기말 문제 정리

==========

상수 버퍼에 대해 설명

상수 버퍼란 여러 개의 쉐이더 상수들의 그룹을 동시에 갱신할 수 있는 버퍼입니다. 상수 변수에 최적화 되어있고 CPU 가 자주 갱신을 합니다. 이때 쉐이더 상수를 그룹으로 묶어 동시에 갱신을 하는데, 이 그룹을 갱신에 빈도에 따라 묶어야 성능이 최적화됩니다. 파이프라인 마다 최대 14 개의 상수 버퍼가 연결 가능하고, 파이프라인 연결을 위해 뷰가 필요합니다. 읽기 전용입니다. 크기 제한도 있습니다.

텍스처 버퍼: 인덱스화 된 데이터에 적합. 파이프라인 마다 최대 128 개의 텍스처 버퍼 연결 가능. 파이프 라인에 연결을 위해 뷰가 필요하고 텍스처 슬롯에 연결되어야 함.

==========

directx 3d 디바이스의 상태 설정에 대해 설명

==========

hlsl 의 structure 버퍼에 대해 설명

쉐이더 모델 5. 5.1 에서 사용한다. SRV 로 GPU 에 바인딩되어 셰이더에서 사용하는 유형의 데이터. 배열처럼 인덱싱 가능하고 병렬로 작업 가능하다. 많은 양의 데이터를 효율적으로 처리 가능해서 그래픽 및 계산 작업의 성능이 개선된다

==========

루트파라미터에 대해(함수)

그래픽 파이프라인에서 사용되는 정보를 루트 시그니처에 바인딩 할 때 사용한다.

루트 상수: 작은 상수 데이터 세트. 빠른 접근이 가능하다는 장점

루트 서술자: CBV, SRV, UAV 와 같은 서술자를 직접 루트 시그니처에 바인딩

서술자 테이블: 여러 서술자를 그룹화하여 루트 시그니처에 바인딩 하는 구조. 여러 그래픽 리소 스를 동시에 전달 가능

CBV: 상수 버퍼 뷰. 상수 버퍼를 루트 시그니처와 바인딩 할 때 사용. 상수 버퍼를 GPU 에서 쉽 게 읽어 올 수 있다.

SRV: 쉐이더 리소스 뷰. 그래픽 리소스를 루트 시그니처와 바인딩 할 때 사용. 그래픽 리소스를 쉽게 읽을 수 있다.

UAV: 언오더드 억세스 뷰. 그래픽 리소스를 루트 시그니처와 바인딩 할 때 사용. 쉐이더에서 그 래픽 리소스에 쓰기 연산 가능

관련 함수는 다음과 같습니다.

먼저 루트 시그니처 생성을 하는 CreateRootSignature 함수

PSO 를 생성하는 CreateGraphicsPipelineState, CreateComputePipelineState 함수

이후 PSO 에 루트 시그니처 바인딩

루트 파라미터를 루트 시그니처에 바인딩 하는 SetDescriptorHeaps, SetGraphicsRootDescriptorTable, SetGraphicsRootConstantBufferView 함수

==========

hlsl 파이프라인에서 입력조립기에서 정점데이터를 어떻게 읽어들이는지 설명

버텍스 버퍼 단위로 읽어들입니다. 버텍스 버퍼란 정점 데이터를 버텍스 단위로 묶어서 저장한 메모리 공간입니다.

정점 데이터는 일정한 데이터 구조와 순서로 정의되어 있으며 이 구조를 정의하기 위해서는 입력 레이아웃 체계가 필요합니다.

입력 레이아웃 정보를 기반으로 입력 조립기에서 정점 데이터를 읽고, 정점 쉐이더에서 사용할 정점 데이터를 생성합니다.

입력 조립기에서는 레이아웃과 버텍스 버퍼, 인덱스 버퍼를 사용하여 정점 데이터를 읽고, 인덱스 에 따라 해당 위치의 정점 데이터를 추출하여 출력합니다. 이후 해당 정점 데이터가 정점 쉐이더 로 전달되며, 그래픽 처리가 이어집니다.

입력 레이아웃(Input Layout) 개체 생성: 입력 레이아웃은 정점당 구조(요소)의 크기와 타입, 그리 고 각 버텍스 요소가 어느 위치에 있는지 등을 기술합니다.

버텍스 버퍼(Vertex Buffer) 생성: 버텍스 버퍼는 정점 데이터가 저장된 메모리 공간을 의미합니다. 입력 조립기에서는 이 버텍스 버퍼에서 데이터를 읽어들입니다.

인덱스 버퍼(Index Buffer) 생성(Optional): 인덱스 버퍼를 사용하여 입력 소스로부터 올바른 인덱 스를 추출하여 적절한 정점 데이터와 연관시킵니다. 인덱스 버퍼를 사용하지 않는 경우는 일련의 범위 정보를 이용해 정점 데이터를 사용합니다.

정점 데이터 추출: 입력 조립기는 입력 레이아웃과 버텍스 버퍼, 인덱스 버퍼를 사용하여 정점 데 이터를 추출합니다. 이때 버텍스 버퍼의 레이아웃 정보를 기반으로 각 버텍스 요소의 구조와 순 서를 확인합니다.

정점 데이터 전달: 정점 데이터는 정점 쉐이더(Vertex Shader)로 전달되며, 그래픽 처리가 이어집 니다.

==========

쉐이더 리소스 종류와 특징 차이 설명

읽기 전용: 정점 버퍼, 인덱스 버퍼, 상수버퍼, 버퍼, 텍스처 읽기/쓰기: 무순서 접근 뷰(UAV)

정점/인덱스 버퍼: 쉐이더 직접 사용 불가. 뷰가 필요. 서술자 힙은 필요 없다

상수 버퍼: 연결된 쉐이더 단계에서 전역변수처럼 사용. 구조체와 유사하고 뷰가 필요하다. 크기 제한 있음

버퍼: 배열, 인덱스로 사용. 구조체 배열, 크기제한 없음. 뷰가 필요. 모든 쉐이더 단계에서 사용하고 여러 쉐이더 단계에 연결함. 뷰가 UAV 면 쓰기 가능.

텍스처: 모든 쉐이더 단계에서 사용. 뷰가 필요. 샘플러 객체를 통하여 사용 가능 쉐이더

리소스 뷰: 모든 단계에 연결. 읽기 전용일단 여러 쉐이더 단계에 연결 가능 무순서 접근 뷰:

픽셀 쉐이더와 계산 쉐이더에 연결. 쓰기 가능. 하나의 쉐이더에만 연결

==========

hlsl 에서 벡터 데이터 관련된거 다 얘기해

벡터는 4 개의 원소까지 포함. 모든 원소는 같은 스칼라 자료형이다. 자료형 다음의 정수가 원소 의 개수를 나타낸다. 벡터 자료형의 초기화는 배열, 생성자와 유사하다. 스칼라를 벡터에 대입하 면 스칼라를 벡터로 변환한다.

==========

루트파라미터를 디스크립터 힙에 넣어서 사용하고싶은데 어떤 함수 사용해서 어케 하냐

CreateRootSignature 함수를 통해 루트 시그너처를 생성하고 구조체를 통해 루트 파라미터를 정 의한다

PSO 를 생성하여 루트 시그니처를 바인딩 하는 CreateGraphicsPipelineState, CreateComputePipelineState 함수

뷰 생성 함수를 사용하여 자원을 루트 시그너처에 바인딩 한다

SetGraphicsRootDescriptorTable 함수를 사용하여 디스크립터 힙을 루트 시그니처에 바인딩 하여 자원을 쉐이더에서 사용할 수 있게 한다

==========

리소스 스텐실에 대해 설명하시오

리소스 스텐실이란 깊이 버퍼와 사용되는 버퍼입니다. 스텐실 버퍼라고도 불리며 씬 구성 요소 렌더링 과정에서 조건에 따라 렌더링을 허용하거나 차단하는데 사용되는 정수값들을 저장하는데 사용됩니다. 스텐실 버퍼를 사용하면 텍스처, 섀도잉 등과 같은 시각 효과를 구현 할 수 있습니다. 또한 프레임 렌더링 성능을 향상시킬 수도 있습니다

==========

조명 종류

점광원 (Point Light):

점광원은 3D 공간에서 특정 점과 관련된 빛을 나타냅니다. 이 조명 유형의 주요 특징은 빛이 모 든 방향으로 동일하게 발산하여 근거리와 원거리에 대한 조명 영향이 서로 다르게 나타난다는 것 입니다. 점광원은 현실 세계의 전구, 양초 및 기타 작은 광원과 유사한 시각 효과를 생성할 수 있 습니다.

방향광 (Directional Light):

방향광은 무한한 거리에서 특정 방향으로 발산하는 빛을 나타냅니다. 이는 빛의 소스 위치가 고 려되지 않으므로 주로 태양 또는 원거리 광원에 대한 시각 효과를 생성하는 데 사용됩니다. 방향 광은 그림자를 던지기 위한 알고리즘과 같은 추가적인 효과를 연출할 수 있기 때문에 복잡한 전 역 조명(Global Illumination)과 함께 사용될 수 있습니다. 스포트라이트 (Spot Light):

스포트라이트는 3D 공간의 특정 점에서 발산되어 원뿔형태로 제한된 영역에 빛을 비추는 빛을 나타냅니다. 스포트라이트는 각도와 범위가 정의되어 있어 특정 부분에만 빛을 비추는 시각 효과 를 생성할 수 있으며, 무대 조명 또는 특정 위치의 그림자를 강조하는 데 사용됩니다.

==========

HLSL 상수 표현 방법

==========

HLSL 최적화 하는법

내장 함수 사용

적합한 자료형 사용

자료형 변환 줄이기

정수 자료형 사용에 주의

스칼라 상수 팩킹

불필요한 연산 줄이기

==========

절두체 컬링

투영변환 전에 정점이 카메라에 보이는가를 판단해서 보이는 정점만 렌더링 할 수 있게 하는 것

바운딩 박스의 근접점이 절두체 안에 포함되니?

==========

재질에 따라 반사가 달라진다.

발광 조명: 물체의 표면이 스스로 빛을 발산

주변 조명: 물체의 표면이 빛을 반사. 모든 표면이 같은 양의 빛을 반사하여 균일한 조명이 됨

==========

직접 조명

확산 조명: 거칠고 매끄럽지 않은 표면에서 일어남. 모든 방향으로 균일하게 빛 반사.

스펙큘러 조명: 매끄러운 표면에서 일어남. 빛의 방향은 입사광에 따라 달라짐. 강한 반사가 생긴다. 카메라 위치에 따라 다르게 일어남.

조명 계산식 = 주변 광원 색상 + 확산 광원 색상 + 스펙큘러 색상 + 점 자체 발산 색상

==========

리소스 state

GPU 에서 사용되는 그래픽 리소스의 현재 상태를 말합니다.

리소스 사용 방법과 같은 정보를 제공합니다. 리소스 상태 전환을 최적화하는 것은 directX 12 의 성능 향상에 도움이 된다

==========

다중 메쉬

여러 개의 메쉬들로 모델 표현하기. 메쉬들을 프레임 계층 구조로 표현한다.

참조 프레임: 물리학/수학에서 방향을 가진 점을 표현하기 이해 사용

계층 구조

부모-자식 관계, 트리구조, 노드는 행렬, 메쉬, 자식에 대한 포인터들의 집합. 각 노드는 하나의 참조 프레임이다.

각 프레임의 변환 행렬에 의해 표현된 관계는 공간적인 계층 구조를 나타낸다.