한국컴퓨터그래픽스학회
Korea Computer Graphics Society Vol. 28, No. 3, P. 113~123

헤드 마운티드 디스플레이를 위한 시간 제약 렌더링을 이용한 적응적 포비티드 광선 추적법

김영욱[O] 임인성[*]
서강대학교 컴퓨터공학과
{kimyu7, ihm[*]}@sogang.ac.kr

Adaptive Foveated Ray Tracing Based on Time-Constrained Rendering for Head-Mounted Display

Youngwook Kim[O] Insung Ihm[*]
Department of Computer Science and Engineering, Sogang University

요 약

광선 추적 기반의 렌더링은 래스터화 기반의 렌더링보다 훨씬 더 사실적인 이미지를 생성한다. 하지만 넓은 시야와 높은 디스플레이 갱신 속도를 요구하는 헤드 마운티드 디스플레이(HMD) 시스템을 대상으로 이를 구현할 때에는 여전히 많은 연산량으로 인하여 부담스럽다. 또한, HMD 화면에 고품질 이미지를 표시하기 위해서는 시각적으로 성가신 공간적/시간적 앨리어스를 줄이기 위해 픽셀당 충분한 수의 광선 샘플링을 수행해야 한다. 본 논문에서는 최근 Kim 등[1]이 제시한 선택적 포비티드 광선 추적법을 확장하여 주어진 HMD 시스템에서 고전적인 Whitted-스타일 광선 추적 수준의 렌더링 효과를 효율적으로 생성해주는 실시간 렌더링 기법을 제안한다. 특히, GPU의 광선 추적 하드웨어를 통한 가속과 시간 제한을 둔 렌더링 방법의 결합을 통하여 고속의 HMD 광선 추적에 적합한 사람의 시각 시스템에 매우 효율적인 적응적 광선 샘플링 방법을 제안한다.

Abstract

Ray tracing-based rendering creates by far more realistic images than the traditional rasterization-based rendering. However, it is still burdensome when implemented for a Head-Mounted Display (HMD) system that demands a wide field of view and a high display refresh rate. Furthermore, for presenting high-quality images on the HMD screen, a sufficient number of ray sampling should be carried out per pixel to alleviate visually annoying spatial and temporal aliases. In this paper, we extend the recent selective foveated ray tracing technique by Kim et al. [1], and propose an improved real-time rendering technique that realizes the rendering effect of the classic Whitted-style ray tracing on the HMD system. In particular, by combining the ray tracing hardware-based acceleration technique and time-constrained rendering scheme, we show that fast HMD ray tracing is possible that is well suited to human visual systems.

키워드: 실시간 광선 추적, 포비티드 광선 샘플링, 시간 제약 렌더링, 광선 추적 하드웨어.
Keywords: Real-time Ray Tracing, Foveated Ray Sampling, Time Constrained Rendering, Ray Tracing Hardware.

*corresponding author: Insung Ihm/Sogang University(ihm@sogang.ac.kr)

Received : 2022.06.11./ Review completed : 1st 2022.06.29. / Accepted : 2022.07.06.
DOI : 10.15701/kcgs.2022.28.3.113
ISSN : 1975-7883(Print)/2383-529X(Online)

1. 서론

주어진 가상의 3차원 장면에 대한 실시간으로 사실적인 렌더링 이미지를 생성하는 것은 컴퓨터 그래픽스 분야에서 가장 중요한 연구 주제 중 하나이다. 더욱이, 가상 현실(virtual reality, VR), 증강 현실(argmented reality, AR) 그리고 혼합현실(mixed reality, MR) 사용과 연구[2, 3, 4, 5]가 보편화가 되는 이 시점에 사용자의 만족감 높은 가상환경 체험을 위해서 실시간으로 그림자, 반사, 굴절을 포함한 고화질의 사실적인 렌더링 이미지를 생성하는 것은 더욱 중요성이 부각되고 있다. 하지만 그 가치에도 불구하고 넓은 시야를 가진 고품질 헤드 마운티드 디스플레이(HMD)에서 사용자에게 사실적인 렌더링을 통한 생생한 몰입감과 멀미를 방지하기 위한 HMD의 디스플레이 갱신 속도를 충족하기에 충분히 높은 렌더링 속도를 보장하는 것은 어려운 주제이다.

래스터화 기반 렌더링 방법과 달리 주어진 장면 내에서 빛을 효과적으로 시뮬레이션할 수 있는 광선 추적 기반 렌더링 방법은 다양한 고급 렌더링 효과를 물리적으로 올바른 방식으로 계산 가능하기 때문에 사실적인 렌더링 이미지 생성이 가능하다. 그러나 최근 그래픽카드에 탑재되고 있는 광선 추적 전용 하드웨어(e.g., NVIDIA RT Core[6])를 이용하더라도 광선 추적 기반 렌더링 방법은 매우 비싼 비용을 갖는 것으로 간주된다.

한편, HMD 화면에 시각적으로 성가신 아티팩트를 줄이고, 고품질 이미지를 표시하기 위해서는 높은 샘플링 속도로 픽셀당 충분한 수의 광선 샘플링을 수행해야 한다. 제한적인 컴퓨터 자원 안에서 고화질의 렌더링 이미지를 실시간으로 생성하기 위하여, 예전부터 컴퓨터 그래픽스에서 지각 기반 렌더링의 개념이 꾸준히 연구되어왔다. 이것은 사람의 시각 시스템의 특성과 한계를 기반으로 한다. 사람의 시각 시스템은 망막의 중심으로부터의 각거리인 시각적 이심률(eccentricity)이 작은 중심시는 선명하고 세밀한 시야를 제공하지만, 주변시는 시각적 이심률이 커짐에 따라 훨씬 넓은 부분을 보여주지만 흐릿하게 보인다. 이러한 특성을 이용하는 포비티드 렌더링(foveated rendering) 방법은 지각적으로 최적화된 렌더링을 위해 계산 자원을 이미지 픽셀에 적응적으로 할당하며, 이는 응시하고 있는 중심을 주변보다 더 세밀하고 품질 있게 렌더링 되도록 한다.

본 논문에서는 Kim 등[1]에 의한 선택적 포비티드 광선 추적법에 광선 추적 하드웨어 사용과 시간 제약 렌더링 방법의 결합을 통하여 HMD 시스템에서 실시간 광선 추적에 적합하고, 사람의 시각 시스템에 매우 효율적인 픽셀 샘플링 방법을 제안한다. 특히, 고전적인 Whitted-스타일 광선 추적을 적용함으로써 반사 및 굴절과 같은 조명 효과를 생성할 수 있으며, 계산 자원을 분석하여 시야 전반에 걸쳐 픽셀 샘플링 밀도를 자동으로 제어한다.

2. 관련연구

개발된 최초의 포비티드 렌더링 방법 중 하나는 볼륨 렌더링[7]을 위한 것이었다. 이 렌더링 시스템은 다중 해상도 기술을 사용하여 해상도와 샘플링 속도를 변경하였다. 또 다른 방법들은 시각적 이심률을 기준으로 해상도에 단계를 두었다[8, 9, 10]. Guenter 등[11]은 사용자의 시선을 추적하고, 래스터화 기반의 렌더링을 사용하여 세 가지 다른 샘플링 속도의 이미지 레이어를 렌더링하고, 매끄럽게 합성하여 최종 이미지를 생성하였다. 이를 통하여 HMD 화면 중앙은 픽셀이 선명하게 렌더링 되었고, 외부 영역은 보다 거칠게 표현되었다. Stengel 등[12]은 디퍼드 렌더링(deferred rendering) 방법과 인간 시각 시스템의 여러 측면인 시력, 눈 움직임, 대비 및 밝기에 대한 적응을 통합하는 샘플링 체계 사용하여 포비티드 렌더링을 제안하였다. X. Meng 등[13]은 고전적인 log-polar mapping을 적용한 다항식 커널 함수를 사용하여 시각적 품질과 시간 비용 사이의 균형을 제어할 수 있는 포비티드 렌더링을 제안하였다. Okan 등[14]은 휘도 대비 인식(luminance-contrast-aware) 포비티드 렌더링 방법을 제안하였다. 이 논문에서는 이미지의 지역적 휘도 대비를 분석하고, 이를 시각적 이심률과 결합할 수 있다면 포비티드 렌더링의 계산 절감 효과가 크게 향상될 수 있다는 것을 입증하였다. 앞서 소개한 래스터화 기반 포비티드 렌더링 방법은 고정된 해상도로만 래스터화가 허용되기 때문에 샘플링 수와 속도를 적응적으로 변경하기 어려웠다.

그러나, 광선 추적 기반 방법은 서로 다른 영역에서 서로 다른 샘플링 수와 속도를 생성하는 데 적합하다. Murphy와 Duchowski[15]는 광선 추적을 기반으로 한 하이브리드 이미지/모델 기반 비등방성 시선 추적 렌더링 방법을 제안했다. 이 논문에서는 물체의 실루엣과 대비가 높은 영역 근처에서 샘플링 속도를 증가시켰다. Fujita 등[16]은 계산된 k-최근접 이웃 탐색 알고리즘을 사용하여 넓은 지역에 대하여 밀도가 낮게 샘플링하여 이미지를 재구성하는 샘플링 패턴을 계산하였다.

하지만, 위의 방법들은 공간적 앨리어싱(spatial aliasing) 또는 시간적 앨리어싱(temporary aliasing)의 문제를 해결하지 못하였다. 포비티드 렌더링에서 시각적 이심률에 따라 샘플링 수를 줄이면 주변시에서 공간 및 시간적 앨리어싱 문제가 발생한다. 공간적 앨리어싱은 공간 축을 따라 데이터가 충분히 샘플링되지 않은 반면, 시간 앨리어싱은 시간 공간의 데이터 샘플링이 충분하지 않음을 의미한다. Jin 등[17]은 이미지 공간 및 물체의 속성을 검사하여 픽셀당 9~16개의 샘플링을 효과적으로 수행하여 앨리어싱을 해결

하였다. Vaidyanathan 등[18]은 래스터화 기반 파이프라인에서 일정한 렌더링 속도를 유지하면서 셰이딩(shading) 샘플의 수를 변경할 수 있는 아키텍처인 coarse pixel shading(CPS)을 제안하였다. Weier 등[19]은 시선 추적 포비티드 렌더링 방법과 이전 프레임을 사용하는 투영 렌더링 방법을 결합하여 프레임당 샘플링 수를 대폭 줄였다. Patney 등[20]은 주변시의 부족한 샘플링에서 발생하는 앨리어싱을 해결하기 위해 색상의 대비 값을 사용하여 다중 해상도와 시선 이동을 고려한 안티 앨리어싱 알고리즘을 제안하였다. Kim 등[1]은 포비티드 렌더링 영역 및 샘플링 속도를 사용자가 원하는 대로 조정할 수 있도록 하여 픽셀당 최대 36개의 샘플링을 효과적으로 수행하여 공간적 앨리어싱을 해결했을 뿐만 아니라, 시간적 앨리어싱을 제거하고 렌더링 시간을 줄이기 위해 이전 프레임을 사용하는 재투영 방법을 사용하였다.

3. 선택적 포비티드 광선 추적법

본 논문의 방법은 Kim 등[1]이 제안한 선택적 포비티드 광선 추적법에 광선 추적 하드웨어 사용과 시간 제약 렌더링 방법을 결합하였다. Kim 등이 제안한 방법[1]은 Jin 등이 제안한 방법[17]을 확장하였다. 따라서 본 논문의 방법을 설명하기 전 3장에서는 Kim 등이 제안한 방법[1]과 Jin 등이 제안한 방법[17]에 대하여 간략하게 설명한다.

3.1 시각적 시스템을 반영한 이미지 영역

포비티드 렌더링은 시선의 중심을 기준으로 시각적 이심률이 작은 중심시는 최대 화질을 유지하고, 주변시로 갈수록 유연하게 저화질로 표현함으로써 그래픽 연산 부하를 최소화하는 렌더링 방법이다. 따라서 포비티드 렌더링에서 시각 시스템의 특성을 근거로 렌더링 노력의 차이를 주는 렌더링 이미지 영역을 구분하는 것은 중요하다. Kim 등[1]은 이미지의 영역을 다음과 같이 구분하였다.

Fovea : $0 \sim e_f(0 \sim 3.3^\circ)$
Parafovea : $e_f \sim e_p(3.3^\circ \sim 5.5^\circ)$
Perifovea : $e_p \sim e_m(5.5^\circ \sim 12.1^\circ)$
Near-periphery : $e_m \sim e_{np}(12.1^\circ \sim 30.0^\circ)$

그림 1은 Kim 등[1]이 구분한 영역을 도식화 한 것이다. 가장 안쪽부터 순차적으로 f(Fovea), p(Parafovea), m(Perifovea), np(Near-periphery)를 의미한다.

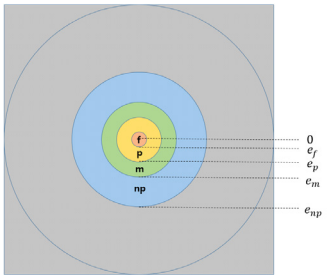


Figure 1. Partitioning of image space by visual eccentricity

3.2 알고리즘

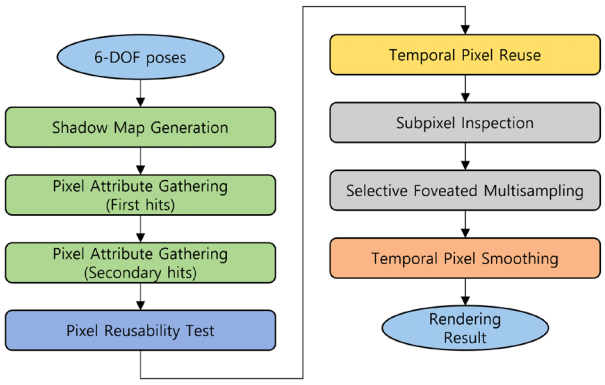


Figure 2. Selective foveated ray tracing pipeline [1]

Kim 등이 제안한 방법[1]의 핵심 아이디어는 픽셀의 중앙에서 샘플링을 수행하여 픽셀의 이미지 및 물체 속성을 모두 수집하고, 주변 픽셀의 값들과 비교하여 선택적으로 설정된 기준과 시각적 이심률에 따른 함수에 따라 각 픽셀 영역에서 최대 36개의 광선 샘플링을 수행하는 것이다. 그림 2는 감지된 HMD 6-DOF 포즈(pose)를 사용하여 스테레오 이미지가 합성되는 선택적 포비티드 광선 추적 파이프라인을 보여준다. 6-DOF의 포즈가 입력으로 들어오면 각 눈에 대하여 렌더링이 시작된다.

Shadow Map Generation 단계에서는 e_{np}영역 밖의 그림자에 한해서 OpenGL을 활용한 그림자 렌더링을 수행한다.

Pixel Attribute Gathering(First hits) 단계에서는 이미지 각 픽셀의 중앙에 대하여 1차 광선에 대한 물체와의 교차지점의 속성들을 OpenGL을 사용한 디퍼드 렌더링(deferred rendering)으로 수집한다. 수집하는 속성들은 표 1과 같이 두 개의 속성 그룹으로 나뉜다. 첫 번째 그룹은 이미지 공간상에서의 속성으로 Shadow Map Generation, 2단계의 Pixel Attribute Gathering을 통하여 계산된 색상을 의미한다. 이 속성은 픽셀의 서브(sub) 픽셀에서 더 많은 광선 샘플링

이 필요한지를 최종적으로 결정하는 색상 차이 검사에서 참조된다. 두 번째 속성 그룹은 물체 공간상의 기하학적 속성이라고 하는데, 이는 광선에 맞은 3D 공간의 표면 위치에서 수집되기 때문이다. 앞서 언급한 대로 해당 단계에서는 OpenGL을 활용하여 표 1의 Primary ray hit point에 해당하는 4가지 다른 유형의 기하학적 속성이 수집되며, 각 특성은 광선 추적에 의해 자주 생성되는 시각적 아티팩트와 깊은 관련이 있다.

Table 1. List of pixel attributes [17]

	Sampling location	Attribute	Threshold
Image space	Pixel center	Color reference	τ_{col}
Object space	Primary ray hit point	Object ID	τ_{poid}
		Surface normal	τ_{psn}
		Shadow count	τ_{psc}
		Texture existence	τ_{pte}
	Secondary ray hit point	Object ID	τ_{soid}
		Surface normal	τ_{ssn}
		Shadow count	τ_{ssc}
		Texture existence	τ_{ste}

Pixel Attribute Gathering(Secondary hits) 단계에서는 표 1의 Secondary ray hit point에 해당하는 기하학적 속성을 수집한다. 앞선 단계를 통하여 첫 교차 지점에 대한 정보를 알고 있으므로, 반사/굴절 효과를 위한 2차 광선을 생성하여 교차하는 표면 지점의 기하학적 속성을 수집한다.

Pixel Reusability Test 단계는 비용이 많이 드는 광선 추적을 통한 색상 계산 대신, 직전 픽셀 색상을 현재 프레임에 재사용하기 위한 단계이다. 수집된 현재 프레임의 픽셀 속성과 직전 프레임의 픽셀 속성을 비교하여 픽셀 재사용 가능성 여부를 검사한다.

Temporal Pixel Reuse 단계에서는 앞선 단계에서 픽셀 재사용 가능함으로 판단된 픽셀에 한하여, 픽셀 색상 계산에 비용이 많이 드는 광선 추적 대신 현재 프레임의 각 픽셀 $p_{cur} = (i, j)$의 색상을 직전 프레임으로 역 투영된 픽셀 $p_{reuse} = (i_{reuse}, j_{reuse})$의 색상으로 단순히 복사한다.

Subpixel Inspection 단계에서는 픽셀 재사용 불가능으로 판단된 픽셀에 한하여 4개의 동일한 크기의 서브 픽셀로 분할하고, 각 서브 픽셀은 그림 3과 같이 서브 픽셀의 위치를 기준으로 3개의 인접 픽셀의 속성과 비교하여 추가적인 광선 추적을 통한 샘플링이 필요로 하는지 판단한다.

Figure 3. Example of subpixel testing [17]

서브 픽셀 검사는 2가지 단계로 구성된다. 1) 주어진 서브 픽셀에 대하여 주변의 세 이웃 픽셀과 속성들을 비교하여 불일치하면 2)를 수행한다. 2) 1)단계에서 계산된 이웃 픽셀들 간의 R, G, B 대비(contrast) 값들 중 슈퍼 샘플링(super sampling)의 기준이 되는 대비 임계값보다 어느 한 값이라도 크다면 다음 단계에서 광선을 추가적으로 생성한다.

$$(\tau_{new} \cdot 1.36, \tau_{new} \cdot 1.02, \tau_{new} \cdot 2.04),$$
$$\tau_{new} = \tau_{fov}(e) \cdot \tau_{sel}$$

위 식은 슈퍼 샘플링의 기준이 되는 대비 임계값 벡터이다. (1.36, 1.02, 2.04)는 Jin 등[17]이 Mitchell이 제안한 방법[21]을 변형한 임계값 벡터이며, τ_{sel}은 표 1의 임계값 중 테스트에 사용되는 임계값, $\tau_{fov}(e)$은 Kim 등이 제안한 방법[1]인 사람의 시각 시스템의 특성을 이용한 시각적 이심률에 따른 함수 $\tau_{fov}(e)$이다.

Selective Foveated Multisampling 단계에서는 검사 결과에 따라, 그림 4와 같이 현재 서브 픽셀에 대해 미리 샘플링된 픽셀 색상을 사용하거나, 각 영역에 대하여 적절한 수의 추가적인 광선을 생성하여 추가 샘플링을 한다. 모든 추가 광선 추적 계산이 완료된 후, 획득한 색상들을 적절한 가중치에 따라 가중 평균한다.

Figure 4. Example of adaptive sampling [17]

Temporal Pixel Smoothing 앞선 단계에서 효율적으로 수행된 많은 샘플링 수를 통하여 공간적 엘리어싱은 개선되었지만, 연속적인 프레임 사이의 시간적 앨리어싱은 여전히 존재한다. 해당 단계에서는 현재 픽셀을 이전 n 개의 프레임으로 역투영하여, 해당하는 픽셀의 색상과 적절히 혼합함으로써 시간적 앨리어싱을 감소시킨다.

4. 본 논문 제안 알고리즘

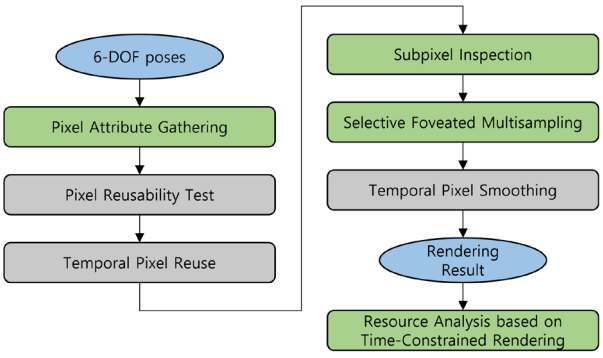


Figure 5. Our adaptive foveated ray tracing based on Time-Constrained rendering pipeline

그림 5는 본 논문의 방법인 시간 제약 렌더링을 이용한 적응적 포비티드 광선 추적법의 파이프라인이다. Kim 등이 제안한 방법[1]과 비교하여 동일한 알고리즘을 사용하는 단계는 회색으로 표시하였으며, 개선 및 추가된 단계는 녹색으로 표시하였다. 본 장에서는 개선 및 추가된 단계를 중심으로 설명하도록 한다.

4.1 Pixel Attribute Gathering

Kim 등[1]은 픽셀의 속성을 얻기 위하여 그림 2의 Shadow Map Generation, 2단계의 Pixel Attribute Gathering을 수행하였다. 그러나 본 논문에서는 OpenGL의 레스터화 기반 렌더링 방식과 광선 추적 기반 렌더링 방식을 함께 사용하는 하이브리드 광선 추적 방식을 취하는 것이 아닌, 최신 NVIDIA GPU에서 광선 추적 하드웨어(RT Core)를 사용 가능케하는 OptiX[22]와 CUDA[23]를 사용하여 Pixel Attribute Gathering 단계를 가속하였다. 반사/굴절/그림자 효과를 표현할 수 있는 Whitted-스타일의 광선 추적을 사용하여 픽셀당 하나의 광선 샘플로 표 1의 속성들을 계산하였다. 또한 색상 속성은 주변 픽셀 영역에 대한 색상으로도 사용하였다.

4.2 Subpixel Inspection

그림 5의 Pixel Reusability Test 단계에서 픽셀 재사용이 불가능하다고 판단된 픽셀은 시간 제약 렌더링을 이용한 적응적 포비티드 광선 추적법이 수행되어야 한다. 이를 위하여 Kim 등의 방법[1]처럼 각 픽셀의 4개 서브 픽셀 영역에 대해 추가 광선 생성이 필요한지에 대한 검사를 수행한다. 만약, 추가 광선 생성이 필요하다고 판단된다면, 마스크 맵(mask map)에 추가 광선 생성이 필요한 서브 픽셀의 위치를 마스크를 하고 이를 다음 단계에서 사용한다. 해당 단계와 4.3 단계의 OptiX를 활용한 효율적인 구현에 대한 내용은 5장에서 자세히 서술한다.

4.3 Selective Foveated Multisampling

서브 픽셀 검사의 결과로 추가 광선 생성이 필요하다고 판단되는 각 서브 픽셀에 대해, 시각 시스템에 의한 이미지 영역에 따라 샘플링 광선이 생성된다. Kim 등의 방법[1]은 이미지 영역의 서브 픽셀당 생성하는 샘플링 광선의 수를 사용자가 설정하여 샘플링 밀도를 제어할 수 있도록 하였다. 이 논문에서는 이미지의 영역에 대한 서브 픽셀당 생성하는 샘플 광선의 수를 (9, 4, 2, 1)로 설정하여 실험을 진행하였으며, 이는 픽셀당 샘플링 수가 fovea, parafovea, perifovea, near-periphery 및 주변부에서 최대 36, 16, 8, 4임을 의미한다. 그러나 해당 방법은 각 영역에서의 서브 픽셀당 생성하는 샘플 광선의 수가 고정되어 있으므로 영역 간의 생성된 최대 광선 수가 급격하게 변화한다. 예를 들어 fovea 영역은 서브 픽셀당 9개의 광선을 생성함으로 최대 36개의 광선이 생성되지만, 그다음 영역인 parafovea 영역에서는 서브 픽셀당 4개의 광선을 생성함으로 최대 16개의 광선이 생성된다. 이는 fovea와 parafovea의 경계 기준이 되는 3.3° 에서 광선의 생성 수가 급격하게 감소됨으로 자연스럽지 않다. 본 논문에서는 해당 문제를 해결하고자 각 영역의 서브 픽셀당 생성하는 샘플 광선의 수가 선형적으로 감소되도록 개선하였다. 만약 사용자가 서브 픽셀당 생성하는 샘플 광선의 수를 (9, 4, 2, 1)로 설정하였다고 한다면, fovea 영역(0 ~ 3.3°)에서는 Kim 등의 방법[1]과 동일하게 최대 36개의 광선을 생성한다. 그러나 parafovea(3.3° ~ 5.5°)는 범위 내에서 서브 픽셀당 생성하는 광선의 수를 9 ~ 4(픽셀당 36 ~ 16)로 선형적으로 감소되도록 하였으며, 이와 같은 방법으로 perifovea는 범위 내에서 4 ~ 2(픽셀당 16 ~ 8_, near-periphery는 2 ~ 1(픽셀당 8 ~ 4)로 선형적으로 감소되도록 하였다.

추가 광선 생성이 필요한 모든 하위 픽셀에 대해 계산이 완료되면, 계산된 광선 색상을 적절한 가중치로 혼합하여 최종 픽셀 색상을 계산한다.

4.4 Resource Analysis based on Time-Constrained Rendering

HMD에는 디스플레이 갱신 속도가 정해져있으며(실험에 사용한 Oculus Rift S의 경우 80Hz), 렌더러에서 이보다 높은 갱신 속도로 렌더링이 가능함에도 불구하고, 때로는 V-Sync (Vertical-Synchronization)로 인하여 한계가 정해진

다. 따라서 본 논문에서는 자원의 여유가 남는 경우 추가적인 광선을 생성하여 슈퍼 샘플링을 보다 효과적으로 수행할 수 있도록 하였다.

본 논문의 렌더러는 매 프레임 렌더링마다 그림 5의 6-DOF 포즈 입력부터 스테레오 이미지가 출력될 때까지의 시간을 계산한다. 만약 사용자가 정해놓은 임계값보다 시간이 적게 소요되었다면, 사용자가 설정한 서브 픽셀당 생성하는 샘플 광선의 수를 증가시킨다. 즉, fovea, parafovea, perifovea, near-periphery 영역 중 fovea를 제외한 parafovea, perifovea, near-periphery의 서브 픽셀당 생성하는 샘플 광선의 수를 증가시키며, 각 영역은 상위 영역의 임계값을 넘지 않도록 한다. 또한 점진적으로 샘플 광선의 수를 변화시키기 위하여 한 번에 한 영역의 샘플 광선 임계값이 변화하도록 하였다. 예를 들면 현재 프레임에서 렌더링 결과 시간이 정해진 임계값보다 적게 소요되어, 계산을 위한 자원이 남는다고 판단되다면, parafovea의 임계값을 1 증가시킨다. 그다음 프레임에서도 자원의 여유가 있다고 판단된다면, perifovea의 임계값을 1증가 시킨다. 반대로 자원의 여유가 없다고 판단된다면 매 렌더링 프레임마다 1개의 영역에 대하여 임계값을 1씩 감소시키며, 하한선은 사용자가 최초에 설정한 임계값이다(본 논문에서는 9, 4, 2, 1로 설정). 이를 통하여 GPU의 계산 자원을 최대한 활용한 시간 제약 렌더링을 구현하였다.

5. OptiX를 활용한 효율적인 구현

본 논문의 방법을 구현하기 위하여 사용한 NVIDIA OptiX는 CUDA 기반 응용 프로그램에 의해 호출되는 CUDA 기반 API이다[22]. 다음의 몇 가지 사항을 고려함으로써 성능 및 기능을 향상시켰다.

첫째, NVIDIA 그래픽 카드의 GPU 코어(core)인 광선 추적에 특화된 RT Core를 활용하는 OptiX를 CUDA와 동시에 사용하여 시스템을 구성하였으며, 그림 5의 각 단계들을 위한 GPU 함수를 다중-스레드(multi-thread), 비동기(asynchronous)가 가능하도록 구현하여 성능 최적화를 진행하였다. 또한 다중 디바이스(multi-device)를 지원하기 때문에, 양안을 렌더링 해야하는 HMD 환경에서 2개의 일반적인 GPU가 각각 한쪽 눈을 렌더링 담당하도록 하여 성능 향상을 이루었다.

둘째, 효율성과 일관성을 위하여 OptiX는 CUDA와 달리 동일한 블록(block) 내의 스레드들 간의 메모리 공유를 위한 셰어드 메모리(shared memory), 스레드들 간의 싱크를 맞추는 API를 제공하지 않는다[22]. 그림 5의 Selective Foveated Multisampling 단계를 CUDA를 이용하여 구현한다면 각 픽셀의 4개 서브 픽셀에서 추가적으로 생성하는 샘플 광선의 수만큼 스레드를 할당하여 색상 계산을 한 후, 글로벌 메모리(Global memory) 접근 최소화를 위하여 블록 내에서 셰어드 메모리를 사용하여 색상을 누적하여 혼합한다. 하지만, OptiX를 활용한 구현에서는 전체 렌더링 이미지 해상도만큼 스레드를 할당하고, 추가적으로 샘플 광선을 생성해야 하는지에 대한 정보가 담긴 마스크 맵을 확인하여, 해당되는 픽셀의 스레드에서 추가 생성 광선의 수만큼 반복문을 돌며 색상을 계산하도록 구현하였다. 이는 하나의 스레드 안에 많은 반복문이 존재하여 비효율적으로 보일 수 있지만, 하드웨어의 가속을 받음으로 오히려 CUDA를 활용한 광선 추적보다 빠른 렌더링 속도를 보였다.

셋째, Kim 등의 방법[1]은 포비티드 광선 추적을 CUDA 기반으로 구현하였으며, kd-tree[24]를 공간 가속 구조로 채택하였기 때문에 정적 기하학 장면으로 사용 장면을 제한하였다. 그러나 본 논문에서 사용하는 OptiX는 동적인 물체에 빠른 공간 자료구조 업데이트를 지원하는 BVH(Bounding Volume Hierarchy)[25] 공간 가속 구조를 사용하기 때문에 동적 장면에서도 빠른 렌더링 결과를 확인할 수 있었다.

6. 실험 결과

실험 장면은 각각 279K, 1,021K 및 6,704K 삼각형으로 구성된 3개의 3D 장면인 Crytek Sponza(CS), Interior(IN) 및 Rungholt(RU)가 사용되었다. 실험 장비는 2개의 NVIDIA GeForce RTX 3090 GPU를 탑재한 PC, HMD 기기는 한쪽 눈당 1280×1440 해상도를 갖는 Oculus Rift S를 사용하였다.

본 논문에서 제안한 렌더링 방법의 효과를 평가하기 위해 4가지의 광선 추적법을 서로 비교하였으며 다음과 같다.

A 픽셀당 36개의 샘플링을 수행한 광선 추적 렌더링 방법(Ground-Truth로 사용)
B 픽셀당 1개의 샘플링을 수행한 광선 추적 렌더링 방법
C 선택적 포비티드 광선 추적 렌더링 방법[1]
D 본 논문에서 제안한 시간 제약 렌더링을 이용한 적응적 포비티드 광선 추적 렌더링 방법

모두 Whitted-스타일의 광선 추적법을 사용하였으며, 렌더링 해상도는 양안의 해상도를 포함하는 2560x1440, 광원은 2개, 최대 바운스는 3으로 하였다.

그림 6과 7는 연속한 500프레임에 대하여 카메라를 이동시키며 광선 추적을 할 때, 본 논문에서 제시한 시간 제약 렌더링의 효과를 보여주고 있다. 그림 6은 추가 광선의 수에 따라 무지개색으로 도식화한 것으로 색상의 의미는 그림 9와 같다. 첫 번째 행은 본 논문의 방법으로 렌더링을 수행한 결과, 두 번째 행은 서브 픽셀당 생성하는 샘플 광선의

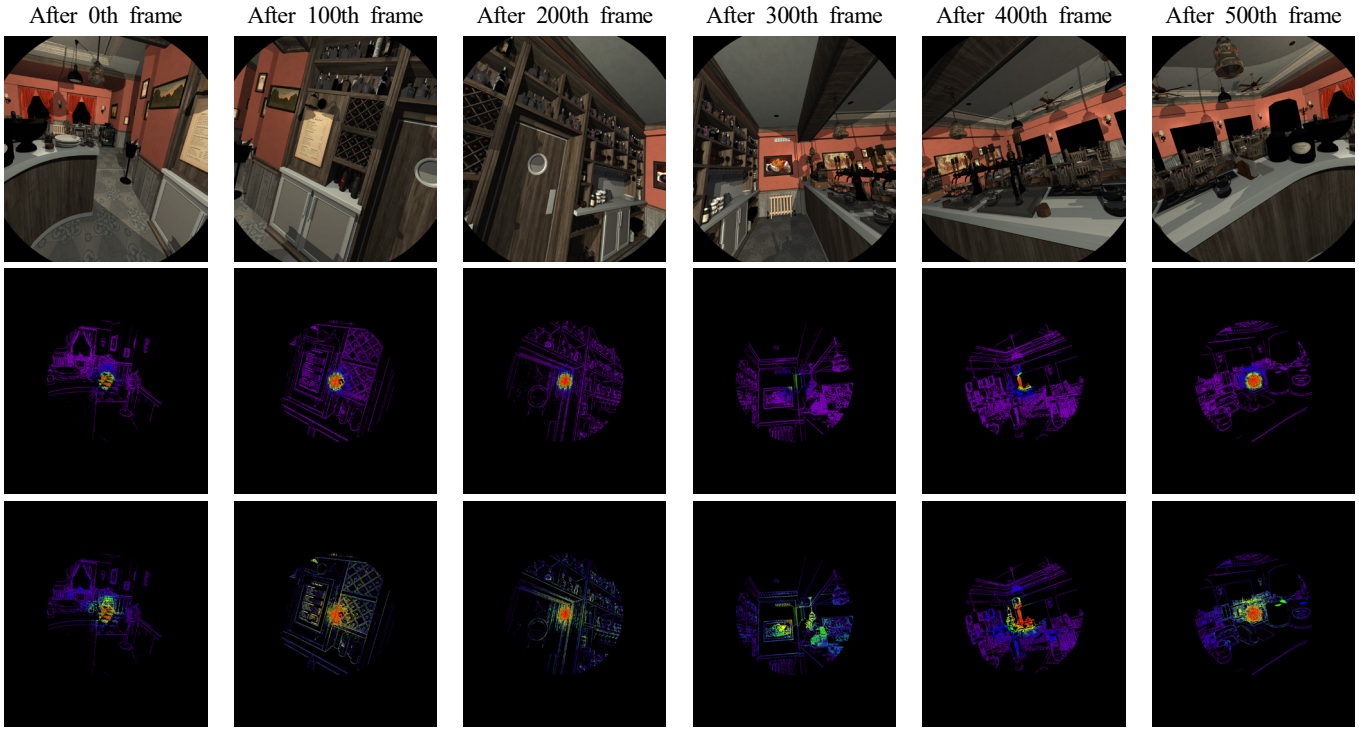


Figure 6. Rendering results of time-constrained ray tracing

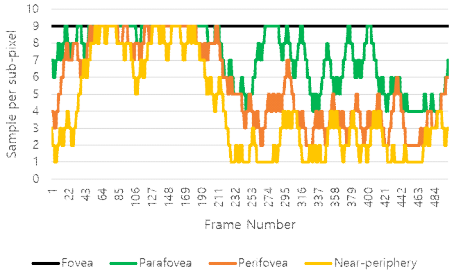


Figure 7. Variation of sampling density according to rendering complexity

수를 사용자가 설정한 대로 고정하여 렌더링을 수행한 결과, 그리고 세 번째 행은 본 논문의 방법을 사용하여 자원의 여유가 있다면 서브 픽셀당 생성하는 샘플 광선의 수를 조절하며 렌더링을 수행한 결과이다. 각 열은 왼쪽에서 오른쪽으로 갈수록 시간이 지나며 카메라가 움직인다. 본 논문의 방법은 자원의 여유에 따라 자동으로 서브 픽셀당 생성되는 광선의 수를 조절하기에 다채로운 색상으로 표현됨을 확인할 수 있다. 그림 7은 각 영역에서의 서브 픽셀당 생성하는 광선의 수를 그래프로 보인 것이다. fovea를 제외한 parafovea, perifovea, near-periphery의 서브 픽셀당 생성하는 샘플 광선의 수가 계산 자원의 여력에 따라 조절이 되며, 각 영역의 광선의 수는 상위 영역의 임계값을 넘지 않는 것을 확인할 수 있다.

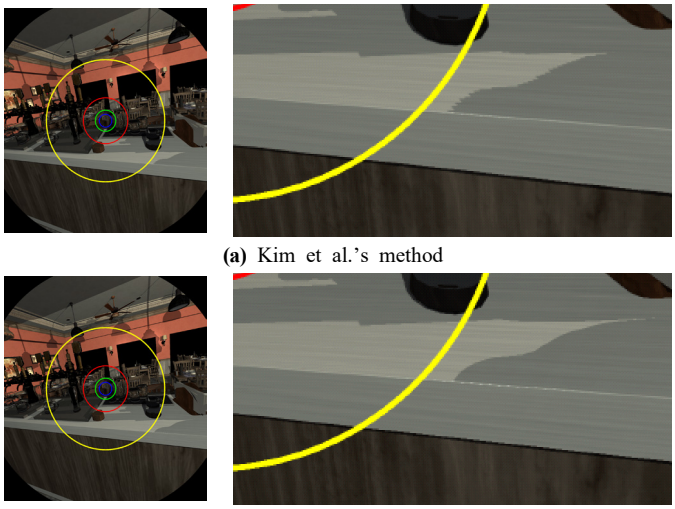
Table 2. Quantitative comparison of four ray-tracing methods

		Time (ms)	PSNR (dB) [fovea/parafovea/perifovea/near-periphery]
CS	A	182.80	-
	B	9.60	32.18/30.31/29.55/33.0
	C	15.36	44.15/40.77/36.06/35.64
	D	13.12	**44.42**/41.98/37.15/37.33
IN	A	159.25	-
	B	8.85	23.55/24.37/27.15/31.52
	C	17.49	36.66/35.70/35.27/36.54
	D	13.4	**36.67**/36.61/36.00/36.80
RU	A	282.61	-
	B	12.60	23.52/23.49/24.29/26.81
	C	23.33	35.24/33.82/31.54/30.87
	D	17.03	**35.25**/34.77/32.07/30.42

표 2는 3개의 실험 장면에 대한 각 렌더링 방법의 소요 시간과 Ground-Truth와의 PSNR 비교이다. 본 논문의 방법(D)은 4가지 영역으로 이미지를 구분하여 샘플링 수를 다르게 하기 때문에 PSNR 측정을 4가지 영역 별로 측정하였다. PSNR 란의 왼쪽에서 오른쪽으로 순서대로 fovea, parafovea, perifovea, near-periphery 영역에서의 PSNR이다. 표에서 확인할 수 있듯이 본 논문의 방법에 해당하는 D의 렌더링 시간이 제일 적음을 확인할 수 있으며, 각 영역별로 PSNR도 가장 높음으로 매우 효율적인 렌더링을 수행하고 있음을 알 수 있다.

그림 10은 각 렌더링 방법의 결과 이미지 비교이다. 행으로는 순서대로 CS, IN 및 RU 장면이며, 첫 번째 열은 본 논문의 방법으로 렌더링 한 결과, 두 번째 열은 A 와 B 렌더링 결과 차이 이미지, 세 번째 열은 A 와 C 렌더링 결과 차이 이미지, 네 번째 열은 A 와 D 렌더링 결과 차이 이미지를 나타낸다. 그림에서 확인할 수 있듯이 본 방법은 중심시에 해당하는 부분은 Ground-Truth와 차이가 적고, 주변시로 갈수록 점진적으로 차이가 남을 확인할 수 있다. 그러나 이러한 차이는 HMD 착용 시 시각적 차이를 구분하기 힘들었다.

그림 11은 CS 장면에 30프레임의 시간 동안 동적으로 움직이는 78K의 삼각형으로 구성된 Ben 모델을 추가하여 렌더링 한 결과이다. 공간 가속 구조 업데이트와 렌더링 시간을 모두 포함하여 한 프레임 당 평균 16.1ms의 시간이 소요되었으며, 빠른 렌더링 속도를 유지하면서 동적 장면을 렌더링 할 수 있었다.



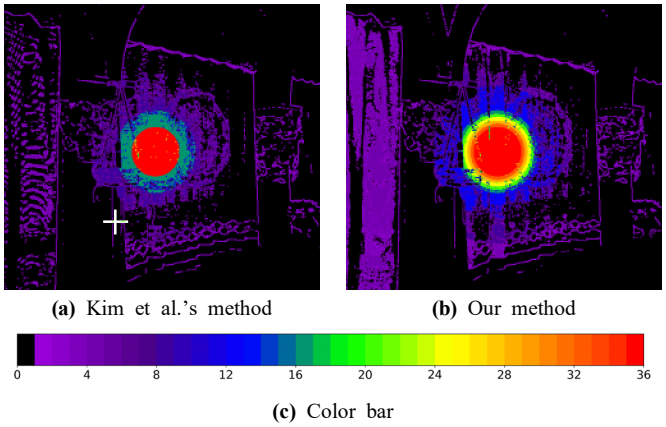
(a) Kim et al.'s method

(b) Our method

Figure 8. Enhanced shadow effect result

그림 8은 4.1 Pixel Attribute Gathering 방법에 의한 그림자 효과 개선 결과이다. Kim 등의 방법[1]은 OpenGL을 활용한 그림자 렌더링을 사용하였으므로, 그림자 매핑(shadow mapping)의 해상도 문제로 인하여, 때로는 그림 8의 (a)와 같이 부정확한 렌더링 결과를 보였으나, OptiX를 활용한 전체 이미지에 대한 광선 추적을 수행한 본 논문의 방법에서는 그림 8의 (b)와 같이 개선된 결과를 보였다.

그림 9은 4.3 Selective Foveated Multisampling 방법의 결과를 도식화한 것으로, 추가 광선이 생성되는 이미지 영역을 광선의 수가 많으면 빨간색, 적으면 보라색으로 무지개색에 따라 나타내었다(그림 9의 (c)는 광선의 수에 따른 색상을 표시한 색상 막대). Kim 등의 결과[1](그림 9의 (a))와 달리 본 논문의 방법(그림 9의 (b))은 다양한 색상의 영역이



(a) Kim et al.'s method **(b)** Our method

(c) Color bar

Figure 9. Enhanced selective foveated multisampling result

보임을 확인할 수 있으며, 이는 각 영역 간의 광선의 수에 대한 급격한 변화가 발생되지 않도록 하여 보다 자연스러운 렌더링 결과를 얻을 수 있음을 보여준다.

7. 결론

본 논문에서는 HMD에서 고화질의 사실적인 렌더링 이미지를 실시간으로 생성하기 위한, 시간 제약 렌더링을 이용한 적응적 포비티드 광선 추적법을 제안하였다. 본 논문의 방법은 OptiX와 CUDA를 활용하여 광학적으로 정확한 그림자, 반사 및 굴절 효과를 렌더링 할 수 있었으며, 추가적으로 광선 생성을 필요로 하는 픽셀을 검사하여 사용자가 정의 한 서브 픽셀당 광선 샘플의 수를 토대로 효과적인 샘플링을 수행하였다. 또한, 갖고 있는 GPU 자원을 최대한 활용하기 위하여 주어진 시간 안에 계산 자원의 여력이 있는지 확인하고, 자동으로 화질을 조절함으로써 사람이 느끼기에 충분한 고화질의 렌더링 결과 이미지를 얻을 수 있었다. 끝으로 정적 장면뿐만 아니라 동적 장면에서 본 논문의 방법이 효과적으로 동작함을 확인할 수 있었다.

비록 매 프레임의 렌더링 시간을 검사 후 자동적으로 서브 픽셀당 샘플의 수를 선형적으로 변경하는 것은 GPU 자원을 최대한 효율적으로 사용할 수 있도록 하나, 사람의 시각 시스템의 정보를 활용하는 것은 아니다. 만약 사람의 시각 시스템의 정보가 반영되어 임계값이 조절된다면 보다 자연스러운 포비티드 렌더링이 될 것이라고 예상된다.

감사의 글

이 성과는 정부 (과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (No. NRF-2020R1A2C2011709).

References

[1] Y. Kim, Y. Ko, and I. Ihm, “Selective Foveated Ray Tracing for Head-Mounted Displays,” *2021 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pp. 413-421, 2021.

[2] 서웅, 권상모, 임인성, “가상 환경에서의 손동작을 사용한 물체 조작에 대한 시각적 피드백 시스템,” *컴퓨터그래픽스학회논문지*, Vol. 26, No. 3, pp. 9-19, 2020.

[3] 김종용, 박동근, 이필연, 조준영, 윤승현, 박상훈, “실감형 가상현실 실전훈련 콘텐츠를 위한 관리 평가 시스템 개발 사례 연구,” *컴퓨터그래픽스학회논문지*, Vol. 26, No. 3, pp. 111-121, 2020.

[4] 이재현, 박경주, “대화형 가상 현실에서 아바타의 립싱크,” *컴퓨터그래픽스학회논문지*, Vol. 26, No. 4, pp. 9-15, 2020.

[5] 홍승현, 나기리, 조윤식, 김진모, “모바일 가상현실에서의 이동 인터페이스에 관한 연구,” *컴퓨터그래픽스학회논문지*, Vol. 27, No. 3, pp. 55-63, 2021.

[6] NVIDIA, “NVIDIA Ampere GA102 GPU Architecture: Second-Generation RTX,” *Whitepaper*, 2021.

[7] M. Levoy, and R. Whitaker, “Gaze-directed volume rendering,” *Proceedings of the 1990 symposium on interactive 3d graphics*, pp. 217-223, 1990.

[8] B. Watson, N. Walker, L. F. Hodges, and A. Worden, “Managing level of detail through peripheral degradation: Effects on search performance with a head-mounted display,” *ACM Transactions on Computer-Human Interaction (TOCHI)*, Vol. 4, No. 4, pp. 323-346, 1997.

[9] B. Watson, N. Walker, and L. F. Hodges, “Supra-threshold control of peripheral LOD,” *ACM Transactions on Graphics (TOG)*, Vol. 23, No. 3, pp. 750-759, 2004.

[10] A. T. Duchowski, D. Bate, P. Stringfellow, K. Thakur, B. J. Melloy, and A. K. Gramopadhye, “On spatiochromatic visual sensitivity and peripheral color LOD management,” *ACM Transactions on Applied Perception (TAP)*, Vol. 6, No. 2, pp. 1-18, 2009.

[11] B. Guenter, M. Finch, S. Drucker, D. Tan, and J. Snyder, “Foveated 3d graphics,” *ACM Transactions on Graphics (TOG)*, Vol. 31, No. 6, pp. 1-10, 2012.

[12] M. Stengel, S. Grogorick, M. Eisemann, and M. Magnor, “Adaptive image-space sampling for gaze-contingent real-time rendering,” *Computer Graphics Forum*, Vol. 35, No. 4, pp. 129-139, 2016.

[13] X. Meng, R. Du, M. Zwicker, and A. Varshney, “Kernel foveated rendering,” *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, Vol. 1, No. 1, pp. 1-20, 2018.

[14] O. T. Tursun, E. Arabadzhiyska-Koleva, M. Wernikowski, R. Mantiuk, H.-P. Seidel, K. Myszkowski, and P. Didyk, “Luminance-contrast-aware foveated rendering,” *ACM Transactions on Graphics (TOG)*, Vol. 38, No. 4, pp. 1-14, 2019.

[15] H. A. Murphy, A. T. Duchowski, and R. A. Tyrrell, “Hybrid image/model-based gaze-contingent rendering,” *ACM Transactions on Applied Perception (TAP)*, Vol. 5, No. 4, pp. 1-21, 2009.

[16] M. Fujita and T. Harada, “Foveated real-time ray tracing for virtual reality headset,” *Light Transport Entertainment Research*, 2014.

[17] B. Jin, I. Ihm, B. Chang, C. Park, W. Lee, and S. Jung, “Selective and adaptive supersampling for real-time ray tracing,” *Proceedings of the Conference on High Performance Graphics 2009*, pp. 117-125, 2009.

[18] K. Vaidyanathan, M. Salvi, R. Toth, T. Foley, T. Akenine-M¨oller, J. Nilsson, J. Munkberg, J. Hasselgren, M. Sugihara, P. Clarberg, et al, “Coarse pixel shading,” *Proceedings of High Performance Graphics*, pp. 9-18, 2014.

[19] M. Weier, T. Roth, E. Kruijff, A. Hinkenjann, A. P´erard-Gayot, P. Slusallek, and Y. Li, “Foveated real-time ray tracing for head-mounted displays,” *Computer Graphics Forum*, Vol. 35, No. 7, pp. 289-298, 2016.

[20] A. Patney, M. Salvi, J. Kim, A. Kaplanyan, C. Wyman, N. Benty, D. Luebke, and A. Lefohn, “Towards foveated rendering for gazetracked virtual reality,” *ACM Transactions on Graphics (TOG)*, Vol. 35, No. 6, pp. 1-12, 2016.

[21] D. Mitchell, “Generating antialiased images at low sampling densities,” *Proceedings of SIGGRAPH 1987*, pp. 65-72, 1987.

[22] NVIDIA OptiX 7.4 Programming Guide, 2022.

[23] NVIDIA CUDA C++ Programming Guide, 2022.

[24] I. Wald and V. Havran, “On building fast Kd-trees for ray tracing, and on doing that in O(Nlog N),” *Proceedings of the IEEE Symposium on Interactive Ray Tracing*, pp. 61-69, 2006.

[25] I. Wald, “On fast construction of SAH-based bounding volume hierarchies,” *Proceedings of the IEEE Symposium on Interactive Ray Tracing*, pp. 33-40, 2007.

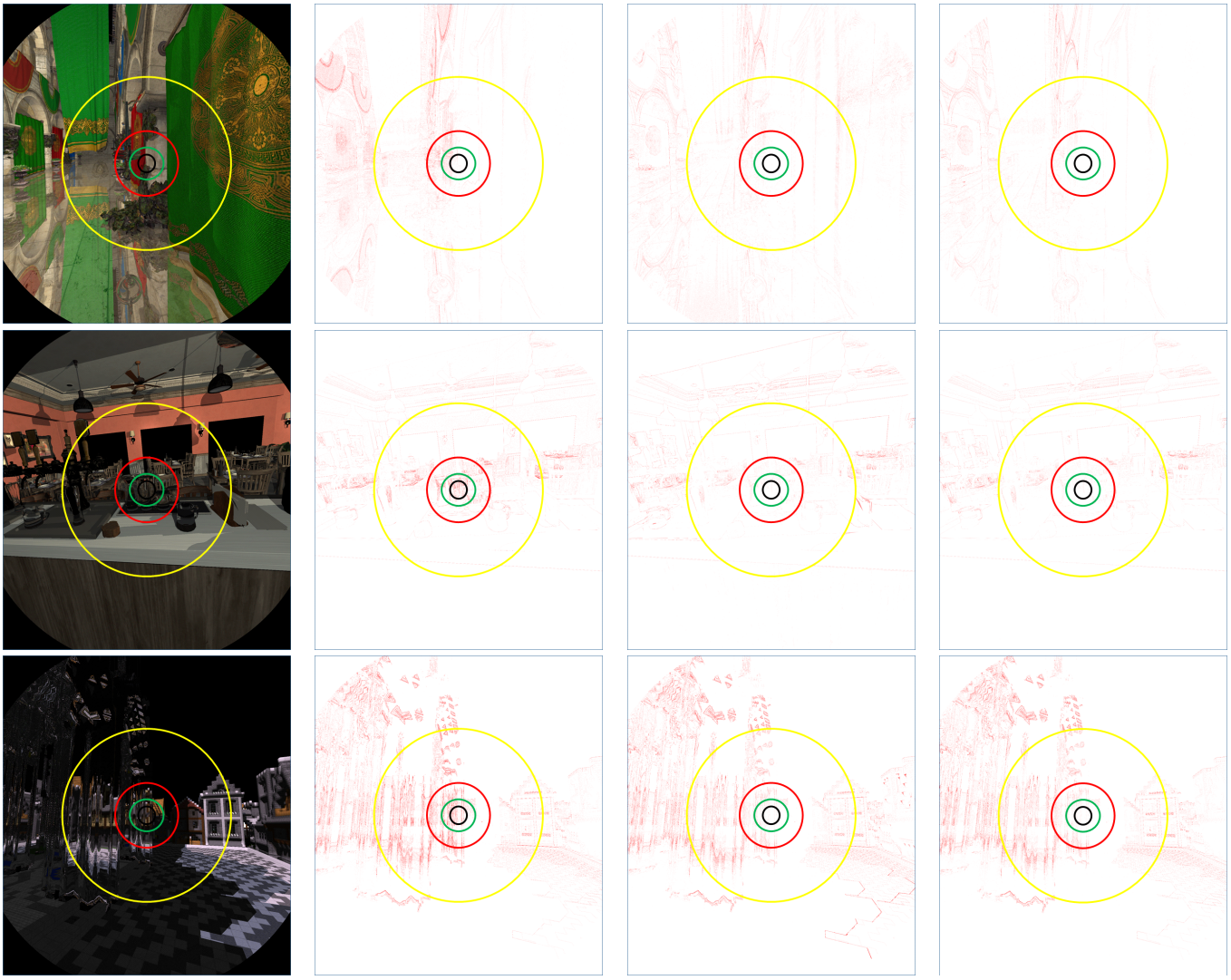


Figure 10. Comparison of rendering results (Crytek Sponza, Interior, and Rungholt)

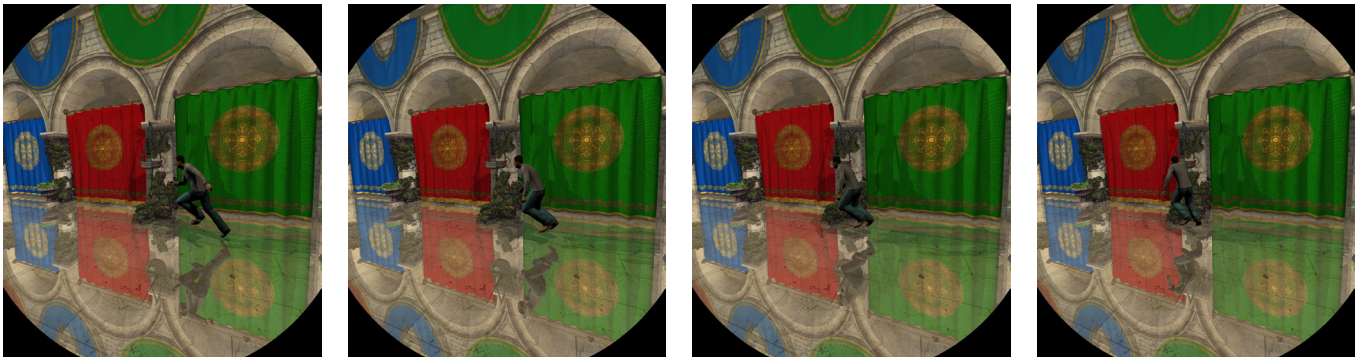


Figure 11. Dynamic scene rendering results (Crytek Sponza and Ben)

〈저자소개〉

김 영 욱

- 2012년 8월 한국외국어대학교 공과대학 컴퓨터공학과 학사
- 2012년 9월 ~ 현재 서강대학교 공과대학 컴퓨터공학과 석박통합과정
- 2016년 9월 ~ 2019년 8월 스타십벤딩머신(주) 연구원
- 관심 분야: 실시간 렌더링, 모바일 환경 렌더링, 가상/증강 현실
- https://orcid.org/0000-0002-4024-5841

임 인 성

- 1985년 2월 서울대학교 자연과학대학 계산통계학과 학사
- 1987년 5월 Rutgers - The State University of New Jersey 컴퓨터학과 석사
- 1991년 7월 Purdue University 컴퓨터학과 박사
- 1999년 7월 ~ 2000년 7월 University of Texas at Austin 연구 교수
- 1993년 3월 ~ 현재 서강대학교 공과대학 컴퓨터공학과 교수
- 관심 분야: 실시간 렌더링, 가상/증강 현실, GPGPU 컴퓨팅
- https://orcid.org/0000-0002-5611-925X