3D 플랫폼의 이해

3D 세계의 구성

2D의 세계의 경우와 달리 3D 세계는 기술적 이슈 및 실작업자의 역량 등의 영향을 매우 많이 받기 때문에 그래픽, 프로그램 직군이 주도해서 설계

Technical Director, Engine Programmer, Shader Programmer, Art Director, Technical Artist 등의 직군이 담당

3D 세계 구성의 포인트

영화, 애니메이션 등 3D 영상 작업과 달리 프레임 레이트를 유지해 주면서 실시간으로 렌더링해야 하기 때문에 아래의 2가지 요소가 매우 중요하다

- 리소스 최적화 (폴리곤, 텍스쳐 등)

- 계산량 최적화 (시점, 광원 등)

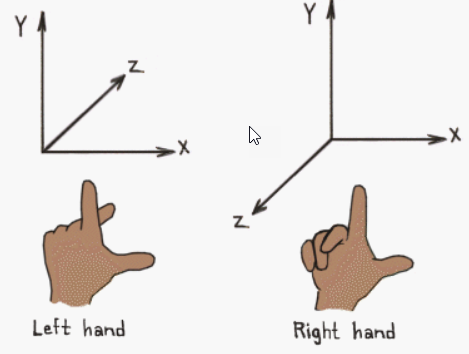
기본적인 3D 그래픽 제작 순서

1. 3D 오브젝트 제작
2. 텍스처, 빛, 그림자 입히기
3. 애니메이션 데이터 제작
4. 카메라 설정
5. 게임 데이터로 변환

3D 세계의 좌표

왼손 좌표계: DirectX, Unity, Unreal에서 사용

오른손 좌표계: 3DS Max, OpenGL에서 사용



3D 월드 구성 요소

1. 모델링
2. 조명(빛)
3. 텍스쳐
4. 빛의 처리 방법(정반사광, 난반사광)

카메라

유저의 눈(시점)이 되는 매개체

카메라의 위치, 회전 각도

화각 = FOV(Field of View)

카메라를 통해서 배경을 표현하는 각도

인간의 화각은 약 50도

인간의 화각보다 좁으면 협소하고 불안한 느낌 🡪 주로 공포 게임에 적용

인간의 화각보다 넓으면 정보가 많아 산만한 느낌

실제 개인의 화각과 게임상의 화각 차이 🡪 FPS 플레이 울렁증 원인

투영

직교 투영(Orthographic Projection)

- 물체가 뒤에 있어도 크기가 변하지 않음

원근 투영(Perspective Projection)

- 앞에 있는 물체는 크게, 뒤에 있는 물체는 작게 표현

3D 모델의 구성

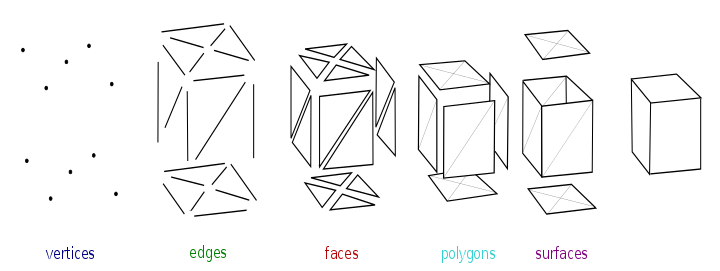
버텍스(Vertex): 공간상에서 위치를 나타내는 것

엣지(Edge): 정점과 정점을 연결하는 선

폴리곤(Polygon): 3개의 정점으로 만들어진 삼각형

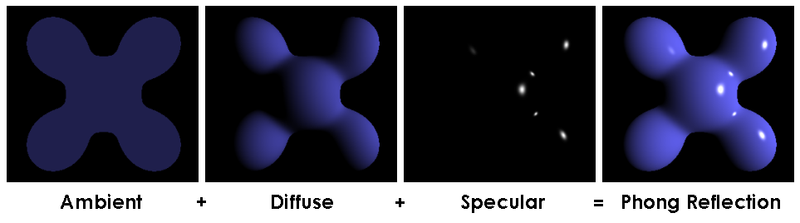
페이스(Face): 2개 이상의 연결된 면

메시(Mesh): 폴리곤이 모여 만든 하나의 3차원 물체



재질의 요소

1. 재질: 메시의 표면 상태를 빛으로 표현하는 것
2. 앰비언트(Ambient), 자연광: 똑 같은 양으로 모든 면에서 나오는 빛
3. 디퓨즈(Diffuse), 난반사광, 확산광: 모든 표면(점)에 균일하게 비춰지는 빛
4. 스펙큘러(Specular), 정반사광: 특정한 방향으로만 반사하는 빛
5. 노멀 벡터(Normal Vector): 물체 표면의 수직 방향 벡터



광원의 형태

Ambient Light: 주변 광원

3차원 공간에서 메시의 배치나 위치와는 상관없이 똑 같은 양으로 모든 곳을 비추는 빛의 강도

모든 물체의 모든 방향에서 일률적으로 적용

게임 전체의 색감과 명도를 조절

주광원에 의한 너무 강한 화면 대비를 적절 비율로 보정

Directional Light: 방향성 광원

태양과 같이 하나의 방향으로 비춰지는 빛, 방향성을 가진 빛

주 광원으로 사용되는 경우가 많음

오브젝트의 그림자를 결정

실내보다는 실외 환경에 적극적으로 활용(실내 사용 시 명암 대비가 너무 커져 공간 구분이 어려워질 수 있기 때문)

Point Light: 점 광원

전구와 같이 한 점에서 주위로 퍼져 나가는 빛

전체적인 공간의 구분을 도와주기 때문에 주로 실내에서 사용

Spot Light: 점적 광원

한정된 위치만을 비춤(예시: 손전등, 무대 조명 등)

텍스쳐(Texture)

3D Mesh에 2D 이미지를 입힘

- 3D 모델에 다양한 방식으로 텍스처를 입힘으로써(매핑:mapping) 게임의 퀄리티를 높이고 실시간 렌더링의 부하를 감소

메테리얼(Material)

메테리얼: 오브젝트가 어떤 재질로 이루어 졌는가 정의하는 것(빛, 색상)

디퓨즈 맵: 기본 음영, 색상 등을 표현하는 텍스처 (명암처리를 전혀 하지 않은 상태)

노말 맵: 메시의 노멀 벡터가 어떤 방향을 나타내는 지 저장하는 맵

- 모델의 라이팅 방향을 속이는 방식으로 적은 폴리곤으로 하이 폴리곤과 같은 비주얼을 제공

패럴랙스 맵: 카메라가 바라보는 방향에 따라 깊이감을 추가하는 방법

스펙큘러 맵: 반사광을 조절하는 텍스처, 빛의 양을 조절하는 텍스처

데칼 맵: 미끄러진 자국이나 총알 자국과 같이 오브젝트 표면에 씌워진 데칼

**디퓨즈 맵 + 노멀 맵 + 스펙큘러 = 맵 물체의 각종 재질**

- 즉, 한 개의 텍스처 = 한 개의 3D 모델이 아닌 멀티 텍스처를 이용해 다양한 기능 구현 가능

알파 매핑: 알파 레이어를 이용해 멀티 텍스처를 적용하는 기법

- 창문 등 투명한 재질의 모델이나 철조망, 풀 등 반대편이 보이는 형태의 모델, 지형의 자연스러운 전환 등에 사용 (풀, 나뭇잎 표현에도 많이 사용), 지형, 지물 연출에도 사용

알파 값 = 이 색이 얼마나 투명한 가에 대한 값

라이트 매핑: 미리 조명, 그림자를 적용하여 계산을 줄이는 방식 – 고정되어 빛이 주어지는 상황에 움직이지 않는 오브젝트에 그림자를 줄 때 주로 사용

- 실내의 빛 배치, 외부 지형의 그림자 등에 사용

리플랙션 매핑: 특정 부분이 반사된 형태의 텍스처를 적용하여 계산의 부하를 줄이는 매핑

빌보드: 보이는 카메라 위치에 상관없이 정해진 방향을 보여주는 오브젝트(카메라가 움직이는 대로 따라간다)

- 시각적 대비 리소스 소모가 큰 오브젝트의 표현(풀, 나무 등)이나 이펙트 연출 등에서 사용

셰이딩(Shading)

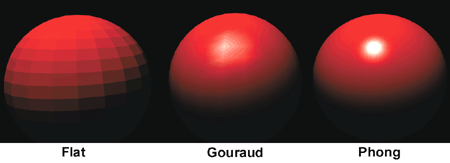
빛의 각도와 광원의 거리를 고려하여 색을 변화시켜 음영을 주는 것

플랫 셰이딩: 하나의 면 = 하나의 음영 관계

고라우드 셰이딩: 모서리 별로 다른 음영을 중간 보정

퐁 셰이딩: 각 점 별로(Pixel Shade) 음영 계산

툰 셰이딩: 음영을 단순화하여 표현하고 실루엣의 아웃라인을 생성(마치 만화처럼)



셰이더(Shader)

셰이더: 3D를 화면에 출력할 픽셀의 위치와 색상으로 변환하는 과정

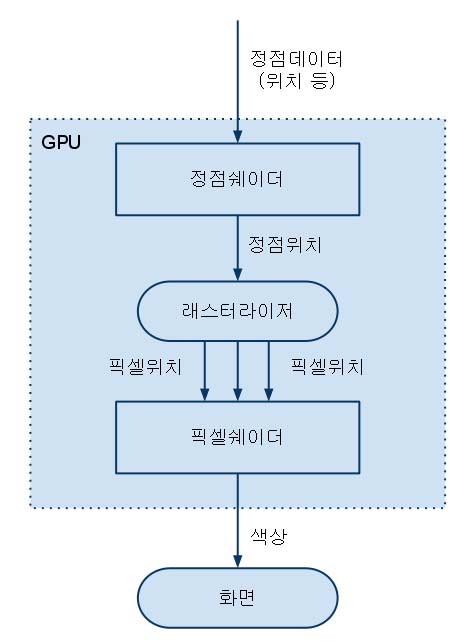
버텍스 셰이더(Vertex Shader)

- 정점(Vertex)의 위치(3D 좌표)를 화면 좌표(2D 좌표)로 변환하는 것 (3D 좌표의 화면을 어떻게 2D 화면의 좌표로 옮길까)

- 래스터라이저(Rasterizer)를 통해 도형에 화면의 픽셀 위치를 표시 (각 모니터의 무슨 픽셀에 찍으면 돼)

픽셀 셰이더(Pixel Shader)

- 픽셀의 최종 색상을 계산하는 것 (그 픽셀에 어떤 색깔을 넣을까?)



애니메이션과 이펙트

3D 에니메이션의 특징

1. 3D 애니메이션의 구현 방법

- 코드에 의한 구현

- 애니메이터의 수작업 (과장된 동작 표현)

- 모션 캡처를 통한 데이터 구현 (실제 움직임을 자연스럽게 표현 가능)

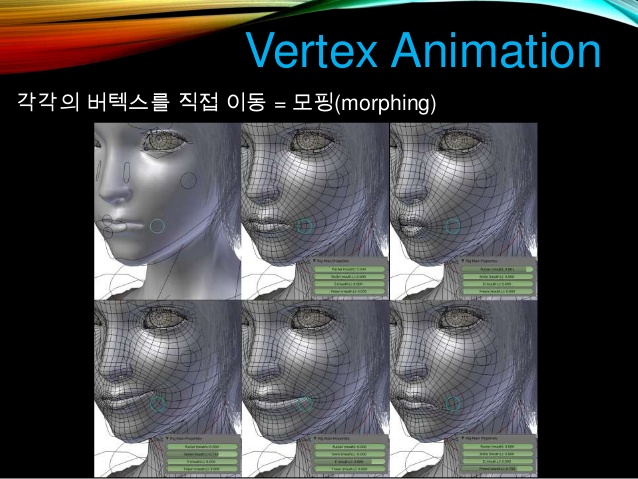
2. 동작과 동작 간의 보간에 대한 명확한 정의가 필요

3. 3D 모델의 파트를 나누어 파트 별로 애니메이션을 따로 만들 경우 파트 별 애니메이션의 자연스러운 조화에 대한 고려가 필요

3D 애니메이션 기초

버텍스 애니메이션(Vertex Animation)

- 각 정점 간의 위치를 프레임 별로 저장 (얼굴 표정, 모핑 등 사용)



트리 계층 애니메이션(Hierarchy Animation)

- 계층 구조를 만들어 부모의 이동이 자식 노드에게 영향을 준다

- 모든 애니메이션 구조의 기본

본 애니메이션 = 스켈레탈 애니메이션(Skeletal Animation)

- 피부 메시 안에 뼈대를 심어 뼈대의 움직임에 따라 메시가 움직이는 방식

- 본의 계층 구조가 같으면 다른 모델에도 동일한 애니메이션 적용 가능

리깅(Rigging), 스키닝(Skinning): 캐릭터 메시와 뼈대를 연결해 주는 작업

바이패드(Biped): 인간형 기본 뼈대를 제공하는 모델

메카님, 페르소나

메카님(Mechanim): 유니티에서 제공하는 캐릭터 애니메이션 솔루션

페르소나(Persona): 언리얼 엔진4에서 제공하는 캐릭터 애니메이션 솔루션

스켈레탈 애니메이션과 편리한 리깅(스키닝)을 제공

리타겟팅: 만들어진 애니메이션을 다른 캐릭터가 공유하여 사용하는 기술

애니메이션 블랜드: 여러가지 애니메이션을 자연스럽게 연결시켜주는 기술

애니메이션 콘트롤: FSM(유한 상태 머신)기반의 애니메이션 콘트롤 제공

- 캐릭터의 움직임을 상태(State)로 정의하고 외부 조건에 따른 상태의 전이(Transition)를 통해 캐릭터의 전체적인 움직임을 구성

3D 이펙트의 특징

3D 이펙트의 구현

1. 메시 사용하는 방식(ex. 메테오, 불기둥 스킬 이펙트)
2. 빌보드 방식(ex. 타격 이펙트, 투사체 이펙트)
3. 파티클 방식(ex. 빗방울, 눈송이, 연기 등)

게임 이펙트는 생성, 유지, 소멸의 3단계로 구성

파티클 시스템

파티클(Particle)

- 눈, 비, 폭파 등을 표현하기 위한 수단으로 사용되는 작은 크기의 입자 오브젝트로 구성된 것

파티클 시스템

- 동적으로 파티클을 생성해주고 움직여주고 형태를 정의하는 역할을 하는 시스템

- 일반적으로 파티클 시스템은 생성기, 애니메이터, 렌더러로 구성

파티클 시스템 구성요소

1. 파티클 이미터(Particle Emitter)

- 파티클의 생성을 담당

- 생성된 파티클은 수명을 가지며 시간이 다 되면 스스로 소멸

- 생성 속도와 수, 수명 시간, 범위와 방향 등을 설정

1. 파티클 애니메이터(Particle Animator)

- 파티클의 움직임을 관장하는 역할

- 크기, 회전, 가속도, 색상, 투명도 등을 설정

1. 파티클 렌더러(Paticle Renderer)

- 파티클이 보여지는 방식을 담당

- 파티클이 보여지는 방향, 재질(Material)을 정의

배경 관련 기술

지형(Terrain)

터레인 시스템: 커다란 야외 환경용 메시 제작 수단을 제공하는 것

- 하이트 맵(Heightmap): 언덕, 계곡, 강 등 높낮이를 구현하는 텍스처

- 다중 맵: 지형의 재질을 표현하기 위해 여러 레이어의 텍스처 사용

- 트리 맵: 말 그대로 나무나 풀들을 배치하기 위해 사용 (파티클과 비슷한 원리)

LOD(Level of Detail)

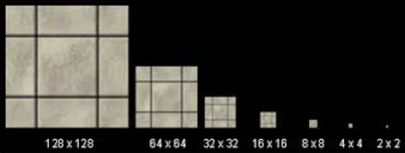
카메라의 거리에 따라 메시의 정밀도를 변화시킨다



밉 맵(Mipmap)

- 하나의 텍스처가 여러 개의 사이즈 이미지를 갖고 있어 LOD에 맞게 해당 이미지 표시

(밉 맵을 사용하지 않을 경우 ‘모아레’ 현상 발생)



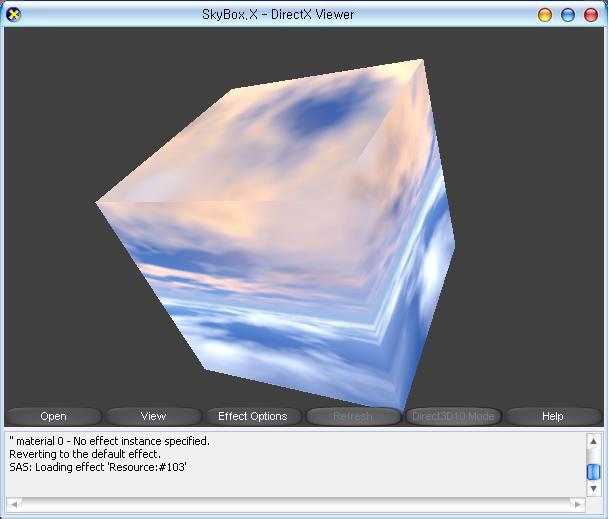


모아레 현상

스카이 박스(Skybox)

원경(보통 하늘)을 표시하기 위해 반구 또는 육면체 형태의 페이스 위에 텍스처를 덮어 사용

- 스카이 박스 말고도 여러가지 형태가 있다(스카이돔)



컬링(Culling)

보이지 않는 곳을 그리지 않고 생략

물리적인 세계의 구현

충돌 체크

충돌 메시(Collider): 충돌 체크를 위해 설정한 메시

물리 엔진

현실 세계의 물리 현상(힘, 관성, 속도, 가속도, 중력, 마찰력)을 실시간으로 시뮬레이션해서 현실감 있는 상황을 나타내는 데 사용

물리 엔진의 주요 기능

1. 강체(Rigid Body) {형태와 질량을 보존하는 물질} 동역학(Dynamics) 계산
2. 충돌 검출(Collision Detection)
3. 입자(Particle)
4. 유체 역학(Fluid Physics)
5. 옷감(Cloth)
6. 래그돌(Ragdoll) – 몸에 충격을 가했을 때 신체가 어떻게 변화하는 지

강체 동역학

강체: 어떤 외부 자극에도 모양과 크기가 유지되는 물체

동역학: 힘 🡪 운동 🡪 상태 변화에 대한 계산

충돌 검출

충돌 자체의 체크 + 충돌 방향, 힘, 반작용 등의 계산

입자

- 눈, 비, 연기 연출에 필요

유체 역학

물결, 파도 등 액체의 움직임을 구현

옷감

- 망토, 머리카락, 깃발 등의 움직임 구현

래그돌

인체의 물리적인 반응 구현

스포츠 게임에서의 상황에 따른 자연스러운 움직임이나 FPS에서의 추락, 사망 처리 등에 사용