한국컴퓨터정보학회 하계학술대회 논문집 제31권 제2호 (2023. 7)

루트 모션을 위한 게임 캐릭터 리깅 연구

이상원O
O청강문화산업대학교 게임콘텐츠스쿨
e-mail: one@ck.ac.krO

A Study on Game Character Rigging for Root Motion

SangWon LeeO
OSchool of Game, ChungKang College of Cultural Industries

● 요 약 ●

실시간 3D 렌더링 게임의 제작 환경에서 캐릭터의 움직임은 모션 캡처(motion capture)를 통해 만들거나 애니메이터에 의해 제작된다. 걷기나 달리기 등 일정한 속도로 캐릭터가 움직이는 모션은 캐릭터가 제자리(in place)에서 움직이도록 한 뒤에 게임에서 프로그램에 의해 일정한 속도로 움직임으로써 구현할 수 있다. 하지만 일정하지 않은 속도로 움직이는 모션을 같은 방식으로 적용하면 캐릭터의 이동이 어색해진다. 이런 어색함을 보완하기 위해 언리얼이나 유니티 3D 등의 엔진에서는 루트 모션(root motion) 기능을 사용하고 있다. 그런데 루트 모션을 위한 계층 구조는 애니메이터의 작업 효율을 위한 계층 구조와 다른 측면이 있다. 본 논문에서는 3ds Max를 사용하여 애니메이터 친화적이고 루트 모션에도 적합한 캐릭터 리깅을 제시한다.

키워드: 루트 모션(Root motion), 애니메이션(Animation), 언리얼(Unreal), 유니티 3D (Unity 3D)

I. Introduction

실시간 3D 렌더링 되는 캐릭터의 움직임은 모션 캡처(motion capture)를 통해 만들거나 애니메이터에 의해 제작된다. 걷기나 달리기 등 일정한 속도로 캐릭터가 움직이는 모션은 캐릭터가 제자리(in place)에서 움직이도록 한 뒤에 게임에서 프로그램에 의해 일정한 속도로 움직임으로써 구현할 수 있다. 하지만 일정하지 않은 속도로 움직이는 모션을 같은 방식으로 적용하면 캐릭터의 이동이 어색해진다. 예를 들어 그림 1처럼 망치를 휘두르는 모션은 준비와 휘두르는 액션과 액션 이후 일어서는 순간마다 캐릭터의 속도가 다르다.

Fig. 1. 루트 모션 - 언리얼 엔진 문서

이 문제는 언리얼이나 유니티 3D 등 게임 엔진에서 '루트 모션(root motion)'이라는 기능으로 해결할 수 있다. 캐릭터 애니메이션의 이동과 회전 값을 실제로 게임에서 콜라이더(collider)가 이동하는 것으로 반영함으로써 어색함 없이 게임 캐릭터가 이동한다.

그런데 루트 모션을 위한 계층 구조는 애니메이터의 작업 효율을 위한 계층 구조와 다른 측면이 있다. 루트 모션을 위한 최상위 본은 때때로 진행 방향의 직진 정보만 필요로 하고 좌우 흔들림은 배제해야 하는 경우가 있다. 좁은 외나무다리를 조심스럽게 건너야 하는 게임에서 캐릭터의 골반이 좌우로 흔들리는 자세 때문에 의도치 않게 외나무 다리에서 떨어지는 상황을 플레이어들은 싫어할 것이다. 점프나 공중을 향해 움직이는 스킬(skill) 모션은 골반이 아닌 발바닥 근처를 기준으로 루트 모션이 추출될 필요가 있다. 이처럼 다양한 상황 때문에 바이패드의 Bip001을 루트 모션으로 사용하는 것은 적절하지 못한 경우가 많다. Bip001보다 상위 부모로 Root 본을 만들고 Root 본에 이동과 회전 키를 넣는 방법도 있지만, 바이패드의 IK 공간이 어긋나는 문제 때문에 역시 바람직하지 않다.

본 논문에서는 애니메이터가 기존 방식대로 Bip001에 애니메이션을 한다는 전제에서, 자동화된 Root 본을 추가 구성하고 종속성 루프(dependency loop) 문제를 피해 갈 수 있는 리깅 구조를 제안한다.

II. Materials & Method

1. Bip001을 참조하는 Root 본(bone)

Fig. 2.

Bip001의 트랜스폼 컨트롤러는 Wire 혹은 Expression으로 참조할 수 없어서 ExposeTransform 헬퍼를 경유해서 Root 본의 트랜스폼 컨트롤러에 전송한다(그림2). ExposeTransform 헬퍼는 Expose Node로 Bip001을 지정한다.

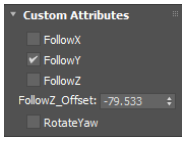

Fig. 3.

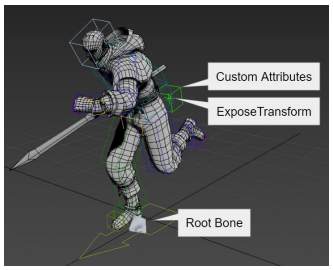

Fig. 4.

Expose된 트랜스폼 정보는 전진/좌우/상하 이동과 회전 정보를 선택적으로 Root 본에 전송하기 위해 그림 3과 같은 커스텀 속성을 가진 컨트롤러 오브젝트를 생성한다.

Root 본의 Position XYZ 컨트롤러는 X, Y, Z 각 축마다 Float Expression 컨트롤러를 사용하여 다음과 같이 적용한다.

```
X Position : WorldPosX*FollowX
Y Position : WorldPosY*FollowY
Z Position : (WorldPosZ+Offset)*FollowZ
```

Z Position의 경우 Bip001의 높이를 그대로 참조하면 Root본의 위치가 공중에 위치하게 되므로 발바닥에 위치하도록 하기 위한 Offset 값이 필요하다. Offset 값은 애니메이터가 입력하는 값인데 ExposeTransform 헬퍼의 World Position Z 값을 참조하여 적용한다. 축 선택 체크박스나 Offset 등 커스텀 속성은 애니메이션이 가능하다는 점을 이용해서 달리다가 점프하는 등 복잡한 상황의 애니메이션에도 직관적으로 응용할 수 있다.

Root 본의 회전을 제어하는 Euler XYZ 컨트롤러는 Z 축에 Float Expression 컨트롤러를 사용하여 다음과 같이 적용한다.

```
Z Rotation : WorldEulerZ*RotateYaw
```

2. 종속성 루프(dependency loop) 피하기

Root 본의 트랜스폼은 Bip001에 의해 결정되고 있는데 계층 구조상 Bip001의 부모가 되면 논리적으로 무한 루프가 발생하여 계층 구조 연결이 불가능하다. 이 문제를 피하기 위해 바이패드 트랜스폼을 참조하는 새로운 계층 구조를 구성하고 Root 본의 자식으로 연결한다. 참고로 바이패드 트랜스폼을 참조하는 새로운 계층 구조를 구성하는 방식은 효율성과 최적화를 위해 게임 제작 파이프라인에서 흔히 사용되는 방식이다.

Fig. 5. - Root본이 최상위에 있고 분리된 계층 구조

바이패드 트랜스폼을 참조하는 새로운 계층 구조는 Position Constrain나 Orientation Constraint 등의 컨트롤러를 사용하는데, Ref Bones Tool 등을 통해 참조 본 생성 자동화가 가능하다.

III. Conclusions

최근 액션 장르 게임의 애니메이션은 점차 복잡하고 정교해지고 있다. 자연스러우면서도 박진감 넘치는 모션을 위해 원본 애니메이션의 트랜스폼 정보가 게임 캐릭터에 그대로 적용되는 Root 본의 자동화 리깅은 제작 생산성과 결과물 품질 향상에 많은 도움이 된다.

하지만 본 논문에서 제시한 리깅 연구는 Offset 높이 값을 직접 입력해야 한다거나 회전 제어 등 개선할 부분들이 아직 남아있다.

REFERENCES

[1] 루트 모션, 언리얼 엔진 문서 - https://docs.unrealengine.com/4.26/ko/AnimatingObjects/SkeletalMeshAnimation/RootMotion/

[2] 루트 모션, 유니티 3D 엔진 문서 - https://docs.unity3d.com/kr/2023.2/Manual/RootMotion.htm

[3] Ref Bones Tool - https://cafe.naver.com/pinksox/5035