The Journal of the Convergence on Culture Technology (JCCT)
Vol. 10, No. 6, pp.641-647, November 30, 2024. pISSN 2384-0358, eISSN 2384-0366

http://dx.doi.org/10.17703/JCCT.2024.10.6.641
JCCT 2024-11-78

VR기기에서 3D 콘텐츠 최적화 방안 연구

Research on 3D Content Optimization Methods in VR Devices

한성진*, 김민지**, 김정이***

Seong-Jin Han*, Min-Jin Kim, Jung-Yi Kim*****

요 약 본 연구에서는 Unity 3D에 적용 가능한 최적화 방안을 조사하고 실험을 통해 최적화가 적용된 콘텐츠의 효율성을 평가하였다. 그리고 이를 적용하여 우주를 배경으로 한 생존을 주제로 하는 가상현실 콘텐츠를 제작하였다. 해당 실험에서는 가상현실 우주 생존 콘텐츠를 Android OS 플랫폼으로 빌드한 뒤, Unity Profiler를 활용하여 CPU 처리 속도, 총 메모리 사용량, 드로우콜, FPS를 측정하고, 최적화 기술 적용 전후의 성능을 비교하여 표로 나타내었다. 그 결과 본 연구에서 제안하고 있는 '라이트 베이크', '오클루전 컬링', '오브젝트 풀링', 'JSON을 활용한 비동기적 씬 로딩' VR 기기에서 3D 콘텐츠 최적화 방안이 매우 효율적인 것을 알 수 있었다. 본 연구는 VR 기기에서 활용될 3D 콘텐츠 개발의 고질적인 문제인 최적화 문제를 Unity 엔진에서 간편하게 해결할 수 있는 방안을 제안함으로써 인디 게임 개발자나 Unity 엔진에 익숙하지 않은 사람들이 VR 콘텐츠를 개발할 때 유용한 가이드라인으로 활용될 것을 기대한다.

주요어 : Unity, VR, 가상현실, 최적화

Abstract In this study, we investigated optimization methods applicable to Unity 3D and evaluated the efficiency of optimized content through experiments. And by applying this, we created virtual reality content with the theme of survival set in space. In this experiment, virtual reality space survival content was built on the Android OS platform, and CPU processing speed, total memory usage, draw call, and FPS were measured using Unity Profiler, and the performance before and after applying the optimization technology was compared and presented in a table. . As a result, it was found that the 3D content optimization methods proposed in this study, including 'Lighting Baking', 'Occlusion Culling', 'Object Pooling', and 'Asynchronous Scene Loading Using JSON', were very efficient in VR devices. This study proposes a way to easily solve the optimization problem, which is a chronic problem in the development of 3D content to be used in VR devices, in the Unity engine, providing a useful guideline for indie game developers or people unfamiliar with the Unity engine when developing VR content. I hope it will be used.

Key words : Unity, VR, virtual reality, optimization

*준회원, 성결대학교 미디어소프트웨어학과 재학생 (제1저자)
**준회원, 성결대학교 미디어소프트웨어학과 재학생 (참여저자)
***정회원, 성결대학교 미디어소프트웨어학과 조교수 (교신저자)*Corresponding Author: ecesss@sungkyul.ac.kr
Dept. of Media Software, Sungkyul Univ, Korea
Received: August 3, 2024 / Revised: September 8, 2024
Accepted: November 5, 2024
접수일: 2024년 8월 3일, 수정완료일: 2024년 9월 8일
게재확정일: 2024년 11월 5일

Ⅰ. 서 론

현대 사회에 들어서 가상현실(VR)기술이 발전함에 따라 혁신적인 VR 기기들이 지속해서 출시되고 있다. 이에 따라 VR 기술을 활용해 실제적인 경험을 제공하는 의료·헬스케어 분야[1], 영상 콘텐츠가 아닌 가정폭력에 대한 인식을 높이는 효과적인 방법[2]과 인터렉션 기능을 통한 몰입형 교육 분야[3], 패션 매장을 활용해 시공간의 제약을 넘어 쇼핑이 가능한 디지털 패션 산업 분야[4] 등의 다양한 콘텐츠가 생산되고 있다. VR이란 Virtual Reality의 약자로 컴퓨터 기술을 활용해 가상의 환경을 구현해 실제적인 공간적·시간적 경험이 가능한 환경을 말한다[5]. 따라서 현실적이고 몰입감 있는 경험을 제공하여 사용자가 가상 세계에 완전히 몰입하게 되고, 현실과 가상의 경계를 모호하게 한다[6]. 가상현실 콘텐츠는 사용자에게 각기 다른 화면을 제공함으로써, 시야를 조절하고 넓은 시야각을 통해 몰입감 있는 공간을 형성하는 HMD(Head-Mounted-Display)를 이용해야 한다[7].

VR 기술의 발전은 새로운 가능성을 열어주고 있지만, 동시에 여러 기술적 과제를 안겨준다. VR 환경은 일반적인 2D 모니터 환경보다 훨씬 높은 성능 요구사항을 가지며, 사용자에게 높은 몰입감을 제공해야 하므로 프레임 속도가 일정하게 유지되지 않으면 사용자는 어지러움이나 멀미를 느낄 수 있다. 또한 콘텐츠 제작 시 시간이 지남에 따라 규모가 커지게 될 경우 HMD의 하드웨어 제약으로 인해 그래픽 성능이 제한될 수 있어 메모리 사용과 CPU/GPU 부하를 최소화하는 것이 필수적이다. 이러한 이유로 Unity 3D를 활용해 VR 콘텐츠를 제작할 시 최적화가 필수적으로 이루어져야 한다. 하지만 이러한 VR 환경의 성능 문제를 해결하기 위한 문헌 자료가 많지 않은 것이 현실이다.

따라서 본 연구에서는 게임 엔진인 Unity를 활용해 VR 콘텐츠를 제작할 때 적용할 수 있는 최적화 방안에 대한 자료를 수집한 후, Unity 최적화 기법을 VR 콘텐츠에 적용할 수 있는지 연구하고, Unity에서 제공하는 Memory Profiler와 VR 기기인 Meta Quest 2를 활용해 최적화 기법들의 효율성을 평가해 최적화 가이드라인을 제시한다.

Ⅱ. 문헌 고찰

1. 최적화

게임의 최적화는 게임의 흥미를 극대화하기 위한 중요한 개념으로[8], 이를 통해 다양한 플랫폼에서 사용자 몰입도를 높이는 것이 중요하다.

기존 Unity 엔진은 객체 생성 시 발생한 싱글 스레드 물리 연산으로 성능 문제를 겪었으나, 현재 Unity는 복잡한 프로그래밍 없이도 게임에 최신 멀티코어 프로세서를 완전히 활용할 수 있도록 Unity의 핵심 기반을 고성능 멀티스레드 데이터 지향 기술 스택(DOTS)으로 개편하였다[9]. DOTS(Data-Oriented Technology Stack)는 기존 객체 중심 시스템을 대체하는 데이터 중심 프로그래밍 시스템이다. 따라서 개발자는 멀티스레드(Multi-Tread) 및 메모리 최적화에 중점을 두고 소스코드를 작성할 수 있다[10].

또한 그래픽스 퍼포먼스 측면에서 최적화는 장면을 렌더링하는 과정에서 발생하는 병목현상을 식별하고 제거하는 작업이라고도 할 수 있다. Unity 3D는 이를 용이하게 파악할 수 있도록 그래픽 사용자 인터페이스(GUI) 기반의 Profiler를 제공한다[11].

2. 라이트맵을 활용한 Lighting Baking

Unity 공식 문서에 따르면 라이트맵(LightMap)은 씬의 정적 오브젝트에 대한 광원 효과가 사전에 렌더링된 텍스처다[12]. Unity는 에디터에서 Baked 된 광원 계산을 처리하고 그 결과를 디스크에 조명 데이터로 저장하는 과정을 Baked라고 한다. 따라서 라이트 베이크(Lighting Baking)는 실시간 렌더링 대신 미리 계산된 전역 조명 정보를 사용하여 조명 및 그림자를 효율적으로 처리하는 기술이다[13]. 이는 씬에 대한 조명을 미리 계산하여 텍스처로 저장하고, 실시간으로 빛과 그림자를 계산하는 부하를 줄이고 프레임 속도를 개선하여 성능을 향상한다. 하지만 Unity 3D에서 Baked 된 라이트를 사용하기 위해서는 오브젝트를 Static으로 설정해야 고정된 위치에서 조명과 그림자를 정확하게 계산할 수 있다. 따라서 Baked 된 조명은 정적 오브젝트만을 대상으로 한다는 한계가 있다.

전역 조명 기법은 광원에서 방출된 빛과 주변 물체에서 반사된 빛을 모두 표현하는 그래픽스 기술이다[14]. 전역 조명은 간접 조명을 다양하게 표현하기 위해

경로 추적, 래디오시티, 포톤 매핑 등의 기법이 사용되지만, 계산이 복잡하고 느리며, 제한된 광원을 전제로 하기 때문에 사실감이 떨어지는 단점이 있다[15].

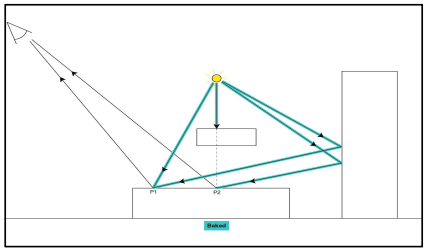

그림 1. Lighting Baking의 시각화 자료 [16]
Figure 1. Visualization of Lighting Baking

3. 오클루전 컬링을 활용한 렌더링 성능 최적화

오클루전 컬링(Occlusion Culling)은 객체의 가시성을 판단하여 렌더링 성능을 향상하는 기법으로[17] 다른 객체에 의해 가려져 카메라에 보이지 않는 객체의 렌더링을 비활성화하는 기능이다. 이는 카메라에 보이지 않는 물체들을 미리 제거하여 보이는 물체만 그래픽 하드웨어에 전달함으로써, 처리해야 할 다각형의 수를 줄여준다[18]. 3D 컴퓨터 그래픽스에서는 일반적으로 카메라에서 먼 오브젝트가 먼저 렌더링 되고, 더 가까운 오브젝트가 그 위에 차례로 덮여 그려지기 때문에 오클루전 컬링이 자동으로 발생하지 않는다[19]. 따라서 오클루전 컬링을 사용하면 드로우 콜과 가시적 지오메트리를 줄일 수 있다.

그림 2. Unity에서 Occlusion Culling을 적용한 예시 [19]
Figure 2. Example of applying Occlusion Culling in Unity

4. 오브젝트 풀링을 활용한 효율적인 자원 관리

오브젝트 풀링(Object Pooling)은 필요한 만큼의 오브젝트나 메모리를 미리 정적으로 할당하고 이후 재활용하는 기법이다[20]. 이는 Unity에서 주로 게임 오브젝트를 짧은 시간에 반복적으로 생성 및 제거할 때 메모리에 큰 부담을 줄 수 있기 때문에, 게임 씬 시작 시 필요한 오브젝트를 미리 생성해 두고 필요할 때마다 재사용하는 방식으로 활용한다[21]. 따라서 오브젝트 풀링은 메모리 할당을 더 간단히 할 수 있으며 동적 메모리 할당 오버헤드와 가비지 컬렉터를 없앨 수 있다.

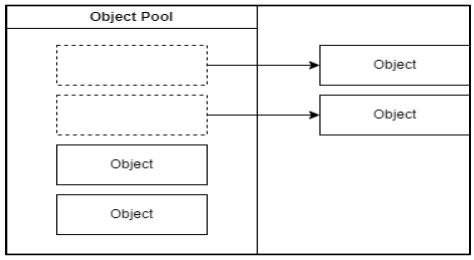

그림 3. Object Pooling의 시각화 자료
Figure 3. Visualization of Object Pooling

5. JSON을 활용한 비동기적 씬 로딩

JSON(JavaScript Object Notation)은 경량화된 데이터 형식으로, 구조화된 데이터를 간결하고 효율적으로 표현하는 포맷이다. 이러한 특성으로 인해 게임 분야에서는 플레이어 정보, 게임 세팅 등의 다양한 정보를 저장하고 전달하는 데 JSON 라이브러리를 활용한다[22]. 이는 게임을 제작할 때 씬 로딩 시 필요한 데이터를 비동기적으로 처리하여 로딩 시간을 단축하고 FPS 저하를 방지하기 위해 사용할 수 있다.

이를 Unity에서 활용하기 위해 구글 스프레드시트로 각 오브젝트의 ID, 경로, Transform 정보를 저장한 후 데이터를 JSON 형식으로 내보낸다. 내보낸 JSON 파일은 Unity의 JSON Utility 라이브러리를 통해 읽어 들여 각 행을 순차적으로 처리하며, 오브젝트의 경로와 Transform 정보를 이용해 씬 내 오브젝트를 동적으로 생성하고 배치할 수 있다.

III. 실험 설계

1. Unity 3D를 활용한 VR 콘텐츠 설계 및 제작

본 콘텐츠는 게임 엔진인 Unity 3D를 통해 우주를 배경으로 한 생존 콘텐츠인 'POS'를 제작하였다. 'POS'는 무중력 환경에 노출된 신체의 실질적인 변화를 조사하고, 스팀(STEAM)에 등록된 생존 게임의 생존 요소를 분석하여, 이를 기반으로 우주를 배경으로 한 플레

이어 캐릭터의 생존 요소로 과학적 사실을 적용할 수 있는 방안을 설계해 생존 시스템을 적용한 콘텐츠이다[23].

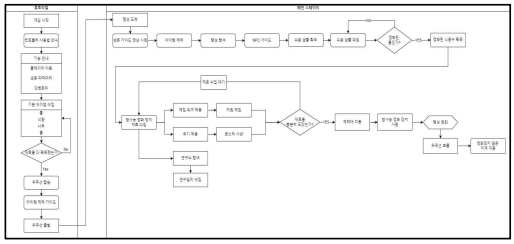

그림 4. 가상현실 우주 생존 멀티 콘텐츠 플로우 차트
Figure 4. Virtual reality space survival multi-content flow chart

본 콘텐츠는 Unity 2022.3.4.f1 버전으로, Universal Render Pipeline(URP)을 적용한 3D 프로젝트 설정으로 XR Interaction toolkit을 사용하는 시작 씬과 메인 씬으로 구성하였다. 시작 씬은 2603개의 오브젝트와 69개의 라이트를, 메인 씬은 2870개의 오브젝트와 55개의 라이트를 배치하였다. 해당 씬들은 공통으로 RealTime 모드로 설정되어 있는 라이트, 그래픽스에서 사용되는 표준 압축 형식인 ASTC를[24] 적용하였다.

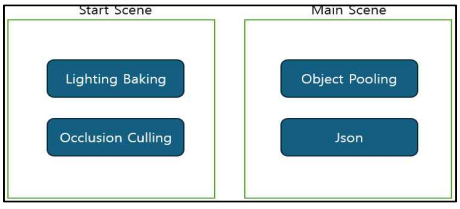

그림 5. 최적화 방안을 적용한 씬 설계
Figure 5. Scene design using optimization methods

그림 6. Unity 3D를 활용해 제작한 우주 생존 콘텐츠 게임 화면
Figure 6. In-game screen of space survival content created using Unity 3D

2. 최적화 방안의 효율성을 평가하기 위한 실험 설계

본 연구는 VR 기기인 Meta Quest 2를 활용해 실험을 진행하였다. 해당 기기는 CPU, GPU, 통합 메모리 등을 하나의 실리콘 패키지에 포함해, 계산 시 GPU와 CPU가 동일한 물리적 메모리를 공유하는 SoC(System on Chip) 형태로 성능을 표로 정리해 아래 표 1에 나타내었다[25].

표 1. Meta Quest 2 성능표 [26]
Table 1. Meta Quest 2 Performance Table

Meta Quest 2	
SoC	Snapdragon XR2 Gen1
DRAM	6GB
메모리	128GB
운영체제	Android 10
디스플레이	Dual 1832 * 1920

그 후 Unity에서 Android OS 플랫폼으로 빌드한 후, Meta Quest 2에서 실행시킨 뒤 Profiler를 활용해 CPU, 주로 객체를 화면에 그리기 위한 드로우콜[27], 애플리케이션이 사용한 총메모리양, FPS를 추출하였다.

프로젝트 실행 시 다른 요소들이 개입되는 것을 고려해 프로그램이 안정화된 101프레임 이후의 100프레임을 측정하였으며 10번의 실험을 통해 나온 결괏값의 평균을 기록해 변화율을 도출하였다.

Ⅳ. 실험 결과

첫째, 라이트 베이크는 시작 씬에 배치된 라이트가 realtime으로 동작할 때와 Baked 된 라이트맵을 적용하여 Baked로 동작할 때의 값을 비교 분석하였다. 그 결과, CPU 처리 속도에서 약 32.94%, 총 메모리 사용량에서 35.48% 감소하였고, 드로우콜에서는 약 18.71%, FPS에서 약 50%의 값이 증가한 모습을 보여주었다.

라이트 베이크를 사용하면 실시간 조명 계산을 줄이고, 사전 계산된 라이트 맵을 사용하여 조명 정보를 적용했기 때문에 CPU의 실시간 조명 계산 부담을 크게 덜어줌으로써 렌더링 효율이 높아짐에 따라 FPS도 향상되었다. 드로우콜과 메모리 용량은 늘어났다. 이는 조명 정보가 포함된 추가 텍스처를 사용하는 오브젝트가 많아지면서 증가하였다. 이 방법은 광원이 많고 그림자가 복잡한 대규모 오픈 월드 게임이나 VR로 구현된 가상 박물관 및 교육 콘텐츠에서 정적 오브젝트의

조명 효과를 미리 계산해 성능을 향상시킬 수 있다.

표 2. 'Lighting Baking' 적용 전/후를 비교
Table 2. Table comparing before and after applying 'Lighting Baking'

	Before	After	변화율 (%)
CPU (ms)	41.60	27.86	32.94↓
드로우콜 (k)	0.85	1	18.71↑
메모리 (GB)	1.55	1.88	35.48↓
FPS	24.04	36	50↑

둘째, 오클루전 컬링은 시작 씬의 오브젝트를 정적으로 배치하여 오클루전 컬링 데이터를 기반으로 렌더링을 진행했을 때와 오클루전 컬링 데이터를 사용하지 않았을 때의 값을 비교 분석하였다. 이는 CPU 처리 속도에서 약 3.04%, 총 메모리 사용량에서 약 1.29%, FPS에서 약 2.71%가 증가하였으며, 드로우콜에서는 약 10.73%가 감소하였다.

카메라에 보이지 않아 렌더링할 필요가 없는 객체들이 줄어들면서 CPU가 처리해야 할 드로우콜의 수가 감소해 FPS가 소폭 증가하였으며, 오클루전 컬링을 사용하기 위해 미리 정적으로 필요한 추가 데이터를 저장해야 하므로 메모리 사용량 또한 약간 증가하였다. 이 기술은 복잡한 도시 환경이나 밀도가 높은 실내 공간을 렌더링하는 콘텐츠, 또는 저사양 디바이스와 모바일 환경에서 배터리 소모를 줄이고 부드러운 성능을 제공하는 데 효율적이다.

표 3. 'Occlusion Culling' 적용 전/후 비교
Table 3. Table comparing before and after applying 'Occlusion Culling'

	Before	After	변화율 (%)
CPU (ms)	41.60	40.39	3.04↑
드로우콜 (k)	0.85	0.75	10.73↓
메모리 (GB)	1.55	1.57	1.29↑
FPS	24.04	24.8	2.71↑

셋째, 오브젝트 풀링은 메인 씬에서 자주 사용할 몬스터를 Instantiate와 Destroy를 순환적으로 실행하는 방식과, 오브젝트를 미리 생성하여 리스트에 저장하고 활/비활성화를 반복하는 경우를 비교하였다. 따라서 CPU 처리 속도에서 약 4.17%, 드로우콜에서 약 7.69%가 감소하였고, 총 메모리 사용량이 약 1.61%, FPS에서 약 5.88%가 증가하였다.

이는 객체의 생성과 파괴를 반복하는 빈도가 줄어들어 CPU 처리 속도와 드로우콜이 약간 감소하면서 전체적인 성능이 향상되어 FPS가 소폭 증가하였다. 하지만 오브젝트 풀링을 위해 미리 생성해 둔 객체들이 메모리에 유지되기 때문에 메모리양은 약간 증가하였다. 따라서 몬스터가 지속적으로 스폰되고 제거되는 슈팅 게임이나 타워 디펜스 게임, 그리고 다수의 플레이어가 참여하는 MMORPG에서 네트워크 지연을 최소화하고 빠른 반응성을 유지하기 위해, 오브젝트 풀링을 사용하여 몬스터나 아이템을 빠르게 생성하고 제거함으로써 성능을 향상시킬 수 있다.

표 4. 'Object Pooling' 적용 전/후 비교
Table 4. Table comparing before and after applying 'Object Pooling'

	Before	After	변화율 (%)
CPU (ms)	56.69	54.89	4.17↓
드로우콜 (k)	2.59	2.4	7.69↓
메모리 (GB)	1.05	1.07	1.61↑
FPS	16.83	18	5.88↑

마지막으로 JSON을 활용한 비동기 씬을 로딩하는 방안은 씬 전환과 동시에 맵 로드를 진행하는 방식과 맵 로드가 완료되고 씬이 전환되는 비동기 방식의 경우를 비교하였다. 따라서 해당 실험은 씬 전환이 일어난 이후부터 100프레임을 측정하였다. 그 결과 드로우콜은 거의 변화가 없으며, CPU 처리 속도가 약 10.05%, FPS에서 약 0.95%가 증가하였으며, 총 메모리 사용량은 약 1.78%가 감소하였다.

비동기 로딩 방식이 씬 전환 시의 프레임 하락으로 인한 끊김 현상을 줄일 수 있었지만, 비동기 방식으로 씬의 로드가 완료된 후, CPU가 씬 전환 이후에도 지속해서 데이터를 처리하거나 초기화하는 작업을 수행하기 때문에 추가적인 CPU 부하가 발생한 것으로 보인다.

Ⅴ. 결 론

본 연구는 VR 콘텐츠 제작 시 사용자가 프레임 드랍으로 인한 사이버 멀미를 경험하거나 HMD의 그래픽 성능 제한으로 인한 메모리 및 CPU/GPU 부하를 최소화하기 위해 Unity 3D에서 적용할 수 있는 최적화 방

안을 조사하였다. 조사한 방안으로는 '라이트 베이크', '오클루전 컬링', '오브젝트 풀링', 'JSON을 활용한 비동기적 씬 로딩'이며, 위의 기술들이 Stand Only 형식의 Meta Quest 2에서 실행했을 때의 효율성을 평가하기 위해 우주를 배경으로 한 가상현실 생존 콘텐츠를 제작하였다[23]. 이 콘텐츠를 Android OS 플랫폼으로 빌드 후 Profiler를 통해 최적화 기술을 적용하기 전과 후의 CPU 처리 속도, 총 메모리 사용량, 드로우콜, FPS를 각각 비교하여 표로 나타내었다. 그 결과 라이트 베이크와 오클루전 컬링이 CPU 처리 속도와 총 메모리 사용량이 감소시키고 FPS를 향상시켰다. 오브젝트풀링은 CPU 처리 속도와 드로우콜을 줄이며 FPS를 상승시켰으며, JSON을 활용한 비동기 씬 로딩은 CPU의 부하를 줄이고 FPS를 약간 향상시켰다. 이는 대규모 콘텐츠 환경에서 성능을 향상시키며, 특히 광원 복잡도가 높고 그림자가 복잡한 환경에서는 라이트 베이크와 오클루전 컬링이 효과적이며, 네트워크 지연을 최소화하고 반응성을 높이는데 오브젝트 풀링이 유용하다.

본 연구의 결과는 Unity 3D를 사용하여 VR 콘텐츠를 개발할 경우 기본적이고 실질적인 최적화 기법을 다루고 있어 인디 게임을 개발하는 사람이나 Unity 엔진이 익숙하지 않은 사람들에게 VR 콘텐츠를 개발할 때 겪을 수 있는 성능 문제를 해결하는 데 유용한 가이드라인을 제공할 것으로 기대한다.

그러나 VR과 가상현실 콘텐츠를 다룬 학술적 자료는 많았지만, 이를 Unity 3D로 제작할 때 필수적으로 진행해야 하는 최적화 기술에 대한 문헌 자료가 많지 않기 때문에 본 연구에서 수행한 최적화 방안에 대해 조사한 연구 자료가 한정적이라 다양한 최적화 방안에 대해서 충분히 다루지 못하였다는 한계가 있을 수 있다. 또한 최적화를 진행하기 위해서는 한가지 요소뿐만 아니라 시스템의 전체적인 성능과 효율성을 고려해 다른 구성 요소와의 조화가 중요한데, Unity 3D에서 제작된 콘텐츠를 Meta Quest 2에서 활용할 때 Profiler에서 GPU를 확인할 수 없다. 따라서 본 연구에서 실시한 실험들은 Unity에서 제공하는 Profiler에서 Meta Quest 2를 활용할 때 호환성 문제로 인해 GPU 측면에서 정확하게 측정할 수 없었다는 한계가 있었다.

향후 우주에서 발생하는 실질적인 신체 변화와 다양한 최적화 기법을 적용한 VR 멀티 생존 콘텐츠를 설계하고자 한다.

표 5. 'JSON을 활용한 비동기 씬 로딩' 적용 전/후 비
Table 5. Table comparing before and after applying 'Asynchronous scene loading using JSON'

	Before	After	변화율 (%)
CPU (ms)	51.9	54.58	10.05↑
드로우콜 (k)	2.92	2.92	0
메모리 (GB)	1.07	1.05	1.78↓
FPS	17.2	19.13	0.95↑

References

[1] M.J. Yoo, "Practical Training Experiences of HMD-based VR Foley Catheterization Education Content in Nursing Students," *Journal of Learner-Centered Curriculum and Instruction,* Vol. 24, No. 2 pp. 541-553, January 2023. https://doi.org/10.22251/jlcci.2024.24.2.541

[2] C. Lim, "Interactive 3D VR Content Design for Close Encounter of Social Issue," *The International Journal of Advanced Culture Technology*, Vol. 11, No.1, pp. 270-27, January 2023. https://doi.org/10.17703/IJACT.2023.11.1.270

[3] J.Y. Jung, S.C. Kim, and T. Woo, "Research on VR-based educational content development for Preschoolers," *Journal of Digital Contents Society*, Vol. 23, No. 11, pp. 2117-2126, November 2022. http://dx.doi.org/10.9728/dcs.2022.23.11.2117

[4] K.S. Park, "A Case Study of Fashion Illustration Using VR Technology-Focusing on iPad use case and comparative analysis," *Journal of the Convergence on Culture Technology (JCCT),* Vol. 9, No. 1, pp. 763-770, January 2023. http://dx.doi.org/10.17703/JCCT.2023.9.1.763

[5] S.H. Kim, C. H. Kim, and S. E. Lee, "A Study on the Spiritual & Mental Force Enhancement Using HMD - Focusing on the virtual reality education contents -," *Journal of Spiritual & Mental Force Enhancement*, No. 71 pp. 71-106, November 2022.

[6] M.K. Hwang, and J.Y. Kim, "A Study on the Development of Healing VR Content Based on Horticulture," *Journal of the Convergence on Culture Technology (JCCT)*, Vol. 9, No. 4, pp. 681-686, July 2023. http://dx.doi.org/10.17703/JCCT.2023.9.4.681

[7] D.J. Kim, "A Study on Building an LED Control Web Server using an HMD-based Controller," *Journal of Digital Art Engineering & Multimedia,* Vol. 10, No. 1 pp. 107-115, March 2023. http://dx.doi.org/10.29056/jdaem.2023.03.10

[8] Y.B. Kim, K.H. Chung, K.S. Ahn, and J.J. Kim, “The Game Optimization using the Action Patterns of Monster in Mobile Arcade Game,” *Journal of Internet Computing and Services (JICS),* Vol. 8, No. 6, pp. 103-114, December 2007.

[9] H.C. Song, “A Study on Optimized Multi-Thread Porgramming - Information and The Use of Unitiy DOTS,” *Journal of Digital Contents Society(JDCS),* Vol. 22, No. 10, pp. 1715-1719, October 2021. http://dx.doi.org/10.9728/dcs.2021.22.11.1715

[10] Y.C. Choi, B.D. Lee, C.H. Lee, C.G. Song, J. Le e, and S.J. Kim, “Performance and Efficiency Analysis of Unity DOTS,” HCI KOREA, pp. 1236-1241, February 2023.

[11] S.K. Kim, J.H. Kang, K.S. Song, H.B. Lee, and S.O. An, “Development of Mobile Arcade Game using Unity 3D Engine,” *Journal of Korean Institute of Information Technology (JKIIT),* Vol. 8, No. 2 pp. 9-16, April 2013.

[12] https://docs.unity3d.com/kr/2023.2/Manual/Glossary.html#Lightmap

[13] https://docs.unity3d.com/kr/2023.2/Manual/LightMode-Baked.html

[14] H.X. Zheng, K.S. Oh, and D.W. Paik, “A Scalable Global Illumination Algorithm for Animated Graphics,” *Journal of Korea Game Society JKGS,* Vol. 16, No. 2, pp. 111-117, April 2016. http://dx.doi.org/10.7583/JKGS.2016.16.2.111

[15] J.Y. Lee, and C.J. Im, “Realtime Global Illumination Rendering based on New Basis Function,” *Journal of The Korean Society for Computer Game,* Vol. 3, No. 21, pp. 97-105, June 2010.

[16] https://docs.unity3d.com/kr/2018.4/Manual/LightMode-Baked.html

[17] J.W. Kang, and S.K. Lee, “GPU-Based High-Precision Adaptive Vertex Depth Rendering,” *Journal of KIISE,* Vol. 48, No. 7, pp. 756-763, July 2021. https://doi.org/10.5626/JOK.2021.48.7.756

[18] S.J. Jung, K.Y. Lee, H.S. Choi, W.J. Sung, and D.Y. Cho, “A Study on the Efficient Occlusion Culling Using Z-Buffer and Simplified Model,” *Korean Journal of Computational Design and Engineering*, Vol. 8, No. 2, pp. 65-74, June 2003.

[19] https://docs.unity3d.com/kr/530/Manual/OcclusionCulling.html

[20] H.Y. Kim, D.H. Ham, and M.S. Kim, “A Study of Object Pooling Scheme for Efficient Online Gaming Server,” *Journal of Korea Game Society*, Vol. 9, No. 6 pp. 163-170, December 2009.

[21] W.G. Iim, B.C. Lee, and S.K. Kim, and S.A, “Development of 3D Defense Space Game using Oculus,” *Journal of The Korea Society of Computer and Information*, Vol. 24, No. 8 pp. 45-50, August 2019. https://doi.org/10.9708/jksci.2019.24.08.045

[22] T.H. Kim, and B.P. Kyung, “Performance Comparison of JSON Libraries for Game Development using Unity Engine,” *Journal of Digital Contents Society*, Vol. 25, No. 3 pp. 771-779, March 2024. http://dx.doi.org/10.9728/dcs.2024.25.3.771

[23] S.J. Han, M.J. Jang, and J.Y. Kim, “Proposal Research for Character's Physical Change System in Space Survival Game through Literature Review,” *Journal of The Institute of Internet, Broadcasting and Communication (IIBC),* Vol. 24, No. 3, pp. 1-7, June 2024. https://doi.org/10.7236/JIIBC.2024.24.3.1

[24] J.H. Nah, “ASTC Block-Size Determination Method based on PSNR Values,” *Journal of the Korea Computer Graphics Society*, vol. 28, no. 2, pp. 21-28, June 2022. https://doi.org/10.15701/kcgs.2022.28.2.21

[25] O.K. Kwon, and G.B. Gu, “A Performance Study on CPU-GPU Data Transfers of Unified Memory Device,” *Journal of KIPS Transactions on Computer and Communication Systems (KTCCS), Vol. 11, No. 5, pp. 133-138, May 2022.* https://doi.org/10.3745/KTCCS.2022.11.5.133

[26] https://www.meta.com/kr/quest/products/quest-2

[27] S.J. Park, M.G. Kim, H.Y. Kim, and D.W. Seo, “Mobile Augmented Reality based CFD Simuation Post-Processor,” *Journal of the Korea Academia-Industrial cooperation Society, vol. 20, no. 4, pp. 523-533, April 2019.* https://doi.org/10.5762/KAIS.2019.20.4.523