|  |  |  |  |
| --- | --- | --- | --- |
| MORPHOSIS 개발일지 12차 | | | |
| 기간 | 2019-05-15 ~ 2019-05-29 | 작성자 | 신재욱 |
| 작업 내용 | | | |
| 애니메이션도 중요한데 일단 그거 말고도 해야 되는 것들 다시 한 번 정리해보자.  - FBX SDK에서 UV 좌표 얻어오게 하기.  - Mesh 렌더링 됐던 코드 다시 살리기.  - 3ds Max에서 레벨 만들고 사용한 텍스쳐 종류들 받아오게 하기.(여러 텍스처를 쓰면 UV 좌표를 0~1, 1~2 이런 식으로 할까? 아니면 그냥 텍스처를 다른 걸 쓰는 메쉬끼리는 아예 떼어내서 다른 오브젝트로 만들까?)  - 유니티 엔진에서 예전에 배웠던 버텍스에 맞춰서 정렬하는 기능 써서 충돌 맵 만들고 점령지점이랑 캐릭터 스폰 포인트 만들고 파일 만들어서 추출하기.  - 캐릭터 충돌처리 부분 깔끔하게 고치기.  - 투사체 만드는 부분 살리기.  - 투사체에 컴포넌트 속성 넣기.  - UI 띄우기.  - 디버그를 위해서라도 알파벳과 숫자는 폰트로 띄울 수 있어야 함.  - 전체화면 하는 것도 해야 됨.  지난 이야기:  컨버터와 기타 등등 문제를 해결했고 이제 본 클라에서 값이 잘 들어가는지만 보면 되는 것인데!  이제 파일은 정상적으로 들어온다고 가정하자. 테스트 클라에서 잘 들어갔으니까 똑 같은 코드인데 잘 들어가겠지(안일). 현재 방식은 앞서 설명했던 대로 1. GenerateToWorldMatrix()에서 Lcl행렬을 만들고, 그걸 본 행렬의 Lcl 행렬에 넣은 다음에 그걸 가지고 MakeToWorldMatrix()를 한 뒤, 그 결과물과 Offset 행렬을 곱해서 최종행렬을 만들고 그걸 HLSL에 올려주는 방식이다. 이제는 GlobalTransform이 있으니까 굳이 로컬행렬을 만들고~ to월드행렬 만들고~ 할 필요가 없다. 글로벌 변환 행렬을 그대로 보간해주고 그걸 Offset행렬과 곱해주면 아마…… 기대하는 결과가 나올 것…… 안 나오면…… 큰 일 남…….    함 해보자!    응~ 어떻게 디버깅을 할 지 다시 계획을 짜보자.  1. 저 글로벌 변환 행렬이 내가 생각한 행렬이 맞는가?(Offset과 곱하면 되는 행렬이 맞는가)  2. 곱하는 방식에 문제는 없는가?(예를 들어 보간하는 과정이나 YZ축 문제 등)  3. 추가적으로 변환(예를 들어 전치행렬 등)을 해줘야 하는가?  앗 너무 행복해  아 진짜 FBX SDK 너무 싫다. 의미부터 다시 공부해야 할 듯. 근데 의미를 안 적어놓잖아. 한 번 다시 찾아보자.      LclTranslation.Get()으로 바로 받아올수 있구나 했는데 생각해보니까 언제 그 값이어야 하는지 알아야 해서 안 됨.    Global Transformation이랑 Local Transformaion이랑 뭐가 다른거지?    함수 설명에 Global Transform이 뭔지 설명이 있을 줄 아셨습니까? 유감!    Remarks에 Local Transform Matrix가 무엇인지는 나와있다. 부모의 글로벌 변환에 글로벌 변환을 곱한거라고?    ‘To evaluate these properties separately without taking pre/post rotation, pivots and offsets into consideration, please use GetNodeLocalTranslation(), GetNodeLocalRotation() and GetNodeLocalScaling().’????  피봇이나 오프셋이나 그런거 다 집어치우고 그냥 로컬 회전만 받을 수 있다고?  테스트를 하려면 많은 변화가 필요할 것 같다…  테스트 용도로 쓰고 있던 Project2 프로젝트에 fbxsdk.h를 추가하고 프로젝트 속성에서 참조 디렉토리에 해당 헤더의 위치를 추가해줬다. (이거 하는 법 몰라서 작년엔 샘플 프로젝트를 컨버터로 바꿨어야 했으므로…)    아니 포맷 해줬더니 이렇게?  원래 FBX SDK를 설치할 때 모든 파일을 다 받았었는데 컴퓨터를 포맷하고 샘플 프로젝트들만 깃으로 푸시해놨었기 때문에 생긴 문제인 것 같다.  그래서~ FBX SDK를 새로 받았는데~~ 얘네가 버전을 업데이트 하면서 프로젝트 파일을 없에고 CMake로 만들게 바꿔두었다~~    ;;;  분명 이전 버전을 남겨두지 않았을까???    감사합니다.    잘 생각해보자. 지금 필요한 것들.  Bone들(Offset 행렬 있어야 함), key들(시간, 관련 있는 Bone)(그래야 그 Bone이랑 시간 가지고 그 시간에 그 Bone이 어떻게 움직이고 있는지 알 수 있음, 아마 그걸로 LocalTransform이든 GlobalTransform이든 나올 것)  저게 있으면 애니메이션 데이터는 끝이다. 시간에 따라서 행렬 저장해주면 될 듯.  파일의 양식은 아마 Bone의 기본 Offset 행렬들과 키들의 정보(키의 시간, 본의 idx와 각 본의 그 시간대별 변환 행렬들).    행렬 얻어오는 함수들은 FbxNode에 있음.  FbxNode가 Bone인지는 Node의 속성을 확인하면 됨.    아마 이런거 써서 하면 될 것 같음.    아 뭐야  분명 디렉토리까지 설정해줬는데 왜 정의를 못 찾는 것일까요?    와!  <https://blessingdev.wordpress.com/2017/09/26/visual-studio%EC%97%90-%EC%99%B8%EB%B6%80-%EB%9D%BC%EC%9D%B4%EB%B8%8C%EB%9F%AC%EB%A6%AC-%EC%B6%94%EA%B0%80%ED%95%98%EA%B8%B0/>  의 글을 보고 참고하여 외부 라이브러리 추가를 해보았습니다.    암 생각 없이 .lib 확장자 파일을 전부 추가해줬더니    룰루~  -md, -mt를 빼고 기본만 넣어주었습니다.    그나마 정상이 되었군요.    저 주소로 가서 Common.cxx를 프로젝트에 추가해주니까 해결되긴 했는데 저걸 그냥 추가 안 하고 쓸 수는 없나… 일단 그게 중요한 내용이 아니므로 패스.    아.  프로젝트를 새로 만들고~  VC++ 디렉터리-포함 디렉터리, 라이브러리 디렉터리  C/C++ 추가 포함 디렉터리  링커-입력-추가 종속성 .lib  빌드 되서 나오는 폴더에 .dll 추가  해서 문제 해결.    드디어 여기까지 옴.    아아 정말 완벽해. OK 계획대로 되고 있어.  Bone들을 순회하면서 글로벌 행렬 얻어보기    아니 생각보다 훨씬 편하고 좋은데?  개발일지 10차 문서에서 내가 했던 방식을 가져와보자.  역회전 = LclRotation, 역이동 = LclTranslation  < 순 서 도 >  Lcl Lcl Lcl  이동 회전 이동 회전 이동 회전 역회전 역이동 역회전 역이동 역회전 역이동  (m\_toParent)^-1 m\_toParent  Parent’s DressposeInv Parent’s toWorld  My DressposeInv My toWorld    **끝나지 않는 삽질.**  Lcl Rotation과 Lcl Translation은 부모 좌표계에서 내 좌표계로 바꾸는 방법.  T를 먼저 적용하고, R을 적용하면 내 좌표축이 됨. (저기 나오는 모든 값은 Y축이랑 Z축을 바꿔줘야 함. 저건 맥스 좌표계니까)  부모 변환(ToParent)은 내 좌표계에 있는 점을 부모의 좌표계로 옮기는 변환.  반복하면 뼈 공간의 점이 월드 공간으로 변환됨.  필요한 변환들과 의도를 적어보자.  부모변환행렬: 뼈 공간의 점을 월드 공간으로 변환.  로컬변환행렬: 시간에 따라 뼈가 회전하고 이동한 값만큼 변환.  부모변환행렬의 역행렬: 월드 공간의 점을 뼈 공간으로 변환.  먼저 T포즈의 점을 ‘부모변환행렬의 역행렬’을 곱하면 그 점이 속한 뼈 공간으로 변환된다.  그 상태에서 로컬변환행렬을 곱해서 그 점을 변환하면 걔가 뼈가 회전하고 이동한만큼 회전하고 이동함.  그리고 다시 ‘부모변환행렬’을 곱해서 현재 뼈 공간에서 변환된 점을 다시 월드 공간으로 옮겨준다.  쉽게 생각하면 저 멀리 있는 점을 회전시킬 기준점이 원점이 되도록 끌고와서 회전시킨 다음에 다시 보내주는 것인데, 다시 보내줄 때 고려해야 될 사안이 로컬변환행렬의 내용이다.  처음에 T 포즈의 점을 뼈 공간으로 끌고 올 때는 Offset 행렬만 해주면 되니까 전혀 문제 없는데 부모 뼈들이 변환된 값만큼 적용이 되어야 함. 그걸 위해서 처음에 뼈 공간으로 가져올 때는 Offset 변환 행렬을 곱해주고 다시 월드 공간으로 변환될 때는 로컬변환행렬과 부모변환행렬을 곱해주면 됨. 로컬변환행렬이 시간에 따라 변하기 때문에 시간마다 Bone 마다 로컬변환행렬과 부모변환행렬을 곱한  아!!! 어쨌든 실시간으로 계산해야 하는거잖아!!! 뭘 계산해줘야 하죠?  각 Bone마다 ToRoot변환행렬을 구해야 함.  무조건 부모가 자식보다 인덱스가 작을 수밖에 없다. 그니까 인덱스 순서대로 ToRoot를 구하면 전혀 문제가 되지 않는다. ToRoot를 구하려면 부모의 ToRoot도 알아야 하니까. 부모부터 먼저 구해야 한다.  회전 = LclRotation, 이동 = LclTranslation  < 수 정 된 순 서 도 >  Lcl Lcl Lcl  이동 회전 이동 회전 이동 회전 역회전 역이동 역회전 역이동 역회전 역이동  (toParent)^-1 toParent  Parent’s Offset Matrix Parent’s toRoot  Offset Matrix toRoot  아이고. Lcl이 단위행렬이라는 가정 하에 toRoot를 구해보고 GlobalTransform이 toRoot의 역행렬인지 함 보자.    아니 근데 진짜 쉽긴 쉬운데?  함수 포인터를 한 번 써봤는데 넘 편하다.    값이 맞게 들어간다고는 안 했음.  혹시 몰라! 곱하면 단위행렬 될지도 몰라!  Lcl들은 부모에서 내 좌표계로 만드는 값 -> toParent의 반대. 월드에서 뼈 공간으로 보냄.  이동 회전 순서로 진행. 이것의 역행렬이 toParent.  toParent는 뼈에서 월드로 보냄.  Local변환의 역행렬을 toParent로 저장하고 그걸로 ToRoot를 만든 뒤,    저 ToRoot를 Global이랑 곱했더니 단위행렬이 나오더라~  그럼 Global은 Offset이 맞다. 맞겠지. 드디어! 의미를 얻어냈어!!!!  근데 그러면 아직도 로컬 변환을 모르는데. Local변환으로 그 Bone이 어느 시간대에 얼마만큼 변환됐는지 알 수 있는 걸까? 이젠 KeyTime을 구해야 할 때.  갑자기 테스트 프로젝트가 링크 오류가 터졌다. 뭐가 문젠지 모르겠다. 왜 이런 걸로 고통받아야 하지.  뭐가 문제인지 모르겠음. 왜 이럴까. Project1의 코드를 샘플 프로젝트로 옮겨서 해야겠다. 정말 너무 싫네.  샘플 프로젝트에서도 그런 문제가 생김.  ~~지금 알았는데 여러 사람들의 애니메이션 코드를 참고할 때, 다 같은 부분이 보였던건 실제로 다 같은 코드였기 때문이었다… 애니메이션 셋, 애니메이션 컨트롤러 등은 이용희 교수님의 예제 코드가 있었다.~~  ~~저기서 구하는 GlobalTransform은 아마 맨 앞의 My Dresspose의 역할을 해주지 않을까?~~  ~~LocalTransform이 Lcl일 것이고, 그럼 시간별로 저 뒤의 toWorld 값을 구해서 그걸 키프레임마다 저장해두면? 그리고 그걸 보간해서 사용한다면? 그렇게 하면 행렬들만 저장해도 되지 않을까?~~  ~~LclRotation과 LclTranslation이 부모 좌표계로 가는 변환이었으니까 그걸 바탕으로 구상을 짜보자.~~  ~~Bone마다 일단 자신의 toParent를 만들어두고 부모의 toWorld를 받아서 내 Lcl이랑 어캐 잘 하면 내 toWorld(이용희 교수님은 toRoot 용어를 쓰심)를 구할 수 있을 것. 그걸 시간마다 구해서 행렬로 저장해보자.~~   1. ~~모든 Bone을 돌면서 LclRotation과 LclTranslation으로 toParent를 만듦.~~ 2. ~~모든 Bone을 돌면서 GlobalTransform을 구함.~~ 3. ~~모든 KeyFrame의 KeyTime에 대해서 그 KeyFrame에서 관리하는 Bone들의~~   멘탈 깨져서 아르바이트 하러 갔다가 거기서도 멘탈 깨져서 약간 영혼 빠진 상태로 돌아옴.  왜 멀쩡하게 잘 되다가 내가 그 파일을 열면 갑자기 오류라고 찍찍 그어대는 것인지. 왜 이런 전혀 상관 없는 일들로 방해 받아야 하는지 전혀 모르겠다.    이 오류가 사라지질 않음.    당신, 정확해야 해. -> 안 됨.  아니 이렇게 쓰기 힘들면 어떻게 쓰나.  마음의 고향, ImportScene 샘플로 돌아옴.  여기선 되네. 진짜 왜 될까. 중요한게 그게 아니니까 넘어가자.  필요한 정보는 KeyTime임.    각 Node의 LclRotation의 X축 요소에 관련된 커브를 얻고, 그 커브의 키의 개수(그 Node의 그 변환의 그 축에 관련된 키프레임 수)만큼 돌면서 keyTime과 그 때의 value를 얻어오는 모습이다.  우리가 필요한건 keyTime 뿐.  저 정보들은 Skeleton Node에 들어있는 것을 확인했으므로 Bone 구조체를 만들고 거기에 KeyTime을 다 넣어주자. Set을 써볼까? Set은 중복 허용 안 하고 자동 정렬도 해주니까 KeyTime만 저장하기에 아주 좋을 것. Value는 필요없다. 내게 필요한 것은 그 시간과 그걸로 얻을 변환 행렬이니까.    구조체로 잘 들어갔다.  위에서 테스트하면서 Lcl 없이 변환 행렬을 받았을 때 단위 행렬이 나왔었으니까 이번엔 로컬변환행렬을 시간에 따라 구해볼 예정.    위에 코드 안 적어놔서 내가 못 찾았기 때문에 나를 위해서 적어둠.  EvaluateLocalTransform()에 인자로 FbxTime을 주면 그 때 로컬변환행렬을 주는거 같음.  테스트 삼아 0.1초마다 값을 저장해서 띄워보자.    정말 균일하기 짝이 없는 값들.    내가 영어를 그렇게 못하나 잠시 생각했고.  아니 근데 저거 맞는데 아…. 이게 이렇게 안 쓰면 뭘 어떻게 쓰냐 인간들아.      ????    ????????????????  또 모호합니다 오류 떠서 다시 껐다 켰음. 아니 대체 뭐야?      감사합니다.    3ds Max에서 0번 키프레임과 40번 키프레임이 같은 값을 갖도록 정해진 키..  40번 키프레임의 KeyTime은 1.3333초. 잘 나온다.  Global과 계산했을 때도 괜찮은 값이 나와야 함. 아니. 괜찮을 것이다.    이걸로 0.6666(다리를 굽히고 있을 때)랑 1.3333(다리를 다시 폈을 때)로 비교해봤는데 잘 되는거 같음.  가자 본 게임으로!  구상을 다시 한 번 적어보자.   * Bone을 순회하면서 GlobalTransform과 ToParent를 저장한다. * Curve를 통해 각 Bone들의 KeyTime을 Bone에 저장. * 그 KeyTime마다 그 Bone의 LocalTransform을 저장. * GlobalTransform은 Offset으로, LocalTransform은 부모의 ToRoot와 곱해서 ToRoot로 Export. * 본 클라이언트에서는 Bone의 개수만큼 Offset을 읽고, 각 Bone마다 KeyTime의 개수만큼 ToRoot를 읽으면 됨. KeyTime도 읽어야 함. 그래야 시간으로 보간할 수 있으니까.   생각해보니까 돌아가기만 하면 된다.  그렇다면 굳이 Bone마다 KeyTime을 넣어줄 필요 없다. 전역에 KeyTime을 관리하게 하고 모든 Bone의 행렬을 그걸로 빼자. (어짜피 지금 데이터는 모든 Bone이 다 같은 KeyTime을 가지고 있음)  파일 양식은  KeyTime 개수  KeyTime 개수 \* Double KeyTime  Bone 개수  Bone 개수 \* Offset Matrix  Bone 개수 \* KeyTime 개수 \* ToRoot Matrix  아 디버깅만 하려고 하면    이렇게 떠서 디버깅을 할 수가 없다. 시간이 많았다면 찾아봤겠지만 지금은 어쩔 수 없다.  내 머릿속 컴파일러.    여기서 문제가 생기네????  아니 왜?????????????????  Node가 NULL 인거 아냐? -> 아니었음. 뭐가 문제지.  테스트해보려고 \*t도 0.0f로 바꾸고 했는데 아 모르겠다.  왜 디버깅이 안되냐…… 왜 계속 이런 로직이랑 상관 없는 문제들로 고통을 겪는지 모르겠음.  문제 찾음.  Bone을 두 군데에서 만들어주고 있어서 Parent에서 꼬였든지 하여간 어디서 꼬였던거 같음.  수정했음.    오른쪽 발가락의 변환 행렬들.  얘는 로컬 변환이 없어서 ToParent는 계속 안 바뀜. (저거 용어가 틀렸는데 귀찮으니까 패스)  ToRoot는 위에서 계속 변환되서 바뀌는걸 볼 수 있음.  R Calf 같은 경우엔 ToParent 값도 계속 바뀌는걸 볼 수 있음.  행렬 저장해서 넘기자!  메쉬도 해야 되는거 아냐?  애니메이션 데이터를 바로 넣고 그걸로 움직이게 하기.    일단 Export는 함.  얘는 YZ축 안 바꿔줬다.  Project2의 내용을 되살려서 테스트하기.  아 Bone 이름도 저장해줘야.    잘 들어감. Import 끝.  본 클라로 가자.  본 클라에서 해줘야 하는 것은   1. KeyTime 개수 고정 2. Bone Idx가 다를 수 있음. 3. 행렬 곱하는 부분 | | | |