|  |
| --- |
| **[ Catch Bear (캐치 베어) ]** |
| [ 3주차 ] 2022. 1. 9 ~ 2022. 1. 15 |
| 1/9 일) 파일 입출력: 텍스트 파일 -> 바이너리 파일 변경  1/10 월) FBX Exporter에서 발생한 msvcrtd.lib 링커 오류를 해결하는데 생각보다 많은 시간이 걸림.(링커->추가 라이브러리 디렉터리: VS\lib\onecore\x64) 성공적으로 바이너리 파일을 만든 후, 바이너리 파일을 불러오는 부분을 다른 프로젝트에 구현함(이 프로젝트는 아직 깃에 추가하지 않음.) 정적 메쉬를 렌더링하는 쉐이더 구현중이고, 내일 이어서 할 예정.  1/11 화) 정적 메쉬를 렌더링하는 쉐이더 구현을 하였다. 일단 FBX Exporter에서 구현을 완료한 메쉬의 정점 정점 정보들을 가지고 와이어 프레임으로 렌더링하는 것을 성공하였다. FBX Exporter에서 메쉬의 재질/텍스처 정보를 로드하는 부분을 구현하기 시작하였다.  1/12 수) FBX Exporter에 메쉬의 재질과 텍스처 정보를 로드하는 것을 추가로 구현하였다.  1/13 목) FBX Exporter에 메쉬의 애니메이션 정보를 로드하는 것을 추가로 구현하였고, 플레이어의 애니메이션 정보들이 잘 로드되는지 확인하였다. 텍스처와 재질 관련 정보들을 어떻게 추출해서 클라이언트에 어떻게 적용할지 내일 회의때 같이 의논해봐야 할 것 같다.(구현할 때 필요한 정보는 무엇인지)  1/14 금) 회의를 한 후, 메쉬에 텍스처를 입히는 것을 구현하기 시작하였다. 일단은 텍스처 좌표들의 정보를 읽어 들여서 버퍼 리소스로 만드는 것 까지 구현하였다.  1/15 토) 클라이언트 프로그램에서 바이너리 파일로부터 텍스처 매핑에 필요한 정보들을 모두  읽어 들여서 변수에 저장하였다. (텍스처, Ambient, Albedo, Emissive, Specular 등등) 내일 텍스처  와 재질, 조명을 적용하는 것을 시작해야겠다.(레지스터 번호 다시 확인해보기!) |
| [ 4주차 ] 2021. 1. 16 ~ 2021. 1. 22 |
| 1/16 일) 전날 텍스처링에 필요한 정보들을 모두 읽어온 것을 바탕으로 환경 오브젝트 메쉬 텍스처링을 구현하였고, 여기에 조명과 재질도 추가해서 구현하였다.(플레이어 메쉬는 아직 렌더링X) 근데 환경 오브젝트의 메쉬가 생각보다 너무 커서 일부러 scale값을 줄여서 렌더링했는데 로우 폴리곤 느낌이 많이 나지 않았다. 플레이어 메쉬까지 같이 렌더링해 본 다음, 환경 오브젝트 메쉬들이 많이 크고, 로우 폴리곤 느낌이 많이 나지 않을 경우, 환경 오브젝트 메쉬들을 다시 찾아봐야할 것 같다.(회의때 정하기로 함!!) 그리고 플레이어 애니메이션 정보들을 FBX Exporter에서 어떤 순서로 적는지 노트에 정리하였다.  1/17 월) 주말부터 목이 건조하더니 감기에 걸린 것 같아서 오후에 코로나 검사를 받고, 병원에 다녀와서 밥과 약을 먹고 쉬느라 작업을 거의 하지 못했다.  1/18 화) 어제에 비해 증상이 나아지기는 했지만 여전히 목이 아프고, 기침도 많이 나와서 작업을 많이 하지는 못했다. 플레이어 메쉬의 fbx파일을 추출해서 그것을 바탕으로 바이너리 파일을 만들어서 텍스처, 재질, 조명을 적용해서 렌더링을 해보았다. 다음으로 스킨 디포메이션 정보를 읽어들이는 부분을 디버깅으로 확인해보면서 구현해야겠다.  1/19 수) 기존 프로젝트의 플레이어에서 스키닝 정보(스킨 디포메이션)도 추가로 읽어들여서 렌더링하는 것을 구현하였다. 텍스처링을 위한 메쉬와 쉐이더(TexturedMesh, TexturedShader)와 스키닝을 위한 메쉬와 쉐이더(SkinnedMesh, SkinnedShader)를 합친 형태인  TexturingSkinnedMesh, TexturingSkinnedShader를 새로 만들어서 텍스처링을 하면서 스키닝 애니메이션을 할 수 있는 형태로 만들었다. (-> 이 3종류의 쉐이더는 나중에 상속관계로 더 간단하게 표현할 수 있을 것 같지만 지금은 공부를 위해 각각 따로 구현하였다.)  1/20 목) 플레이어 메쉬를 성공적으로 렌더링하였다. 하지만 Idle 애니메이션을 아직 추가하지 못하였다. .fbx 파일로부터 애니메이션을 읽어 들이는 부분을 다시 공부해서 디버깅해보면서 데이터들이 제대로 저장되는지 확인하였다.  1/21 금) 애니메이션을 재생하도록 코드를 변경하였더니 애니메이션은 커녕 플레이어조차 제대로 렌더링되지 않는다…(이상한 색으로 그려진다) 스키닝 관련 정보들을 쉐이더로 제대로 넘겼는지, 레지스터 번호를 맞게 설정했는지 확인해봐야겠다. |
| 5주차 |
| 1/22 토) 플레이어 IDLE 애니메이션을 렌더링해보았다. 근데 제대로 되지 않았다. 몸은 움직이지만 몸의 뼈는 움직이지 않아서 이상하다. 렉 걸린 것 같다. 관련해서 어디가 잘못 되었는지 찾아보았다.  1/23 일) 어제 계속 찾아보다가 스키닝에 대해 이해가 잘 안된 것 같아서 애니메이션 특강 1일차 녹화본을 다시 보면서 노트에 정리해보면서 공부하였다.(프레임 계층구조, 스키닝)  1/24 월) 다른 프로젝트에서 애니메이션 렌더링을 해보고있다. 하지만 똑같다. 안된다.  1/25 화) 애니메이션 특강 2일차를 녹화해둔 것을 다시 듣고 노트에 정리해보면서 공부하였다. 다른 프로젝트들도 살펴보면서 스키닝을 어떻게 구현하고 있는지, 메쉬와 쉐이더가 어떤 구조로 짜여져 있는지 위주로 공부하였다. 계속 하다가 너무 안돼서 결국 질문 게시판에 질문을 올렸다.  1/26 수) 애니메이션 특강에서 진행했던 프로젝트들을 복습하였다. 정점의 구조와 어떻게 쉐이더로 넘기는지, 애니메이션이 돌아갈 때 행렬들이 어떻게 변환되는지, 키 프레임 애니메이션에 대해 집중적으로 살펴보았다.  1/27 목) 질문을 다시 보내고 문제점을 확인하다가 해결하지 못했다. 친구 생일이라 노느라 많이 못했다.  1/28 금) 교수님께 답장을 받고 문제점을 확인하였다.(모델 파일에 애니메이션 데이터가 많이 없고, 동작 별 애니메이션 파일이 따로 존재한다.) 그래서 플레이어 메쉬와 애니메이션을 불러들이는 함수를 따로 만들어서 추가하였다. 근데 또 다른 문제가 생겼다.(플레이어 파일에는 없는 뼈 이름이 애니메이션 파일에는 존재한다) 애니메이션 파일에만 있는 뼈들의 행렬은 모두 초기화해서 렌더링했더니 다리가 몸쪽으로 가서 이상하게 렌더링된다. 다른 뼈들 애니메이션은 잘 재생된다. |
| 6주차 |
| 1/29 토) 금요일에 발생한 문제점의 원인을 파악했다. 모델fbx파일에 없는 뼈가 애니메이션fbx파일에는 존재해 계층구조가 조금 달라서 발생한 문제였다. 모델과 애니메이션 fbx파일에서 읽어들일 데이터를 정리하였다.  1/30 일) 어제 정리한 데이터를 바탕으로 Load하는 부분을 다시 설계하였다.(진행중) 모델과 애니메이션 파일의 프레임들을 부분적으로 읽어 들여야할 것 같은데 프레임의 중간부터 읽어들이는 것이 문제인 것 같다. 잘 안된다.  1/31 월) 가족과 시간을 보내느라 작업을 많이 못했다.  2/1 화) 데이터를 로드하는 부분을 전에 설계한 내용을 바탕으로 코드를 수정했다. 로드하는 방법을 변경한 만큼 익스포터의 코드도 변경해야해서 생각보다 시간이 오래 걸릴 것 같다.  2/2 수) 정보를 로드하는 방법을 새로 구현하였다. 계층구조에 이어서 애니메이션 동작의 정보까지 로드하였다. 그동안 만든 함수들이 많아서 새로 정리하였다.  2/3 목) 쉬었다.  2/4 금) 쉬었다.  2/5 토) 바이너리 파일에서 데이터들은 잘 읽어 들였는데 애니메이션 컨트롤러를 생성할 때 모델 객체의 메쉬 부분이 비어있다.(분명 메쉬 정보를 읽어 들여서 메쉬를 생성했는데….) 정보들정 읽어들일 때 하나의 객체가 아닌 하나의 객체 안에서 여러 객체를 생성하는 것 같다. 이와 관련된 것 같은 부분들을 찾아보고 있다. |
| 8주차 |
| 2/13 일) 유니티에서 모델의 계층구조와 애니메이션fbx파일에 나타나는 계층구조와 뼈들을 살펴보고, 서로 존재하지 않는 뼈들을 확인했다. 그리고 계층구조로는 모델 륱를 사용하기로 하였고, 애니메이션 커브 노드에서 없는 뼈들과 같은 뼈로 보이는 뼈들을 확인하였다. 새로 추출하기로 한 정보를 바탕으로 FBX Exporter를 바이너리/텍스트 버전 모두 변경하였다.  2/14 월) 추출하기로 한 데이터를 기반으로 fbx파일들을 읽어들이는 구조를 다시 설계하고 구현하고 있다. (70%) |