Journal of Korea Multimedia Society Vol. 7, No. 11, November 2004(pp. 1550-1555)

시간일관성을 이용한 그림자 텍스처 생성방법

오경수[†], 신병석[††]

요 약

그림자는 화상의 사실성을 높이며 물체들 사이의 공간적 관계를 유추할 수 있도록 해준다. 기존의 그림자 텍스처 생성 방법에서는 한 물체의 텍스처를 생성하기 위해서 다른 모든 물체들을 렌더링 해야만 하므로 N개의 물체가 있을 때 모든 물체들의 그림자 텍스처를 만들기 위해서는 $O(N^2)$의 시간이 필요했다. 이 논문에서는 깊이버퍼를 그림자 텍스처 생성에 사용하여 각 물체들에 대해서 상수 시간에 그림자 텍스처를 생성하는 알고리즘을 제안한다. 또한 시간 일관성을 이용해서 속도를 더욱 향상시키는 방법을 제안한다. 정적인 물체의 깊이 값은 시간이 지나도 변화가 없다. 이 값들을 재사용하여 정적인 물체들은 되도록 그리지 않고 동적인 물체들만 렌더링 함으로써 깊이버퍼와 그림자 텍스처를 효율적으로 생성할 수 있다. 이 방법의 비용은 움직이는 물체의 개수에만 비례한다.

Shadow Texture Generation Using Temporal Coherence

Kyoung-su Oh[†], Byeong-Seok Shin[††]

ABSTRACT

Shadows increase the visual realism of computer-generated images and they are good hint for spatial relationships between objects. Previous methods to produce a shadow texture for an object are to render all objects between the object and light source. Consequently entire time for generating shadow textures between all objects is $O(N^2)$, where N is the number of objects. We propose a novel shadow texture generation method with constant processing time for each object using shadow depth buffer. In addition, we also present method to achieve further speed-up using temporal coherence. If the transition between dynamic and static state is not frequent, depth values of static objects does not vary significantly. So we can reuse the depth value for static objects and render only dynamic objects.

Key words: Shadow(그림자), Shadow Texture(그림자 텍스처), Depth Buffer(깊이 버퍼), Temporal Coherence(시간 일관성)

1. 서 론

그림자는 화상의 사실성을 높이며 물체들 간의 공간적 관계를 유추할 수 있게 한다. 이러한 효과 때문에 그림자를 실시간으로 생성하기 위한 많은 연구가 있었다. 그러나 아직까지 실시간으로 모든 물체들 간의 그림자를 생성하는 경우는 드물다.

그림자 텍스처 (shadow texture) 방법은 많이 사용되는 실시간 그림자 생성 방법들 중 하나이다. 그림자 텍스처는 특정 물체의 그림자 정보를 담고 있는 화상 텍스처로서 광원을 투영점 (center of projection)에 놓고 투영해서 만들 수 있다. 그림자 텍스처의 각 픽셀 값은 그 점이 광원에서 직접 보이는지 여부를 나타낸다. 즉, 텍스처에 해당하는 물체가 그 픽셀을 통해서 보이는 물체들 중에 광원에서 가장 가까운 물체이면 조명색(흰색)을 저장하고 그렇지

※ 교신저자(Corresponding Author) : 오경수, 주소 : 서울특별시 동작구 상도동 1-1(156-743), 전화 : 02)828-7261, FAX : 02)822-3622, E-mail : oks@ssu.ac.kr
접수일 : 2004년 8월 26일, 완료일 : 2004년 10월 26일
[†] 숭실대학교 미디어학부 전임강사
[††] 정회원, 인하대학교 컴퓨터공학부 조교수
(E-mail : bsshin@inha.ac.kr)
※ 본 연구는 숭실대학교 교내연구비 지원으로 이루어졌음.

않으면 그림자색(검정색)을 저장한다. 그림자 텍스처가 그림자를 표현할 수 있는 좋은 수단이기는 하지만 생성 시간이 물체 개수의 제곱에 비례하는 문제가 있다. 지금까지의 그림자 텍스처 생성 방법들은 물체와 광원 사이의 모든 물체들을 그려야 한다. 결과적으로 N개 물체들의 그림자 텍스처를 만들기 위해서는 $O(N^2)$의 시간이 필요하다.

Nguyen은 그림자를 드리우는 물체(shadow caster)와 그림자가 드리워지는 물체(shadow receiver) 가 알려진 상태에서 그래픽스 하드웨어를 사용해서 그림자 텍스처를 만드는 방법을 제안하였다[1]. Herf는 그림자가 드리워질 수 있는 물체의 수를 제한한 상태에서 부드러운 그림자(soft shadow)를 생성하는 방법을 설명하였다[2]. Soler는 FFT(Fast Fourier Transform)을 사용해서 부드러운 그림자를 표현하는 그림자 텍스처를 생성하는 방법을 제안하였다[3]. 이러한 방법들은 물체의 개수가 많아질수록 속도가 급격히 느려지기 때문에 물체의 개수가 많은 경우에는 적합하지 않다. 이들 방법들에서 속도가 느려지는 이유는 그림자가 드리워지는 물체들과 광원 사이에 있는 모든 물체들을 찾아내서 그리기 때문이다[2].

본 논문에서 제안하는 방법은 그림자 깊이 버퍼(shadow Z-buffer) 를 사용해서 그림자가 드리워지는 물체만을 렌더링 하도록 함으로써 그림자 텍스처를 상수시간에 생성한다. 그림자 깊이 버퍼는 광원까지 가장 가까운 물체까지의 거리를 저장한다. 그림자 깊이 버퍼를 일반 깊이 버퍼로 지정하고 깊이 버퍼 알고리즘을 이용하여 렌더링하면 물체의 가장 가까운 부분의 색이 변한다. 즉, 그림자 텍스처를 그림자 색으로 초기화 하고 물체들을 조명 색으로 렌더링하면 그림자가 아닌 부분들만 조명색으로 변한다. 이렇게 하면 한번의 렌더링으로 그림자 텍스처를 만들 수 있다.

시점이 고정되었을 때 현재 프레임과 이전 프레임의 깊이값과 컬러값은 거의 차이가 없다. 이런 성질(시간 일관성) 을 이용하면 가시성 검사 시간을 줄일 수 있다[4-9]. 마찬가지로 광원이 고정되어 있을 때 현재 프레임과 이전 프레임 사이에 그림자는 거의 변하지 않는다. 따라서 시간 일관성을 이용하여 그림자 깊이 버퍼 생성과 그림자 텍스처 생성 시간을 줄일 수 있다. 움직이지 않는 물체들의 그림자 깊이버퍼 값과 그림자 텍스처들을 최대한 재사용하고 움직이는 물체들만을 렌더링 하면 현재 프레임의 그림자 깊이버퍼와 그림자 텍스처를 생성할 수 있다. 결과적으로 N_{dyn} 를 움직이는 물체들의 개수라고 할 때 제안하는 방법은 $O(N_{dyn})$의 시간 복잡도를 가진다.

이후 논문의 구성은 다음과 같다. 2절에서는 그림자 깊이버퍼를 이용해서 그림자 텍스처를 효율적으로 생성하는 방법에 대해서 설명한다. 3절에서는 시간일관성을 이용한 추가적인 속도향상 방법을 다룬다. 4절에서 실험결과를 설명하고 결론을 내린다.

2. 그림자 깊이버퍼를 이용한 그림자 텍스처 생성

이전의 그림자 텍스처 생성 방법들은 각각의 그림자가 드리워지는 물체들에 대해서 그림자 드리우는 물체들을 렌더링하는 방식이다. 즉 광원과 물체 사이에 있는 모든 물체들을 렌더링 해야 한다. 그림 1은 광원과 물체들이 있는 장면(a)과 그림자 텍스처를 생성하는 기존 방법(b)을 설명한다. 이 논문에서는 그림자 깊이 버퍼를 사용해서 효율적으로 그림자 텍스처를 생성하는 방법을 제안한다. 이 방법에서는 어떤

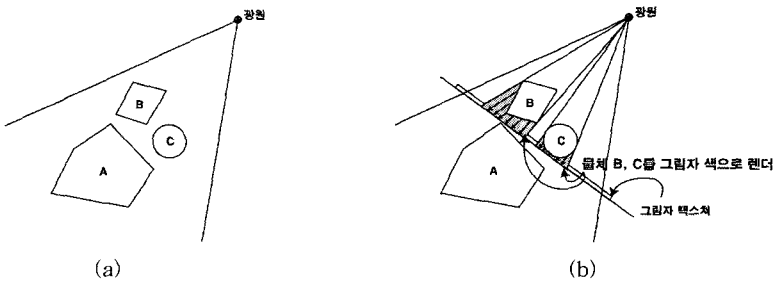


그림 1. 이전의 그림자 텍스처 생성 방법 (a) 광원과 그림자 텍스처가 필요한 물체들 (b) 물체 A의 그림자 텍스처를 만들기위 광원과 A 사이에 있는 물체들(B, C)을 렌더한다.

물체에 드리워지는 그림자 정보를 나타내는 그림자 텍스처를 생성할 때 그 물체 만을 렌더링 해서 만드는 것이 가능하다.

우선 그림자 깊이버퍼를 생성한다. 그림자 깊이버퍼를 생성하기 위해서는 광원의 위치를 관측점(view point) 으로 하여 깊이 버퍼 알고리즘으로 모든 물체들을 그린다. 결과로 생성된 깊이 버퍼값을 그림자 깊이버퍼에 저장한다. 그림자 깊이 버퍼의 각 픽셀들은 광원으로부터 그 픽셀을 통해서 보이는 가장 가까운 물체까지의 거리를 저장한다.

그림자 깊이버퍼를 이용하고 깊이버퍼 알고리즘을 한번 더 적용함으로써 그림자 텍스처를 효율적으로 구할 수 있다. 우선 관측점을 광원의 위치와 같도록 지정하고 일반 깊이버퍼에 그림자 깊이버퍼의 값을 복사한다. 각 물체들을 렌더링 할 때 결과가 그림자 텍스처에 그려지도록 설정하고, 텍스처를 그림자 색으로 초기화 한다. 그리고 깊이 버퍼 알고리즘을 이용하여 그 물체를 그린다. 그릴 때는 조명색을 사용한다. 그림자 깊이버퍼가 광원으로부터 가장 가까운 물체까지의 거리를 저장하고 있으므로 렌더링 되는 물체가 차지하는 픽셀들 중에서 물체가 광원에 가장 가까운 부분에 해당하는 픽셀들의 색만 조명색으로 바뀌게 된다. 그림 2는 깊이버퍼를 이용해서 그림자 텍스처를 생성하는 각 단계를 설명한다.

3. 시간 일관성을 이용한 가속화 방법

많은 경우 광원과 물체들은 대부분의 프레임동안 움직이지 않는다. 또한, 이전 프레임에 움직이지 않는 물체들은 다음 프레임에서도 움직이지 않을 확률이 높다. 이 논문에서는 이러한 시간적인 일관성을 이용해서 그림자 텍스처 생성 속도를 향상시키는 방법을 제안한다. 앞 절에서 그림자 깊이버퍼를 사용해서 그림자 텍스처를 생성하는 방법을 설명하였다. 이 절에서는 시간일관성을 이용해서 그림자 깊이 버퍼와 그림자 텍스처의 생성속도를 높이는 방법을 설명하겠다.

3.1 그림자 깊이버퍼 생성의 가속화

실시간 프로그램이 진행되는 동안 모든 물체들은 동적상태(dynamic state) 또는 정적상태(static state) 중 한가지를 가진다. 만약 두 상태간의 변화가 자주 일어나지 않는다면 정적인 물체들만의 깊이 값은 매 프레임 마다 거의 변하지 않는다. 또한 만약

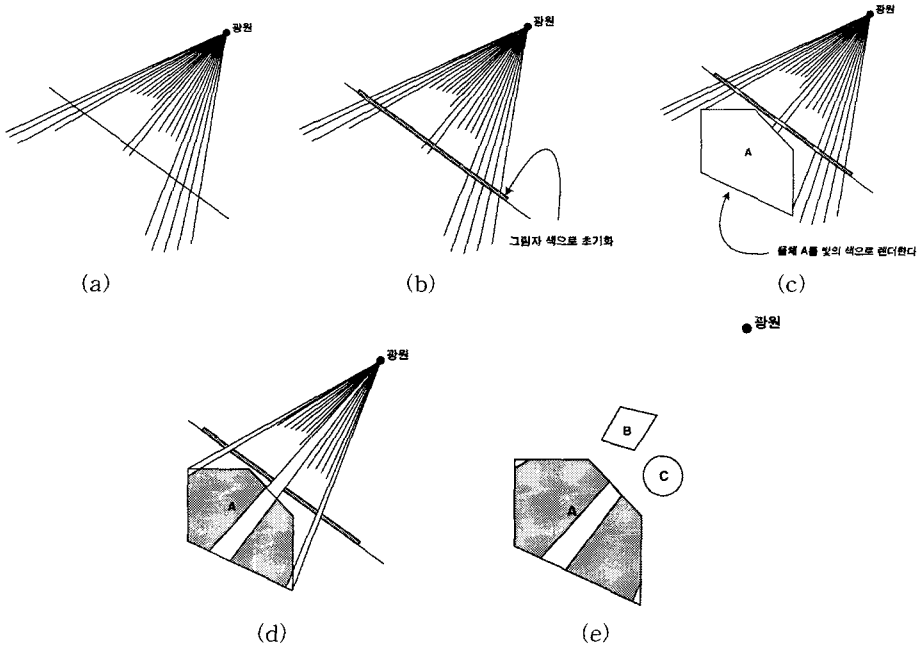


그림 2. 그림자 깊이버퍼를 이용한 그림자 텍스처 생성방법 (a) 그림자 깊이버퍼의 내용, 선들의 길이가 깊이값을 나타낸다. 그림자 텍스처를 그림자 색(검정색)으로 초기화 (c) 그림자 텍스처가 필요한 물체(A)를 빛의 색(흰색)으로 렌더한다 그림자 텍스처를 사용해서 생성한 그림자 (e) 모든 물체에 그림자를 입혀서 그린 장면

정적인 물체들만의 깊이 값이 계산되어 있다면 나머지 동적인 물체들만 렌더링 하여 모든 물체들의 깊이 값을 구할 수 있다. 이전 연구에서 우리는 시간 일관성을 이용한 가시성 검사의 가속화 방법을 제안하였다[4]. 그 방법은 시점이 고정되었을 때 정적인 물체들의 깊이 값들을 효율적으로 계산하여 깊이버퍼 방법의 속도를 향상시키는 것이다. 이 방법을 그림자 깊이 버퍼 생성에 적용한다.

모든 물체들은 이전 프레임과 현재 프레임에서 움직였는지 여부에 따라 SD, DD, DS, SS네가지 상태로 구분된다. 예를 들어, 이전 프레임에서 정적이고 현재 프레임에서 동적인 물체는 SD라는 집합에 속한다. 그림 3은 물체들이 가질 수 있는 4가지 상태의 예를 보여준다. 정적인 물체들의 깊이 값을 저장하는 버퍼를 정적 깊이 버퍼라고 하고 모든 물체를 렌더링 했을 때의 깊이값을 저장하는 버퍼를 렌더링 깊이버퍼라고 부르겠다.

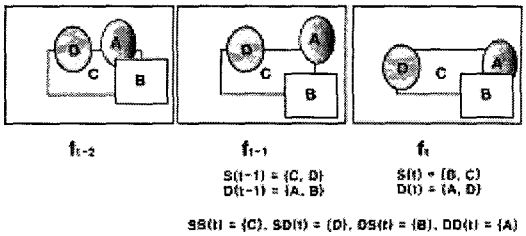


그림 3. 이전 두 프레임동안의 움직임 정보(mobility)에 물체들을 4가지로 분류한 예

시간 일관성을 이용한 그림자 깊이버퍼 생성은 다음 3단계로 구성된다.

(1) 이전 프레임의 깊이 값을 이용해서 현재 프레임에서 정적인 물체의 깊이 값을 구한다.
(2) 현재 프레임에서 정적인 물체의 깊이 값을 정적 깊이버퍼에 저장한다.
(3) 현재 프레임에서 동적인 물체들을 깊이 버퍼 알고리즘으로 렌더링한다.

그림 4는 시간 일관성을 이용한 그림자 깊이버퍼 생성 과정동안의 그림자 깊이 버퍼와 렌더링 깊이버퍼의 내용이 어떻게 변하는지를 보여준다.

첫번째 단계는 다시 다음과 같이 4개의 단계로 세분된다.

1-1. 현재 프레임과 이전 프레임에서 동적인 물체들을 지운다.(DD)

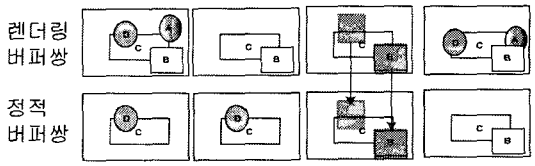


그림 4. 알고리즘의 각 단계에서의 렌더링 깊이버퍼와 정적 깊이버퍼의 내용 (a) 현 프레임을 그리기 전 (b) 정적 물체들의 깊이값을 구한 후 (c) 렌더링 깊이버퍼로부터 정적 깊이버퍼로의 복사 (d) 동적인 물체들을 렌더

1-2. 현재 프레임에서 동적이고 이전 프레임에서 정적인 물체들을 지운다.(SD)
1-3. 현재 프레임과 이전 프레임에서 정적인 물체들을 다시 그린다.(SS)
1-4. 현재 프레임에서 동적이고 이전 프레임에서 동적인 물체들을 다시 그린다.(DS)

그림 5는 첫번째 단계인 현재 정적인 물체들의 깊이값을 구하기 위한 네가지 부분 단계들을 그림으로 설명한다. 시작하기 전에는 버퍼들에는 이전 프레임의 깊이값들이 저장되어 있다. 첫번째와 두번째 부분 단계는 현재 프레임에서 동적인 물체들을 지우기 위한 것이고 세번째와 네번째 부분 단계는 동적인 물체를 지우는 과정에서 함께 지워진 정적인 물체들을 다시 그려주기 위한 것이다.

첫번째 부분 단계는 정적 깊이버퍼로부터 렌더링 깊이버퍼로 복사하는 것으로 두 프레임동안 동적인 물체들(DD)을 지운다. 두번째 부분 단계에서는 현재 프레임에서 동적이고 이전 프레임에서 정적인 물체들(SD)을 무한대 값으로 채워서 지운다. 세번째 부분 단계에서는 이전 두 프레임동안 정적인 물체들(SS)을 다시 그려준다. 이들은 첫번째 작은 단계에서는 지워지지 않는다. 그러므로 이들 물체들 중 이전 프레임에 정적이고 현재 프레임에 동적인 물체와 화상영역(extent)이 겹치는 물체들만을 다시 그려준다.

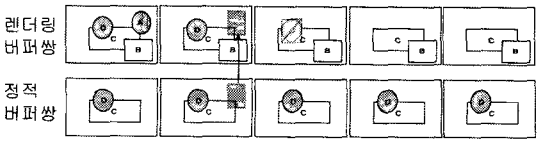


그림 5. 첫번째 단계의 네가지 작은 단계에서의 렌더링 깊이버퍼와 정적 깊이버퍼의 내용 (a) 시작하기전 (b) DD를 지운다. (c) SD를 지운다. (d) SS를 그린다. (e) DS를 그린다.

마지막 네번째 부분 단계에서는 현재 프레임에서 정적이고 이전 프레임에서 동적인 물체(DS)들 중 현재 프레임에 동적인 물체들과 화상영역이 겹치는 물체들을 다시 그려준다.

3.2 시간일관성을 이용한 그림자 텍스처 생성

광원이 고정되어 있을 경우 광원에서 보았을 때 움직이는 물체들과 겹치지 않는 고정된 물체들은 그림자 텍스처를 다시 생성 할 필요가 없다. 즉, 움직이는 물체와 고정된 물체들중 움직이는 물체와 겹치는 것들에 대해서만 그림자 텍스처를 다시 생성하면 된다. 현재 프레임에서 동적인 물체들과 그들과 겹치는 정적인 물체들은 그림자 깊이 버퍼를 생성하는 과정에서 이미 찾아 냈기 때문에 이들에 대해서만 2절에서 설명한 그림자 버퍼를 이용한 그림자 텍스처 생성 방법을 적용하면 된다.

5. 구현 및 실험 결과

펜티엄IV 2.66GHz, 1.00GB RAM, ATI Radeon9800 XT가 장착된 컴퓨터에서 이전방법과 제안된 방법을 구현하였다. 실험을 위해서 구면 모델을 임의의 위치에 생성하고 임의의 움직임을 지정했다. 각 구면 모델은 2037개의 정점들로 구성된다. 그림 6은 실험중 생성된 화면의 일부이다.

이전 방법과 제안된 방법 중 시간 일관성을 적용하지 않은 경우의 렌더링 시간을 측정하여 비교하였다. 그림 7은 물체의 개수를 늘려가면서 렌더링 시간을 측정하여 비교한 것이다. 그래프에서 보는 바와 같이 이전 방법은 물체의 개수의 제곱에 비례하고 제안된 방법은 물체 개수에 비례한다.

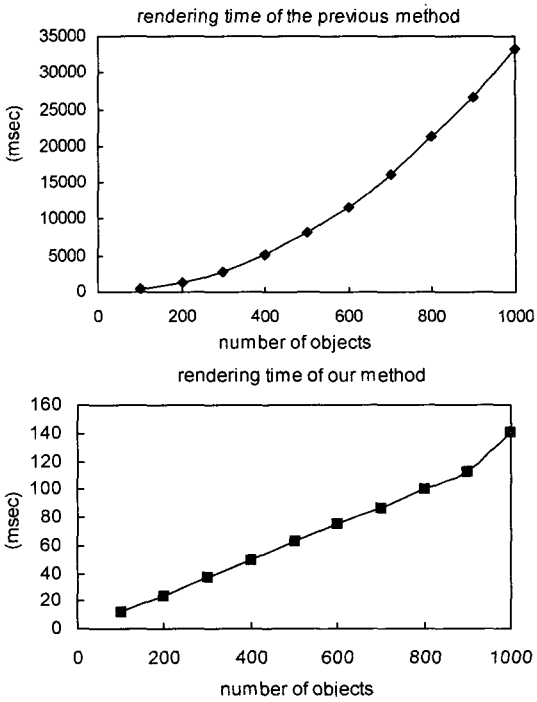
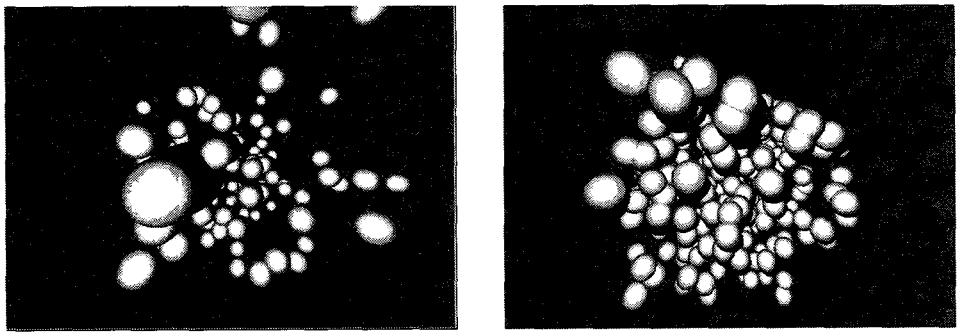


그림 7. 이전 방법과 제안된 방법의 렌더링 시간 비교

시간일관성을 적용했을 경우의 효과를 알아보기 위해 시간일관성을 적용한 방법과 적용하지 않은 방법의 렌더링 시간도 측정하여 비교하였다. 그림 8에서 보는 바와 같이 동적인 물체들의 비율이 낮아질수록 시간일관성을 이용한 속도향상 효과가 커진다. 동적인 물체가 전체 물체의 10%이면 속도는 23.4% 향상된다. 이러한 속도향상은 전체 렌더링 시간을 기준으로 계산한 것이다. 그림자 텍스처 생성 시간에서의 속도향상은 더 크다.



(a) (b)

그림 6. 실험에 얻어진 화상 중 일부 (a) 100개의 구모델을 그린 화상 (b) 300개의 구 모델을 그린 화상

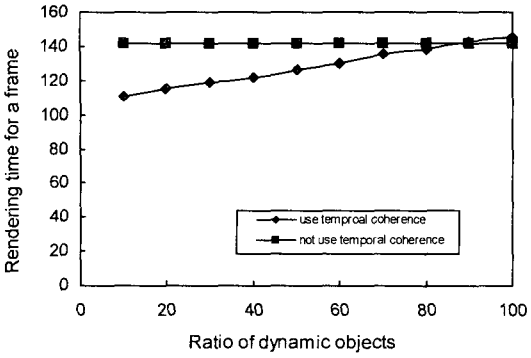


그림 8. 동적인 물체들의 비율이 달라지는 동안 시간일관성을 사용한 방법과 그렇지 않은 방법의 시간 비교

6. 결 론

이 논문에서는 각 물체들의 그림자 텍스처를 상수 시간에 생성하는 방법을 제안했다. 또한 움직이지 않는 물체들의 그림자 텍스처를 생성하지 않는 방법도 제안했다. 구현결과 제안된 방법은 전체 물체들의 개수에 선형적인 시간에 그림자 텍스처를 생성할 수 있었다. 제안된 방법의 속도는 물체의 개수의 제곱에 비례하는 그림자 텍스처 생성 속도를 물체의 개수에 선형적으로 비례하도록 개선했다. 또한 시간일관성을 이용하여 추가로 속도를 개선하였다. 추가의 속도 개선 방법에 의해 동적인 물체가 적을수록 속도가 개선된다.

참 고 문 헌

[1] Nguyen, H.H, Casting Shadows on Volumes, *Game Developer*, Vol. 6, No. 3, pp. 44-53, 1999.

[2] Herf, M, Efficient Generation of Soft Shadow Textures, CMU-CS-97-138, CS Dept, Carnegie Mellon U., May 1997.

[3] Soler, C. and Sillion, F, Fast Calculation of Soft Shadow Textures using Convolution, *SIGGRAPH*, pp. 321-332, 1998.

[4] Oh, K.S., Shin, B.S. and Shin Y.G, Mobility culling: an efficient rendering algorithm using temporal coherence, *Journal of Visualization and Computer Animation*, Vol. 12, No. 3, pp. 159-166, 2001.

[5] Moeller, T. and Haines, E, Real-Time Rendering, *A. K. Peters*, Natick, MA, pp. 157-158, 1999.

[6] Scaufer, G, Exploiting frame to frame coherence in a virtual reality system, *Proceedings of VRAIS*, pp. 95-102, 1996.

[7] Chapman, J., Calvert, T.W. and Dill, J, Spatialtemporal coherence in ray tracing, pp. 196204, 1990.

[8] Badt, S. Jr., Two algorithms for taking advantage of temporal coherence in ray tracing. *Visual Computer,* Vol. 4, No. 3, pp. 123132, 1988.

[9] Jevans, D, Object space temporal coherence for ray tracing, *Proceedings of Graphics Interface*, pp. 176183, 1992.

오 경 수

1994년 서울대학교 계산통계학과 학사
1996년 서울대학교 전산과학과 석사
2001년 서울대학교 전기컴퓨터공학부 박사
2001년~2002년 (주) 조이멘트 연구원
2003년~현재 숭실대학교 미디어학부 전임강사
관심분야 : 컴퓨터 그래픽스, 컴퓨터 게임, 실시간 렌더링, 화상기반 렌더링

신 병 석

1990년 서울대학교 공과대학 컴퓨터공학과
1992년 서울대학교 대학원 컴퓨터 공학과 석사
1997년 서울대학교 대학원 컴퓨터공학과 박사
2000년~현재 인하대학교 컴퓨터공학부 조교수. 인하대학교 전산정보원부원장.
관심분야 : 실시간 렌더링, 볼륨그래픽스, 의료영상