한국컴퓨터그래픽스학회
Korea Computer Graphics Society Vol. 24, No. 5, P.21~30

사운드 트레이싱을 위한 적응형 깊이 조절 알고리즘

김은재O 윤주원 정우남 김영식 박우찬*

세종대학교 컴퓨터공학과

한국산업기술대학교 게임공학과

(ejkim, juwon)@rayman.sejong.ac.kr, woonam@sju.ac.kr, kys@kpu.ac.kr, pwchan@sejong.ac.kr

Adaptive depth control algorithm for sound tracing

Eunjae KimO Juwon Yun Woonam Chung Youngsik Kim Woo-Chan Park*

Department of Computer Engineering. Sejong University

Department of Game & Multimedia Engineering, Korea Polytechnic University

요 약

본 논문에서는 현실감을 높이기 위한 청각적 기술로 기하학적 방법을 사용하는 광선 추적(ray-tracing) 기반의 3D Sound rendering기술인 Sound-tracing을 사용한다. Sound-tracing은 사운드 전파(sound propagation)단계에서 많은 비용이 든다. 사운드 전파 비용을 감소시키기 위해 제안하는 알고리즘은 이전 프레임들의 평균 유효 frame 수를 계산하고 그 수치를 기반으로 공간에 따른 depth를 조절하는 방법이다. 실험 결과 depth를 조절하지 않은 결과와 비교하면 음원이 실내에 있었을 때 path 손실률은 0.72%이고 탐색 및 충돌검사 단계(traversal & Intersection test)가 85.13%의 계산량 감소를 보이고 전체 frame rate는 4.48% 증가하였다. 음원이 실외에 있었을 때 path 손실률은 0%이고 탐색 및 충돌검사 단계가 25.01%의 계산량 감소를 보이고 전체 frame rate가 7.85% 증가하였다. 이는 path 손실률을 최소화하면서 렌더링 성능을 올릴 수 있었다.

Abstract

In this paper, we use Sound-tracing, a 3D sound technology based on ray-tracing that uses geometric method as auditory technology to enhance realism. The Sound-tracing is costly in the sound propagation stage. In order to reduce the sound propagation cost, we propose a method to calculate the average effective frame number of previous frames using the frame coherence property and to adjust the depth according to the space based on the calculated number. Experimental results show that the path loss rate is 0.72% and the traversal & Intersection test calculation amount is decreased by 85.13% and the frame rate is increased by 4.48% when the sound source is indoors, compared with the result of the case without depth control. When the sound source was outdoors, the path loss was 0% and the traversal & Intersection test calculation amount is decreased by 25.01% and the frame rate increased by 7.85%. This allowed the rendering performance to be increased while minimizing the path loss rate.

키워드: 사운드 렌더링, 사운드 트레이싱, 가상현실, 광선 추적

Keywords: sound rendering, sound tracing, virtual reality, ray tracing

1. 서론

최근 모바일, 그래픽스 및 감각의 입출력 등의 기술 발달로 가상현실 분야에 대한 관심이 급격하게 증가되고 있다. 대부분의 가상현실 관련 기술들은 시각적인 요소에만 집중되어 있다. 하지만, 현실감 있는 가상현실 환경을 지원하기 위해서는 시각적 공간감 이외에 청각적인 공간감의 재현이 필수적이다. 청각적 공간감을 재현하기 위해 사용되는 기술로는 멀티채널 오디오 시스템, 머리기반 전달함수(Head related transfer function), 3차원 기하모델 기반의 사운드 렌더링 등을 사용한다[1,2].

3D 사운드 재현을 위해 사용되는 멀티채널 오디오는 전용 스피커 시스템 설치 공간 및 비용 등과 같은 물리적인 제약이 많다. 또한, head mount display(HMD)에 이용되는 Figure 1의 머리 기반 함수는 주변 환경과 물체를 고려하지 않아 소리의

*corresponding author: Woo-Chan Park/Sejong University(pwchan@sejong.ac.kr)

Received : 2018.11.02./ Review completed : 1st 2018.11.25. / Accepted : 2018.11.28.
DOI : 10.15701/kcgs.2018.24.5.21
ISSN : 1975-7883(Print)/2383-529X(Online)

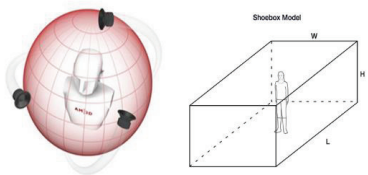

Figure 1 : Head Related Transfer Function based VR 3D sound system

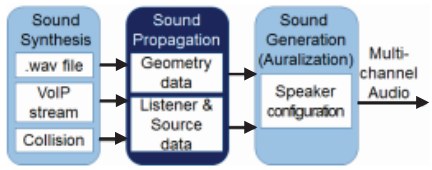

Figure 2 : Sound tracing pipeline

물리적인 특성을 반영하지 못한다.

3차원 기하모델 기반의 사운드 렌더링 기술은 가상의 공간에서 청취자의 위치, 음원의 개수와 위치, 주변 물체와 재질을 반영하여 시뮬레이션 한다. 이를 통해 소리의 물리적인 특성인 반사, 투과, 회절, 흡수를 재현하므로 청각적 공간감을 사용자에게 제공한다. 하지만, 실시간으로 주변 물체와 재질에 대해 물리 기반의 사운드 시뮬레이션을 하려면 계산 비용이 많이 들고, 전력 소모 또한 높다[3]. 제한적인 리소스를 가진 모바일 환경이나 standalone 형태의 HMD에서는 사운드 렌더링의 실시간 처리가 어려워지면 현실감 있는 소리의 몰입감을 사용자에게 제공할 수가 없다. 따라서, 다양한 플랫폼에서의 청각적 공간감 제공을 위해서는 전력 소모 및 계산 비용을 줄이는 것이 필수적이다.

본 논문에서는 사운드 렌더링의 전력 소모 및 계산 비용 이슈를 해결하기 위해 다음과 같은 방법을 제안한다. 사운드 렌더링의 처리 단계 중에서 계산량이 가장 많은 단계는 sound propagation 이며, 이 비용을 줄이기 위해 sound propagation의 frame coherence 특성을 이용하였다.

Frame coherence의 factor로 depth별 유효 평균 frame 수를 사용하였다. Depth는 ray와 object가 부딪혀서 생기는 반사 횟수를 의미하며 유효 프레임이란 유효 path가 존재하는 프레임을 의미한다. 이전 프레임들의 대해 depth 별로 유효한 reflection path 가 발견된 평균 frame 수를 이용하여 depth를 조절하고, 이를 통해 sound propagation 의 계산량을 줄여 성능을 향상시키는 방법을 제안하였다.

논문의 구성은 다음과 같다. 2장에서는 사운드 렌더링의 관련연구 및 배경에 대해서 설명하고, 3장에서는 sound propagation의 depth별 특성들을 분석을 하고. 4장에서는 depth를 조절 하는 알고리즘에 대해 설명하고, 5장에서는 실험 결과 및 검증을 제시한다. 마지막으로 6장에서는 결과 내용을 요약하고 결론에 대하여 논한다.

2. 관련 연구 및 배경

2.1 Sound tracing

Sound tracing은 3D 그래픽스에서 사용하던 ray tracing 기술과 sound processing 기술이 결합된 3D 사운드 렌더링 기술이다. Figure 2는 sound tracing의 파이프라인을 보여준다. 사운드 합성(sound synthesis)은 하드웨어 또는 소프트웨어를 이용하여 사용자의 상호작용에 따른 사운드 효과를 생성하는 단계이다. 사운드 전파(sound propagation)는 음원(sound source)에서 청취자까지 도달하는 과정을 시뮬레이션 하는 단계이다. 이 단계에서 음원 혹은 청취자로부터 ray를 쏘고 ray가 어떤 오브젝트와 충돌하면 해당 오브젝트의 재질에 따라 얼마나 반사, 흡수, 투과 등이 될 것인지를 계산 한다.

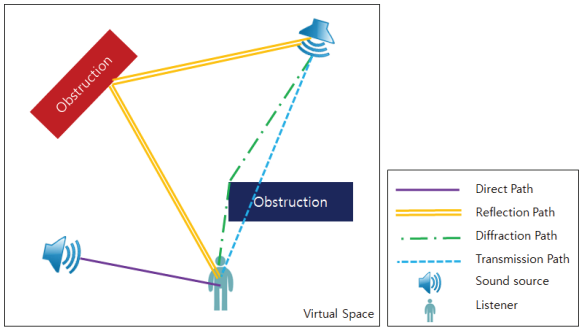

Figure 3 : Sound propagation paths

Figure 3과 같이 음원에서 청취자로 직접 들어오는 direct path, 물체를 투과하는 transmission path 그리고 다른 오브젝트와 hit 되면 hit 된 지점을 기준으로 조건에 따라 reflection 하거나 diffraction 하는 path를 생성시킬 수 있다. 현재 ray의 depth가 설정된 최대 depth보다 적을 경우 ray를 더 반사 시키는 것이 가능하다. Ray를 하나 더 생성시킨 후 다시 반사 시키는 작업을 재귀적으로 반복하게 된다. 이렇게 생성된 ray들 중 음원에서 청취자까지 도달하는 path들을 찾고 유효 path로 저장한다.

마지막으로 사운드 생성(sound generation) 단계는 이전 단계에서 구한 유효 path와 반사·투과·흡수 계수, 거리 감쇠, 시간과 같은 소리의 물리적인 특성을 이용하여 청취자에 스피커의 구성을 바탕으로 입력 음향을 재생성 하는 단계이다. Sound tracing pipeline에서 3개의 단계 중 사운드 전파 단계가

많은 비용이 들기 때문에 사운드 전파 단계를 가속화 시킬 필요성이 있다.

2.2 Sound Propagation

사운드 전파를 계산하기 위한 방법은 2가지가 있다[4]. 수치적인 계산을 기반한 방법(numerical method)과 기하학적인 계산을 기반한 방법(geometric method)이 있다. 수치적인 방법은 2차 편미분 방정식 형태인 파동 방정식[5][6]을 직접 푸는 방식이다. 이 방법은 가장 정확한 방법이지만 방정식을 푸는 비용이 커서 게임과 같은 인터랙티브 어플리케이션에는 적합하지 않다. 이런 어려움에도 불구하고 finite-difference time-domain(FDTD)을 이용해서 푸는 방식[7]과 frequency-domain[8]을 이용한 방식 등 수치적인 방법을 가속화 시킨 방법으로 사운드 전파를 푸는 방법들이 제안되었다. 하지만 이 방식들은 static scene으로 제한되어 있다.

기하학적인 방식은 2.1에서 설명했던 방식으로 ray tracing 기술을 이용한 시뮬레이션 방식이다. 이 기술도 여러 가지 방법들이 제안되었다. Beam을 이용해서 추적하는 beam tracing 방법 제안되었다[9]. Beam에 대한 비용 문제 때문에 frustum 형태로 쏘는 conservative frustum tracing[10]방식과 ray와 frustum을 합친 ray-frustum tracing[11] 방법 그리고 ray tracing만을 이용한 방식들도 제안되었다[12].

최근에는 음원과 청취자 두 쪽 모두 동시에 ray를 쏘는 방식인 bidirectional path tracing이 제안되었다[13]. 이 방식은 기존 방식보다 더 많은 유효 path를 찾을 수 있다. 또한 사운드 전파를 가속화 시키기 위해 소프트웨어가 아닌 하드웨어[3]도 제안되었다.

2.3 공간에 따른 소리의 반사

음원으로부터 청취자까지 오는 ray는 직접 오는 경우(직접음)가 있고 다른 오브젝트들에 의해 튕겨져서 오는 (간접음)경우가 있다. 직접음은 direct path에 해당하며 간접음은 reflection path, diffraction path, reverberation path에 해당한다.

Figure 4는 실내와 실외에서의 소리의 반사 모습을 볼 수 있다. 실내는 폐쇄적이기 때문에 하나의 ray가 여러 번 반사되는 모습을 볼 수 있다. 반대로, 실외의 경우 막혀있는 공간이 아니기 때문에 반사가 여러 번 되지 않는 것을 볼 수 있다. 즉, 주위에 오브젝트가 많고 막혀있는 부분이 많으면 많을수록 depth가 깊어진다는 것을 알 수 있다.

Figure 5는 Figure 4와 약간 다른 실내와 실외 공간을 보여주고 있다. 왼쪽의 그림은 실내지만 부분적으로 뚫려 있는 공간이다.

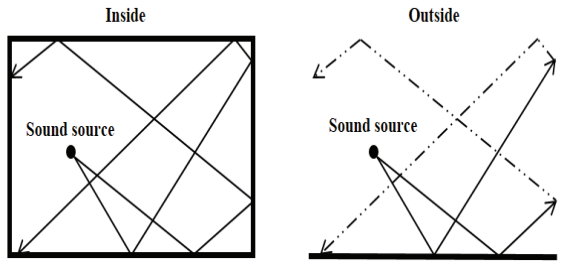

Figure 4 : Reflection of sound in indoor and outdoor

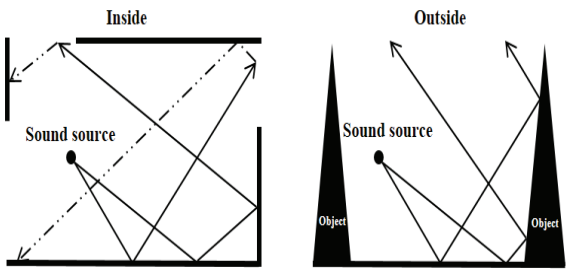

Figure 5 : Reflection of sound in a partially open indoors and outdoors surrounded by objects

이는 뚫려 있는 공간으로 인해 depth가 줄어든다. 반대로 오른쪽의 그림은 실외지만 가려지는 오브젝트가 있어서 어느 정도 반사가 생겨 이전보다 depth 좀 더 깊게 생긴다.

3. Sound propagation의 depth에 따른 특성들 분석

3.1 공간에 따른 depth별 평균 reflection ray 수

Sound propagation은 ray tracing 기반으로 사운드 렌더링을 수행한다. 사운드 렌더링을 수행하기 위해 sound propagation은 direct path, reflection path, reverberation path를 탐색한다. Sound propagation은 reflection path를 찾기 위해 청취자 위치를 기준으로 ray를 생성하고 ray의 반사를 최대 네 번까지 수행한다. 다양한 환경에서의 실험을 위해 Figure 6은 4가지 환경을 보여주고 있다. Sibenik, Concert hall, Bootcamp (Inside)의 경우 실내 환경이며 Bootcamp (Outside) 실외 환경이다[14][15].

Sound propagation 실험을 위해 GSound 기반에 sound propagation simulation software를 사용하였다[16]. Sound propagation은 reflection path를 찾기 위해 1,000개의 ray를 생성한

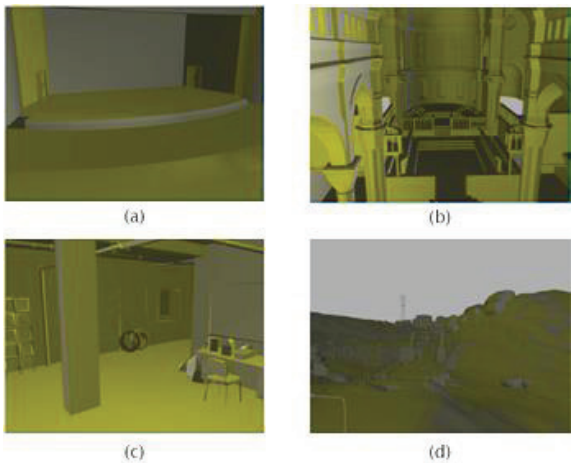

Figure 6 **:** (a) : Concert hall – 24k triangle, (b) : Sibenik - 75k triangle (c) & (d) : Bootcamp(Inside) & Bootcamp(Outside) – 130k triangle

Table 1 : Reflected average ray number per each depth of reflection path

Scene	Depth1	Depth2	Depth3	Depth4
Sibenik	992	989.8	986.44	989.22
Concert hall	992	991.92	991.8	991.2
Bootcamp(Inside)	992	991.12	975.72	959.02
Bootcamp(Outside)	992	671.63	297.13	186.6

다. 생성된 모든 ray들이 최대 depth까지 반사된다면 한 프레임 동안 약 4,000번의 반사가 이루어진다. Sound propagation이 reflection path를 찾기 위해 모든 반사에 대한 계산을 수행한다. Reflection path를 찾기 위한 ray의 depth를 줄인다면 ray의 반사 횟수는 감소한다. 반사 횟수가 줄어들면 sound propagation의 계산 량은 감소하고 계산량이 줄이든 만큼 성능이 향상 된다.

Table 1은 청취자와 음원에 다수의 위치에서 네 개의 scene에 대해 사운드 렌더링을 수행하여 얻은 결과이다. 사운드 렌더링은 각 scene에 대하여 10프레임 동안 수행되었다. 10프레임 동안 reflection path를 찾기 위해 청취자로부터 ray를 생성하고 각 depth 별로 반사된 ray의 수를 평균 1프레임의 수치로 나타냈다.

예를 들면, Sibenik scene은 한 번 반사된 ray가 평균 992개, 두 번 반사된 ray가 평균 989.8개, 세 번 반사된 ray가 평균 986.44개이다. Table 1에서 실내 환경인 Sibenik, Concert hall Bootcamp(Inside)에서는 depth가 늘어나도 많은 ray 반사가 생긴다. 반면에 실외 환경인 Bootcamp(Outside)의 경우 depth가 늘어날수록 ray 반사수가 줄어드는 것을 알 수 있다. 결과적으로 depth를 조절이 가능하면 유효한 reflection path의 손실을 최소화 하면서 sound propagation 의 계산 양을 줄여 sound propagation의 성능을 향상시킬 수 있다.

3.2 Frame coherence

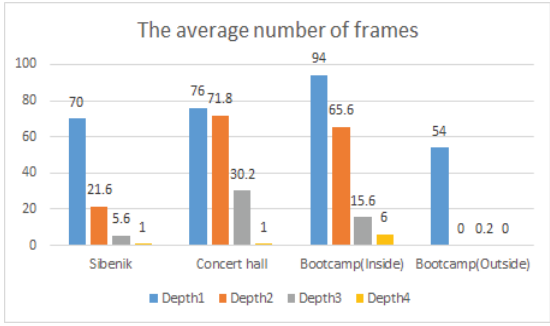

Figure 7 : The average number of frames for which a valid reflection path was found by each depth

Table 2 : Probability that the current valid frame equals the previous valid frame for 100 frames

Scene	Depth1	Depth2	Depth3
Bootcamp(Inside)	70.71%	90.91%	86.87%
Bootcamp(Outside)	96.97%	100.00%	100.00%

Figure 7는 네 개의 scene에 대해서 sound propagation unit이 100 frame 동안 사운드 렌더링을 여러 번 수행한 결과이다. 각각의 depth별로 유효한 reflection path를 발견한 평균 frame 수를 나타낸다. 예를 들면, Sibenik scene은 발견된 유효한 reflection path들 중 한 번 반사된 path를 발견한 frame 수가 평균 70개이고 두 번 반사된 path를 발견한 frame 수가 평균 21.6개이다.

Figure 7의 결과를 보면, depth가 늘어날수록 평균 유효 frame 수는 줄어들고 상대적으로 depth 4에서 유효한 reflection path를 발견한 frame 수가 다른 depth에 비해 적은 것을 확인할 수 있다. 이것은 reflection path를 찾기 위해 청취자의 위치에서 생성된 ray가 네 번 반사되어 음원까지 도달하기 어려운 것을 의미한다.

Table 2는 Bootcamp(Inside)와 Bootcamp(Outside)에 대해서 100 frame 동안 사운드 렌더링을 수행하면서 현재 유효 frame이 이전 유효 frame과 같을 확률을 보여주고 있다. 예를 들면 현재 frame에 depth1에서 reflection path가 존재하고 이전 프레임에서도 depth1에서 reflection path가 존재하면 유효 frame들은 서로 같다고 판별한다. 반대로 현재 frame에서 depth1에 reflection path가 존재하고 이전 frame에 depth 1에서 reflection path가 존재하지 않는다면 두 유효 frame은 서로 다르다고 판별한다.

이렇게 판별을 100프레임 동안 수행하면서 서로 유효

frame이 같을 확률을 계산하였다. Inside의 경우 depth가 증가하면 할수록 높은 확률로 유효 프레임이 같았다. Outside의 경우에는 모든 depth에서 유효 frame이 높은 확률로 같았다. 즉, 짧은 시간 내에 frame들 간의 유효 프레임이 같을 확률이 높다는 것을 의미하고 이것은 곧 frame coherence가 높다는 것을 알 수 있다. 뿐만 아니라 어떤 프레임에서 이 프레임이 유효한 프레임인지 아닌지 판별하면 그 다음 프레임이 유효한 프레임인지 아닌지를 어느 정도 예측이 가능하다는 것을 알 수 있다.

4. 제안하는 depth 조절 알고리즘

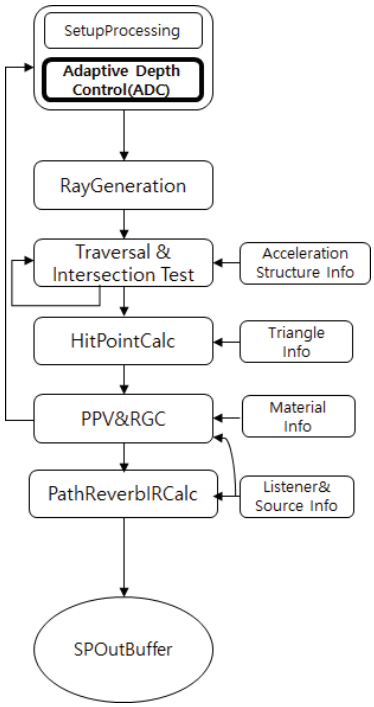

Figure 8 : Proposed Sound Propagation Flow chart

Figure 8는 전형적인 sound propagation의 흐름도에 제안하는 Adaptive Depth Control(ADC)를 SetupProcessing안에 넣은 그림이다. SetupProcessing은 sound propagation의 모드를 관리하고 필요로 하는 ray 정보를 준비해서 RayGeneration 단계에 넘긴다. RayGeneration은 이전 단계에서 받은 정보를 이용하여 ray를 생성하고 ray 데이터를 Traversal&Intersection Test 단계에 넘긴다.

Traversal&Intersection Test는 생성된 ray 쏘고 ray와 object간의 충돌 검사를 한다. Ray를 쏠 때는 가속구조체를 정보를 받아서 탐색을 가속화시킨다. HitPointCalc는 충돌된 지점을 계산하고 해당 hit triangle ID에 대한 triangle 정보를 요청한다.

PPV&RGC는 ray 특성에 따라서 Propagation Path Validator(PPV)나 Reverb Geometry Collector(RGC) 둘 중 하나 처리가 된다. PPV는 path가 유효한지 검사를 한다. RGC는 잔향음에 필요한 시뮬레이션을 한다.

마지막으로 PathReverbIRCalc은 PPVnRGC로 받은 Impulse Response(IR) 생성 정보를 이용하여 소리 감쇠나 지연 시간과 같은 path/reverb sound rendering information 데이터를 생성한다. 생성된 데이터는 SPOutBuffer에 쓴다.

본 논문에서 제안하는 알고리즘은 3.2에 frame coherence 특성을 이용한다. frame coherence 특성을 이용한 factor로 이전 frame들의 평균 유효 frame 수를 사용하여 depth를 조절한다. 이를 위하여 SetupProcessing 단계에서 depth 조절을 위한 Adaptive Depth Control(ADC)를 추가하였다.

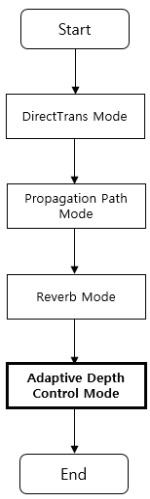

Figure 9 : SetupProcessing Mode

Figure 9는 SetupProcessing의 모드 단계를 보여준다. 직접음/투과음 모드와 반사음/회절음 모드 그리고 잔향음 모드를 거치면서 ray를 생성하기 위한 정보들을 준비한다. ADC Mode는 Reverb Mode를 다 끝낸 후에 수행된다.

Figure 10은 제안하는 알고리즘 Flow chart를 보여준다. 먼저 ADC가 활성화 된지 체크를 하고 활성화가 되어있으면 ADC 수행하고 그렇지 않다면 나가게 된다. ADC가 활성화 되어있다면 ADC에 필요한 변수들을 초기화한다. Check valid frame 단계에서는 이전 프레임의 정보를 가져와서 depth별로 유효path가 있었는지 체크를 한다. Buffering 단계는 이전 단계에서 구한 정보들을 담아 쌓아놓는다. 만약 현재까지

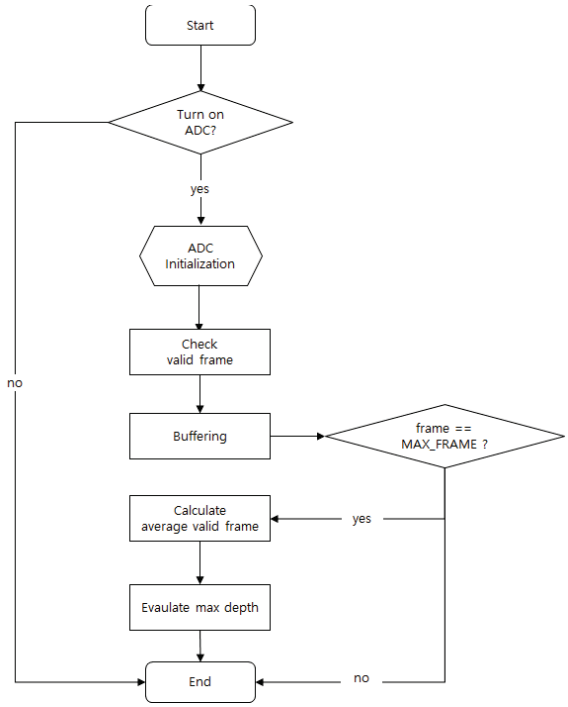

Figure 10 : Proposed Algorithm Flow chart

ADC를 지나간 프레임 수가 최대 프레임 수(MAX_FRAME)랑 같아진다면 버퍼에 담아두었던 정보를 가지고 다음 단계로 넘어간다. MAX_FRAME에 도달하지 못했다면 다른 계산 없이 끝낸다. MAX_FRAME에 도달했다면 Calculate average valid frame 단계에서 depth별로 평균을 구하고 그 값이 한계값(0.5)보다 크다면 해당 프레임에 depth는 유효하다고 본다. Evaluate max depth에서 내림차순으로 depth를 검사하면서 유효했는지 체크하고 유효했으면 그 값을 최대 max depth로 설정한다.

5. 실험 결과 및 검증

5.1 실험 방법 및 환경

본 논문에서는 depth 조절 실험에 의미가 있는 복합적인 공간을 가진 환경인 Bootcamp 환경에서 실험하였다. 또한 음원에 대한 위치를 실내 1곳과 실외 1곳을 사용하고 청취자 위치의 경우 실내 1곳과 실외 2곳을 사용하여 움직이게 하였다. 100 frame 동안 사운드 렌더링을 수행하면서 청취자의 위치를 실내에서 실외로 혹은 실외에서 실내로 이동하였다.

처음 최대 depth 세팅은 3으로 놓았고 MAX_FRAME은 5로 놓았다. 최소 depth 값은 1로 놓고 실험하였다. ADC로 일단 조절된 max depth값은 ADC를 구했던 시간을 포함해서 1초 동안 유지되며 본 논문은 1초의 30frame을 기준으로 놓았기 때문에 조절된 max depth값은 다음 25개의 frame에 적용된다. 실험 환경은 Intel(R) Core(TM) i7-7700 CPU @ 3.60GHz, RAM 16GB, Window 10에서 실험하였다.

5.2 ADC에 따른 유효 path 수 결과

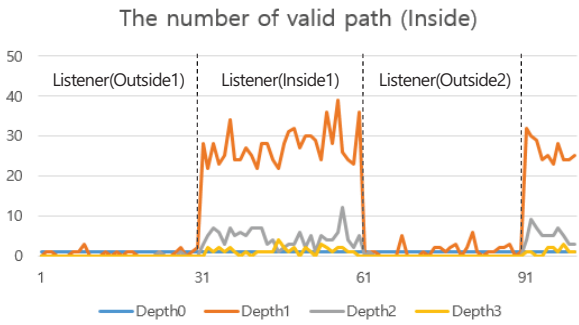

Figure 11 : The effective path number during 100 frames when the sound source was in the room without adjusting the depth

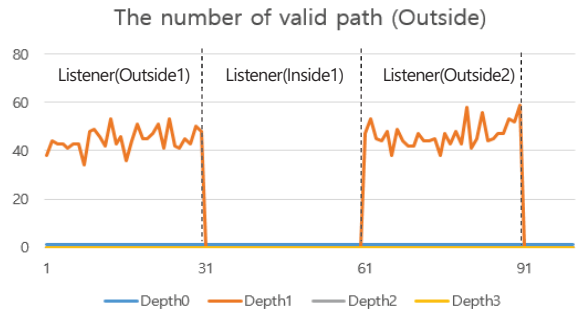

Figure 12: The effective path number during 100 frames when the sound source is outdoors without adjusting depth

Figure 11과 Figure 12에서는 depth을 조절하지 않았을 때 100 frame 동안에 유효 path 수를 보여주고 있다. 세로축은 유효 path 수이고 가로축은 frame이다. 청취자의 위치는 0부터30 frame까지 실외, 31부터 60 frame까지 실내, 61부터 90 frame까지 실외, 91부터 100 frame까지 실내에 있다.

Figure 11은 음원이 실내에 위치하였을 때 결과를 보여주고 있다. 실내에 있었을 때 depth2와 depth3에서 유효 path가 어느 정도 존재하는 것을 볼 수 있다. 이는 청취자와 음원이 같은 실내 공간에 있기 때문에 depth가 높은 곳에서도 유효 path를

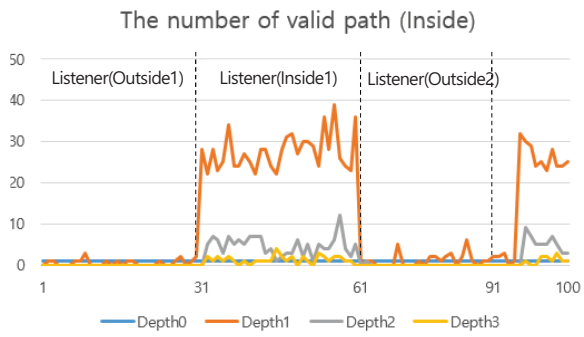

Figure 13 : The effective path number when the depth was adjusted and the sound source was indoors.

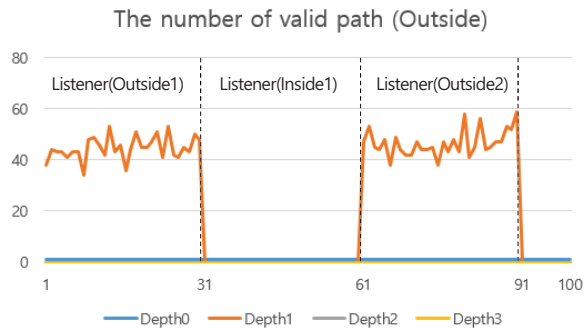

Figure 14 : The effective path number when the depth was adjusted and the sound source was outdoors.

찾을 수 있었다. Figure 12는 음원이 바깥에 위치하였을 때 결과를 보여준다. 청취자가 실외에 있었을 때는 depth1에서 유효 path들을 찾는 반면 실내에 있었을 때는 depth0인 direct path를 제외하면 못 찾는 것을 볼 수 있다.

Figure 13와 Figure 14는 ADC를 한 후 유효 path 수의 결과를 보여준다. depth를 조절한 것과 조절하지 않았을 때를 비교하면, 음원이 실내 환경에 있을 경우에 수치상으로 약간의 차이는 가지지만 그 차이가 매우 적어서 그림 외관상 거의 차이를 보이지 않는다. 음원이 실외에 있을 경우에도 depth를 조절하기 전과 유효 path는 동일하다.

5.3 ADC에 따른 depth 결과

Table 3 : Accumulative effective path count during 100 frames

	Depth1	Depth2	Depth3	Sum
Inside	1146	197	50	1392
ADC+ Inside	1146	187	49	1382
Outside	2736	0	0	2736
ADC + Outside	2736	0	0	2736

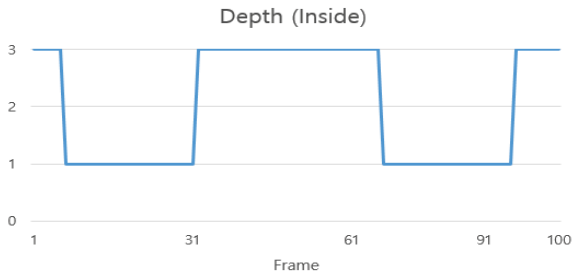

Figure 15: The depth adjustment value when the sound source was indoors

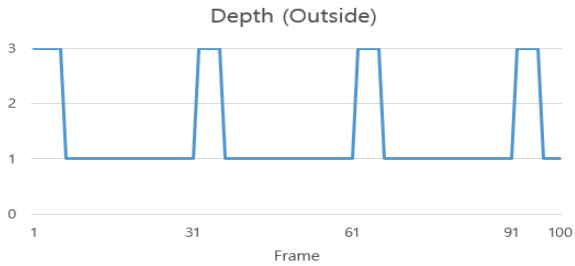

Figure 16 : The depth adjustment value when the sound source was outdoors

Table 3에서는 누적된 유효 path 수를 depth별로 정리한 표이다. 음원이 실내에 있을 경우 depth를 조절했을 때 depth2와 depth3에서 약간의 차이가 생긴다. Depth2에서 10개의 path가 손실이 되었고 depth3에서는 1개의 path가 손실이 됐다. 반면에 음원이 실외에 있을 경우에는 depth2 이상부터는 path를 찾지 못하기 때문에 손실 되는 path가 존재 하지 않는다. Figure 15과 Figure 16는 ADC에 의해 depth를 조절한 결과를 보여준다. 가로축은 frame 을 나타내며 세로축은 해당 frame에 설정된 depth 결과 보여주고 있다.

Figure 15에서 음원이 실내에 있을 경우 depth 조절한 결과이다. ADC를 통해 처음 depth가 1로 설정된다. 이는 청취자가 실외에 있어서 깊은 depth를 가진 유효 path를 갖지 못하기 때문이다. 30 frame에서 청취자가 실내로 이동하게 되어 반사가 많아지게 되고 유효 frame들이 depth가 깊은 곳에서도 생기기 때문에 depth가 3으로 올라간다. 반면에 60 frame이후에 다시 실외로 갔을 때 유효 frame 들이 줄어들어 depth가 1로 낮아지는 것을 볼 수 있다.

Figure 16은 음원이 실외에 있을 때 depth를 조절한 결과이다. 음원이 실외에 있으면 반사가 덜 되기 때문에 depth가 낮게 설정이 된다. ADC를 통해 처음 depth가 1로 설정이 된다. 그 이후 30 frame부터 60 frame까지 청취자는 실내 환경으로

이동된다. 음원이 외부에 있고 청취자는 실내에 있기 때문에 유효 path를 찾기 힘들다. 이 때문에 depth가 낮게 설정돼야 하지만 30 frame 과 60 frame 부분에 depth가 3으로 잡혀있다. 이는 depth를 조절하기 위해 다시 갱신 타이밍을 잡기 때문에 나타난 현상이다. 약간의 frame time을 손해를 보더라도 좀 더 정확한 depth 조절을 위해 trade-off를 한 것이다.

5.4 ADC에 따른 depth별 ray 수 결과

Table 4 : Comparison of the number of ray according to presence or absence of depth control and environment

	Depth1	Depth2	Depth3
Inside	99220	83937	66469
ADC + Inside	99220	46441	42460
Outside	99220	83937	66469
ADC + Outside	99220	18002	14731

Table 4 은 depth 조절 유무 및 환경에 따른 ray 수를 비교하고 있다. 청취자의 위치는 그대로 두고 음원의 위치를 실내 혹은 실외로 이동시켰다. ADC 를 적용하지 않은 2 행과 4 행의 결과는 청취자의 위치를 이동시키지 않았기 때문에 같다. 반면에, depth 를 조절한 결과는 음원이 실내인 경우 depth2 의 ray 수는 약 44.67%, depth3 의 ray 수는 36.12%만큼 줄어들었다. 음원이 실외인 경우 ray 수는 depth2 에서 약 78.55%, depth3 에서 약 77.83% 줄었다. 즉, depth 를 조절함으로써 많은 ray 의 수를 줄일 수 있는 것을 볼 수 있다.

5.5 ADC에 따른 frame time & frame rate 결과

Table 5 Comparison of the frame time according to presence or absence of depth control and environment

	Frame time(ms)
Inside	110419
ADC + Inside	105472
Outside	110496
ADC + Outside	101814

Table 5는 depth 조절 유무와 환경에 따른 frame time을 나타냈다. Frame time은 100 frame 기준으로 sound propagation을 수행한 시간을 의미한다. 3행과 5행은 depth를 조절했을 때 frame time을 나타내고 있다. Depth를 조절한 후 frame time은 depth 조절 하기 전과 비교하면 음원이 실내인 경우에는 약 4.48% 감소하였고 음원이 실외인 경우에는 약 7.85% 감소하였다. Frame rate 기준으로는 음원이 실내인 경우에는 약 4.48% 증가하였고 음원이 실외인 경우에는 약 7.85% 증가하였다.

5.6 ADC에 따른 Sound Propagation 단계별 감소율

Table 6 : Sound propagation step-by-step performance reduction rate according to presence or absence of Depth (Source Outside)

	Non-ADC(ms)	ADC(ms)	차이(%)
SetupProcessing	35.68	24.84	43.63
RayGeneration	78.49	46.89	67.37
T&I	6743.91	3642.82	85.13
HitPointCalc	117.90	66.18	78.16
PPV&RGC	149315.58	146747.40	1.75
PathReverbIRCalc	18.75	11.89	57.73
SPOutBuffer	16.43	9.90	65.93

Table 7 : Sound propagation step-by-step performance reduction rate according to presence or absence of Depth (Source inside)

	Non-ADC(ms)	ADC(ms)	차이(%)
SetupProcessing	37.08	32.69	13.43
RayGeneration	84.07	67.69	24.20
T&I	7375.99	6020.72	22.51
HitPointCalc	135.64	112.59	20.47
PPV&RGC	148240.78	147070.74	0.80
PathReverbIRCalc	20.07	16.95	18.38
SPOutBuffer	18.29	15.02	21.82

Table 6과 Table 7은 depth 조절에 따른 Sound propagation 단계별 성능 감소율을 보여주고 있다. 실험은 Bootcamp(Source Inside & Outside) 환경을 사용하였고 100 frame 사운드 렌더링을 실행한 결과와 depth를 조절했을 때 와 조절 하지 않았을 때 단계별로 실행 시간을 기록했다. 실험 결과 Source Outside 일 때는 SetupProcessing는 29.03%, RayGeneration 는 42.21%, T&I 은 47.40%, HitPointCalc는 89.85%, PPV&RGC는 5.59%, PathReverbIRCalc는 45.11%, SPOutBuffer는 45.35% 성능의 차이를 보였다. Source Inside 일 때는 SetupProcessing이 13.43%, RayGeneration이 24.20%, HitPointCalc는 20.47%, PPV&RGC는 0.80%, PathReverbIRCalc는 18.38%의 성능 차이를 보였다.

6. 결론

본 논문에서는 사운드 렌더링의 성능 향상을 위한 depth 조절 방법을 제안하고 분석했다. Depth 조절을 위해 frame coherence의 특성을 이용하였고 이것을 위한 factor로 이전 frame들의 평균 유효 frame수를 사용하였다. 제안한 방법으로 depth 조절 실험 결과, path 손실률의 경우 음원이 inside일 때 0.72%이고 음원이 outside인 경우 0%의 손실률을 보였다.

성능의 경우 음원이 실내인 경우 frame rate는 4.48% 증가하였고 음원이 실외인 경우 frame rate는 7.85% 증가하였다. 즉, Depth를 조절하여 path손실은 최소화하면서 동시에 frame

rate를 높일 수 있었다. 그러나 ray 반사 수가 줄어든 것에 비해 frame time의 줄어든 수치가 낮다. 이는 Sound propagation 에서 ray 반사 수를 줄였을 때 성능적 이득을 많이 보지 못하는 부분들이 있다는 것을 알 수 있다. Validator & Reverb Geometry Collector(PPV&RGC)는 사운드 전파 단계에서 많은 시간을 사용하지만 depth 조절하였을 때 성능 증가가 미미했다.

향후 연구로는 Sound propagation 에서 많은 병목을 가지는 Propagation Path Validator & Reverb Geometry Collector에서 Sort 알고리즘의 성능을 올리는 연구를 할 계획이다.

감사의 글

본 연구는 과학기술정보통신부 및 정보통신기술진흥센터의 대학 ICT 연구센터육성지원사업의 연구결과로 수행되었음(IITP-2018-2016-0-00312)

References

[1]. M. Vorlander, “Auralization: Fundamentals of Acoustics, Modelling Simulation, Algorithms and Acoustic Virtual Reality,” *Springer*, 2008.

[2]. A. Febrettia, A. Nishimotoa, T. Thigpena, J. Talandisa, L. Longa, J. Pirtlea, T. Peterkaa, A. Verloa, M. Browna, D. Plepysa, D. Sandina, L. Renambota, A. Johnsona, and J. Leigha, “Cave2: a hybrid reality environment for immersive simulation and information analysis,” in *Proceedings SPIE Electronic Imaging*, vol. 8649, p. 864903, 2013.

[3]. D. Hong., T. H. Joo, W.-C. Park. "Real-time sound propagation hardware accelerator for immersive virtual reality 3D audio," *Proceedings of the 21st ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, p. 20, 2017.

[4]. Funkhouser, Thomas, Nicolas Tsingos, and Jean-Marc Jot, "Survey of methods for modeling sound propagation in interactive virtual environment systems," *Presence and Teleoperation*, 2003.

[5]. RAGHUVANSHI, Nikunj; SNYDER, John. “Parametric wave field coding for precomputed sound propagation”. *ACM Transactions on Graphics (TOG)*, vol. 33, no. 4, p 38, 2014.

[6]. Savioja, Lauri. "Real-time 3D finite-difference time-domain simulation of low-and mid-frequency room acoustics," *13th Int Conf on Digital Audio Effects*, vol. 1, p. 75, 2010.

[7]. MEHRA, Ravish, et al., “Wave-based sound propagation in large open scenes using an equivalent source formulation,” *ACM Transactions on Graphics (TOG)*, vol. 32, no. 2, pp 19:1-19:13 2013.

[8]. Funkhouser, T., Carlbom, I., Elko, G., Pingali, G., Sondhi, M., & West, J. “A beam tracing approach to acoustic modeling for interactive virtual environments,” *In Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, pp. 21-32, 1998.

[9]. Funkhouser, Thomas, et al. "A beam tracing method for interactive architectural acoustics," *The Journal of the acoustical society of America*, vol. 115, no.. 2, pp. 739-756, 2004

[10]. Lauterbach, Christian, Anish Chandak, and Dinesh Manocha. "Interactive sound rendering in complex and dynamic scenes using frustum tracing," *IEEE Transactions on Visualization and Computer Graphics,* vol. 13, no. 6, pp 1672-1679, 2007.

[11]. Chandak, Anish, et al. "Ad-frustum: Adaptive frustum tracing for interactive sound propagation." *IEEE Transactions on Visualization and Computer Graphics*, vol. 14, no. 6, pp. 1707-1722, 2008.

[12]. C. Schissler and D. Manocha, “Interactive sound propagation and rendering for large multi-source scenes,” *ACM Transactions on Graphics (TOG)*, vol. 36, no. 1, p. 2, 2016.

[13]. C. Cao, Z. Ren, C. Schissler, D. Manocha, and K. Zhou, “Interactive sound propagation with bidirectional path tracing,” *ACM Transactions on Graphics (TOG)*, vol. 35, no. 6, p. 180, 2016.

[14]. J. Yun, CG Kim, and W.-C. Park. "The design of a Software Development Kit for Virtual Reality 3D Audios," *Proceedings of the 2nd EEECS*, pp. 44-46, 2016.

[15]. Taylor, Micah T.,et al. "Resound: interactive sound rendering for dynamic virtual environments," *Proceedings of the 17th ACM international conference on Multimedia. ACM*, pp. 271-280 2009.

[16]. Schissler, Carl, and Dinesh Manocha. "Gsound: Interactive sound propagation for games," *Audio Engineering Society Conference: 41st International Conference: Audio for Games. Audio Engineering Society*, 2011.

〈저자소개〉

김 은 재

- 2017년 한국산업기술대학교 게임공학과 학사
- 2017년~현재 세종대학교 컴퓨터공학과석박사통합과정
- 관심분야: 실시간 사운드 트레이싱, 컴퓨터 그래픽스, 컴퓨터구조, 게임엔진

윤 주 원

- 2012년 한국산업기술대학교 게임공학과 학사
- 2012년~현재 세종대학교 컴퓨터공학과 석박사통합과정
- 관심분야: 실시간 사운드 트레이싱, 모바일 GPU, 컴퓨터 그래픽스, 게임엔진

박 우 찬

- 1993년 연세대학교 컴퓨터과학과 학사
- 1995년 연세대학교 컴퓨터과학과 석사
- 2000년 연세대학교 컴퓨터과학과 박사
- 2000년~2001년 연세대학교 아식공동설계연구소 연구원
- 2001년~2003년 연세대학교 컴퓨터과학과 연구교수
- 2003년~현재 세종대학교 컴퓨터공학과 교수
- 관심분야: GPU 하드웨어 구조, 실시간 그래픽스, 컴퓨터 구조

정 우 남

- 2000년 연세대학교 컴퓨터과학과 학사
- 2002년 연세대학교 컴퓨터과학과 석사
- 2011년 연세대학교 컴퓨터과학과 박사
- 2011~2017년 ㈜실리콘아츠 책임연구원
- 2017년~현재 세종대학교 컴퓨터공학과 수석연구원
- 관심분야: GPU 하드웨어 구조, 실시간 그래픽스, 컴퓨터 구조, 전역조명

김 영 식

- 1993년 연세대학교 컴퓨터과학과 학사
- 1995년 연세대학교 컴퓨터과학과 석사
- 1999년 연세대학교 컴퓨터과학과 박사
- 1999년~2005년 삼성전자 System LSI 책임연구원
- 1999년~2005년 삼성전자 System LSI 책임연구원
- 2013년 University of Pittsburgh 방문교수
- 2005년~현재 한국산업기술대학교 교수
- 관심분야: 게임기구조, 컴퓨터구조, 3차원 그래픽 가속기, 임베디드 시스템