴퓨터과학 박사과정
한성대학교 멀티미디어공학과 겸임교수
관심분야 : 2D, 3D 컴퓨터 그래픽스, 게임, 가상현실, 애니메이션

**박 화 진**

1983년: 숙명여자대학교 전산학과(학사)
1989년: 숙명여자대학교 대학원 전산학과(석사)
1997년: Arizona State Univ. Computer Science 컴퓨터 그래픽전공(공학박사)
1997년 - 1998년 : 삼성 SDS 연구소 선임 연구원
1998년 - 2000년 : 평택대학교 전임강사
2000년 - 현 재 : 숙명여자대학교 멀티미디어과학과 교수
관심분야 : 컴퓨터 그래픽, 게임, 3D 모델링, 가상현실, 멀티미디어