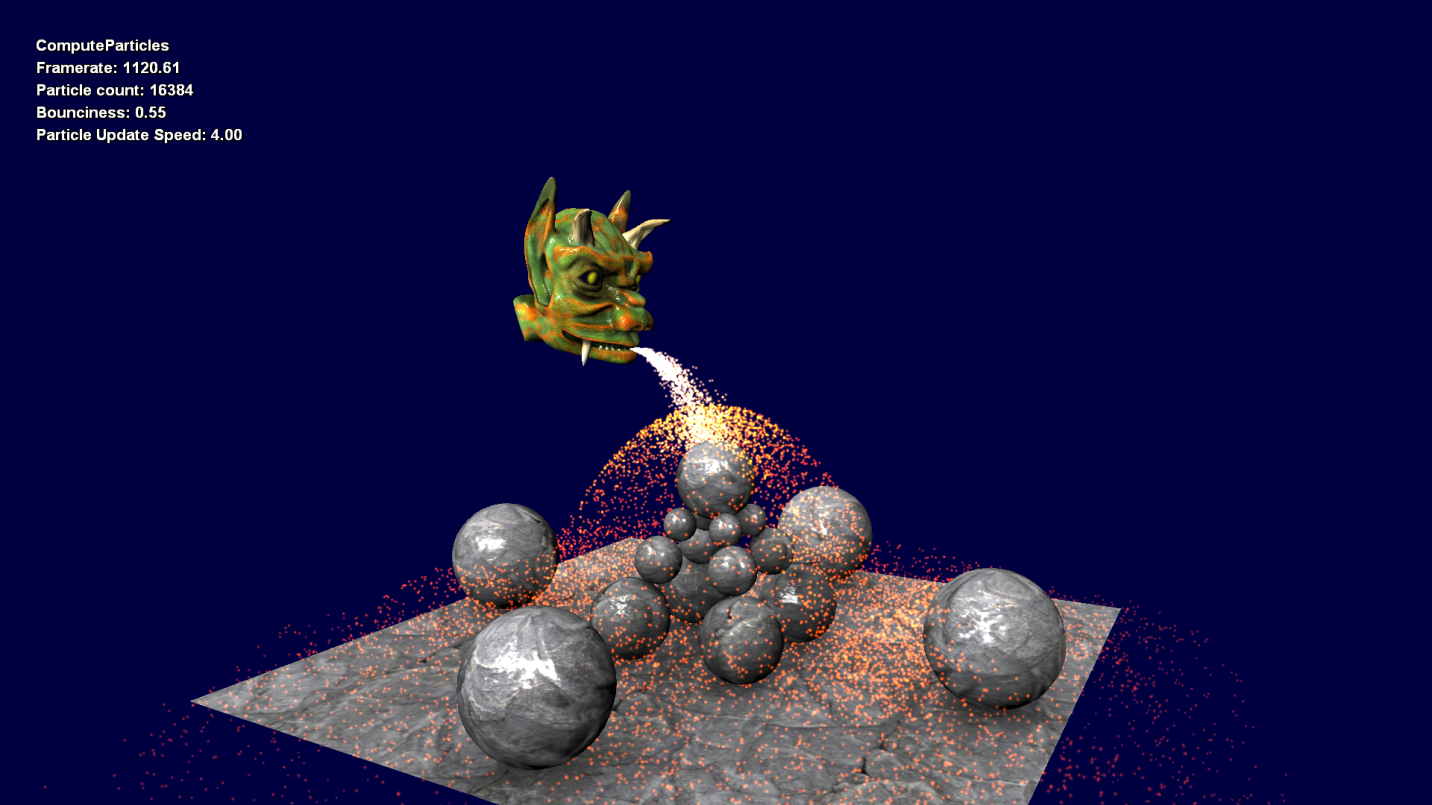


파티클 계산

*이 샘플은 Microsoft 게임 개발 키트 미리 보기(2019년 11월)와 호환됩니다.*

# 설명

이 샘플에서는 계산 셰이더 및 추가 버퍼를 사용하여 기본 파티클 시뮬레이션을 수행하고 뛰어난 성능으로 적절한 수의 파티클을 렌더링하는 방법을 보여 줍니다.



# 샘플 빌드하기

Xbox One 개발 키트를 사용하는 경우 활성 솔루션 플랫폼을 Gaming.Xbox.XboxOne.x64로 설정하세요.

Project Scarlett을 사용하는 경우 활성 솔루션 플랫폼을 Gaming.Xbox.Scarlett.x64로 설정하세요.

*자세한 내용은 GDK 문서에서* 샘플 실행하기*를 참조하세요.*

# 샘플 사용하기

이 샘플은 다음 컨트롤을 사용합니다.

|  |  |
| --- | --- |
| 작업 | 게임패드 |
| 샘플을 종료합니다. | 선택 |
| 파티클 바운스 증가/감소 | 오른쪽/왼쪽 트리거 |
| 카메라 회전 | 왼쪽 및 오른쪽 스틱 |
| 파티클 송신기 이동 | 왼쪽 및 오른쪽 스틱 + 오른쪽 숄더 |
| 파티클 렌더링 토글 | A 단추 |
| 파티클 업데이트 토글 | B 단추 |
| 파티클 수 증가/감소 | D-패드 위/아래로 |

# 구현 참고 사항

이 샘플에서는 D3D11 및 계산 셰이더에서 사용할 수 있는 보다 창의적이고 흥미로운 몇 가지 방법을 보여 줍니다. 이 샘플에는 세 가지 흥미로운 부분이 있습니다. 처음 두 가지는 파티클을 업데이트하고 선별하는 방법과 관련이 있으며, 세 번째는 일반 렌더링 파이프라인에서 계산 셰이더 스테이지의 결과를 사용하는 방법과 관련이 있습니다.

1. **파티클 시뮬레이션**

AdvanceParticlesCS 컴퓨터 셰이더 시뮬레이션 단계는 두 가지 주요 단계를 포함합니다. 첫째, 파티클 위치, 속도 및 사용 기간을 UAV 버퍼에서 읽어온 후 세계 좌표에서 시뮬레이트합니다. 그런 다음, 간소화된 세계 기하 도형(지평면과 구)에 대한 충돌이 무작위 접근 방식을 사용하여 계산됩니다. 새 위치, 속도 및 사용 기간이 이전에 읽은 데이터를 덮어쓰면서 동일한 UAV 버퍼로 다시 쓰여집니다.

1. **파티클 선별 및 쓰기**

각 파티클은 뷰 절두체 내에서 보이는지 여부를 고려하기 위해 간단한 평면 선별 알고리즘이 적용됩니다. 파티클이 보이면 해당 위치가 렌더링을 위해 추가 버퍼에 추가됩니다.

ID3D12Device::CreateUnorderedAccessView(…) API에서 버퍼 리소스에 대해 UAV(순서가 지정되지 않은 액세스 뷰)를 만들고 두 번째 리소스를 'PCounterResource' 매개 변수로 지정하여 추가 버퍼를 사용할 수 있습니다. 추가 버퍼의 현재 개수(1개의 32비트 부호 없는 정수)를 저장하려면 카운터 리소스가 4바이트 이상이어야 합니다. 카운터 리소스에 대해 ID3D12GraphicsCommandList::ClearUnorderedAccessViewUint(…) API로 각 프레임 개수를 지울 수 있도록 하는 UAV도 생성됩니다. 일단 만든 후에는 버퍼를 AppendStructuredBuffer<…>로 선언된 UAV 셰이더 슬롯에 간단히 바인딩합니다.

AdvanceParticlesCS 계산 셰이더는 활성 파티클 인스턴스를 시뮬레이트하고 추가 버퍼에 추가합니다. 작업이 완료되면 ID3D12GraphicsCommandList::CopyBufferRegion(…) API를 사용하여 카운터 리소스의 파티클 수를 ID3D12GraphicsCommandList::ExecuteIndirect(…) API에 대한 입력으로 사용할 수 있는 간접 인수 버퍼 리소스로 복사합니다. 이렇게 하면 AdvanceParticlesCS의 절두체 선별 테스트로 구분된 보이는 파티클만 그릴 수 있습니다.

1. **렌더링**

ID3D12GraphicsCommandList::ExecuteIndirect(…) API는 파티클 렌더링을 디스패치하는 데 사용됩니다. ID3D12Device::CreateCommandSignature(…)를 사용하여 만든 명령 시그니처는 ExecuteIndirect가 디스패치할 명령 유형을 지정하는 데 필요합니다. 명령 유형은 간접 인수 버퍼의 내용이 해석되는 방식을 결정합니다. 이 경우의 명령 유형은 ‘Draw’로, ID3D12GraphicsCommandList::DrawInstanced(…) API - 32비트 부호 없는 정수와 상호 연관됩니다. 여기서는 VertexCountPerInstance를 4로 하드 코드하고 파티클 개수를 각 프레임의 InstanceCount 위치에 복사합니다.

정점 특성은 정점 셰이더의 상수 조회 테이블로 하드 코드됩니다. 정점 ID(SV\_VertexID)는 각 정점 특성에 액세스하기 위해 이 조회 테이블에 인덱싱하는 데 사용됩니다. 인스턴스 ID(SV\_InstanceID)는 각 인스턴스의 속성에 액세스하기 위해 파티클 인스턴스 버퍼로 인덱싱하는 데 사용됩니다.

# 업데이트 기록

2019년 3월 – 기존 Xbox 샘플 프레임워크에서 새 템플릿으로 포팅.

# 개인정보처리방침

샘플을 컴파일하고 실행할 때 샘플의 사용을 추적하는 데 도움이 되도록 샘플 실행 파일의 파일 이름이 Microsoft에 전송됩니다. 이 데이터 수집을 옵트아웃하려면 Main.cpp에서 "샘플 사용 원격 분석"이라고 레이블이 지정된 코드 블록을 제거할 수 있습니다.

Microsoft의 일반 개인정보취급방침에 대한 자세한 내용은 [Microsoft 개인정보처리방침](https://privacy.microsoft.com/en-us/privacystatement/)을 참조하세요.