**게임알고리즘심화과정**

**과목명 : 게임 일반 프로그래밍**

**능력단위 : 게임 그래픽 프로그래밍**

**제출일자 : 2021년 11월 05일**

**포트폴리오 : Skinning/Bind Pose 애니메이션, 캐릭터의 모션 변경 및 이동 처리 구현**

**작성자 : 최지원**

**(제출내역)**

1. **Skinning/Bind Pose 애니메이션이 구현된 3D 엔진 프로젝트**
2. **기능이 구현된 3D게임 엔진 체계도 및 세부 문서**

**<목차>**

1. **프로젝트의 소개 및 개요**
   1. 프로젝트 소개
   2. Skinning
   3. Bind Pose
2. **프로젝트의 설계 및 세부적 기능 단위로 다이어그램 기술**
   1. UML 다이어그램
      1. 클래스 다이어그램 (첨부)
      2. 시퀀스 다이어그램 (첨부)
3. **프로젝트의 구현된 주요 기술 단위로 분석 및 설계 기술**
   1. Skinning/Bind Pose 애니메이션
      1. 주요 변수, 함수 설계
   2. 캐릭터의 모션 변경 및 이동 처리
      1. 주요 변수. 함수 설계
4. **최종 결과 및 추가된 내용**
   1. 결과물 화면 (첨부)
   2. 범용성/ 유연성/ 확장성/ 간결성 고려하여 추가된 내용
5. **프로젝트의 소개 및 개요**
6. **프로젝트 소개**

DirectX 11 윈도우 기반 3D 그래픽 프로그래밍으로 FBX형식의 파일을 로드해서 오브젝트를 렌더링할 수 있다.

데이터가 많지만 계산이 적은 샘플링 방식으로 애니메이션을 렌더링 할 수 있다.

캐릭터의 이동 처리를 구현하여, 이동할 시에 걷기, 달리기 등 원하는 구간의 애니메이션을 출력하거나, 특정 키를 누르면 해당 오브젝트의 모든 애니메이션 구간을 재생하도록 구현하였다.

일반적으로 게임에서는 자체적인 3D 오브젝트 Exporter 플러그인을 사용한다. 개발자가 엔진에 필요한 정보만을 추출해서 원하는 포맷 형식으로 출력하고 빠르게 로드하기 위함이다. FBX파일을 Export, Import하기 위해서는 3DS MAX SDK를 사용해야한다.

본 프로젝트에서는 FBX파일을 직접 Import해서 애니메이션을 구현하기 위해 3DS MAX SDK를 사용했다. 3DS MAX 는 Up Axis가 Z축인 반면, 본 다이렉트X11은 Y축이다. 또한 변환 행렬 계산방식도 다르다. (DirectX는 행기준) 행렬의 저장 방식에 따라 계산 방식이 달라지기 때문에 본 프로젝트에서는 FbxAxisSystem을 MayaZup를 사용해 Up Axis가 다르던 간에 똑같이 출력되는 효과를 얻을 수 있다.

1. **Skinning**

애니메이션은 오브젝트 애니메이션과 캐릭터 애니메이션으로 구분할 수 있다. 오브젝트 애니메이션은 단일 메쉬에 포함된 정점 (버텍스 버퍼)에 동일한 행렬이 적용된다. 캐릭터 애니메이션을 이와 똑같이 적용하면 근육이 떨림이나, 관절의 움직임에 디테일한 애니메이션을 구현하지 못한다.

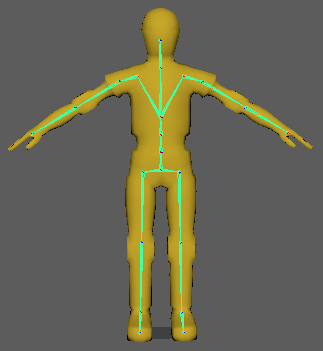
따라서 캐릭터 애니메이션은 가중치 (Weight)를 사용하여 여러 행렬들을 가중치를 따로 주어서 곱해서 합산한 것으로 최종 애니메이션 버텍스 좌표로 쓴다. 즉, 가중치는 정점당 어떤 행렬의 영향을 얼만큼 받느냐를 따져준다. 이와 같은 방식을 **Skinning**이라고 한다.

스키닝에는 블랜딩, 논블랜딩 방식으로 나뉜다. 논블랜딩은 말그대로 혼합하지 않는다는 뜻으로, 하나의 뼈 안에서 대부분의 스킨 정점들은 논블랜딩이다. 가충치를 1로 둔다. 반면에 블랜딩은 팔을 구부리면 어께 근육도 움직이듯이 각각의 움직임에 따라 영향도가 다른 것이다.

텍스트, 클립아트이(가) 표시된 사진

자동 생성된 설명모델링에서는 본과 연결하는 가중치를 입력하여 본의 움직임에 따라가도록 버텍스를 본에 연결하는 것을 리깅이라고 한다.

source : AnyPortrait

1. **BindPose**

source : Autodesk

BindPose는 T-Pose와 유사한 의미로 초기 뼈가 스킨을 묶을 때 있는 위치이다. 초기 위치값 행렬인 BindPose는 T-Pose로 돌아가는 행렬이다. 본 또한 각각 자신의 로컬 좌표계 기준을 갖고 있기 때문에 본 좌표계로 바꿔줘야한다. 그렇게 바꿔준 뼈의 위치 역행렬과 애니메이션 행렬을 곱하면 초기 애니메이션, Start Pose를 만들 수 있다.

본 프로젝트에서는 하나의 애니메이션으로 프레임 마다 다른 애니메이션이 있다. 반면에 캐릭터 오브젝트 파일 따로 동작별로 애니메이션 파일 따로 Skin 파일을 공유하며 사용하는 경우 BindPose 애니메이션이라고도 한다.

1. **프로젝트의 설계 및 세부적 기능 단위로 다이어그램 기술**
   1. UML 다이어그램
      1. 텍스트이(가) 표시된 사진

         자동 생성된 설명클래스 다이어그램 (첨부)

그림 2 클래스 다이어그램 (Sample.cpp)

|  |  |
| --- | --- |
| 자체 엔진EngineCoreD라이브러리를 사용하는 클래스 | |
| Sample 클래스 | EngineCoreD 라이브러리를 사용해서 wWinMain 함수로 윈도우 창을 생성, 다이렉트X 환경을 구성하여 FBX 로드, 캐릭터의 애니메이션, 이동 등 각종 객체를 관리하는 클래스 |
| KLight 클래스 | 라이트 관련 클래스 |
| KMinimap클래스 | 라이트 디버깅 미니맵 클래스 |

텍스트이(가) 표시된 사진

자동 생성된 설명

그림 3 클래스 다이어그램 (EngineCoreD.Lib)

|  |  |
| --- | --- |
| 구현된 기능의 주요 클래스 EngineCoreD 라이브러리 | |
| KFbxObj 클래스 | FBX 파일 해석 및 로드, 애니메이션 |
| KModel 클래스 | 버퍼 생성, 쉐이더 관리, 오브젝트 렌더 클래스 |
| KMesh 클래스 | 모델을 상속받아 매쉬별로 관리하는 클래스 |
| KWindow클래스 | 윈도우 관련 클래스 |
| KDevice클래스 | DirectX11 관련 클래스 |
| KCore 클래스 | Input, Timer, Write, Window, Device 등 게임 엔진 라이브러리 기본 클래스 |

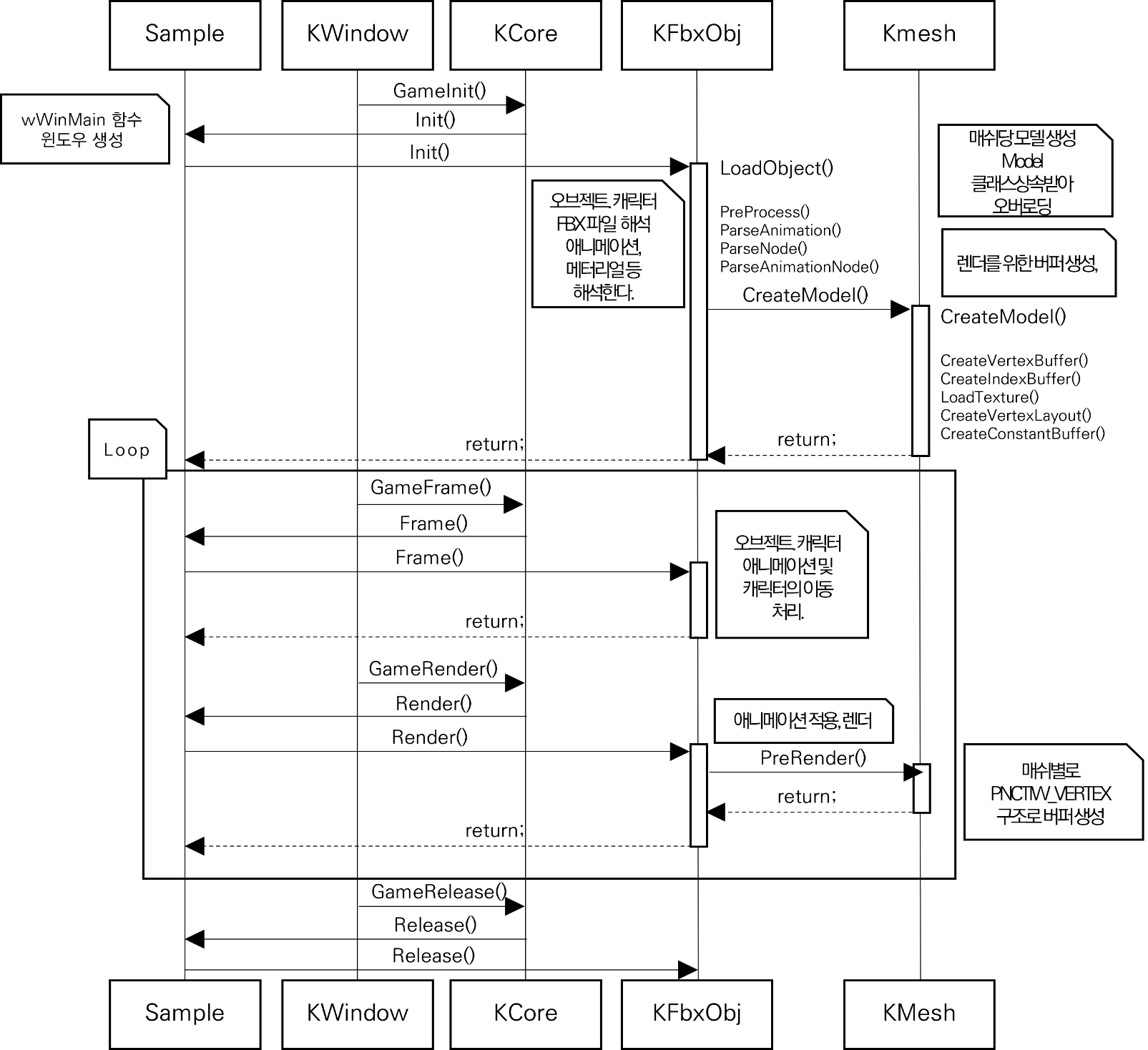
* + 1. 시퀀스 다이어그램 (첨부)

그림 4 시퀀스 다이어그램

|  |  |
| --- | --- |
| 1 | 오브젝트 캐릭터 FBX 파일(버텍스, 애니메이션 정보) 해석, 저장 |
| 2 | 매쉬당 모델 생성 렌더를 위한 버텍스 버퍼 생성 PNCTIW |
| 3 | LookVector 사용해 캐릭터 이동 처리 및 상황에 맞는 애니메이션 |
| 4 | PNCTIW 가중치로 스키닝된 캐릭터 렌더 |

1. **프로젝트의 구현된 주요 기술 단위로 분석 및 설계 기술**
   1. Skinning/Bind Pose 애니메이션
      1. 주요 변수, 함수 설계

|  |  |  |  |
| --- | --- | --- | --- |
| **클래스**  **/구조체** | **타입** | **인스턴스**  **/변수명** | **내용** |
| Sample | KFbxObj | m\_FbxObjA | 캐릭터 타입 FBX 파일의 객체 |
| m\_FbxObjB | 오브젝트 타입 FBX 파일의 객체 |
| KFbxObj | FbxManager\* | m\_pFbxManager | 3DS SDK를 사용해 FBX Importer설정 및 환경 구축 |
| FbxImporter\* | m\_pFbxImporter |
| FbxScene\* | m\_pFbxScene |
| bool | m\_bAnimPlay | 애니메이션 진행 상태 bool 값 |
| float | m\_fElpaseTime | 애니메이션 경과 시간 |
| int | m\_iAnimIndex |  |
|  | KVector3 | m\_vLook |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

* 1. 캐릭터의 모션 변경 및 이동 처리
     1. 주요 변수. 함수 설계

1. **최종 결과 및 추가된 내용**
   1. 결과물 화면 (첨부)
   2. 범용성/ 유연성/ 확장성/ 간결성 고려하여 추가된 내용