

가상환경에서 저글링 움직임을 효율적으로 처리하기 위한 대칭기반 데이터-드리븐 기법

김민지^o, 김종현^{*}

^o강남대학교 소프트웨어융합학부,

^{*}인하대학교 소프트웨어융합대학 디자인테크놀로지학과

e-mail: jonghyumkiun@inha.ac.kr

Symmetry-Based Data-Driven Method for Efficiently Handling Juggling Motion in Virtual Environments

Min Ji Kim^o, Jong-Hyun Kim^{*}

^oSchool of Software Application, Kangnam University,

^{*}College of Software and Convergence (Dept. of Design Technology), Inha University

● 요약 ●

본 논문에서는 데이터-드리븐 기법을 이용해 가상환경에서 사용자의 동작에 따라 아바타의 저글링 움직임을 자연스럽게 처리할 수 있는 방법을 제안한다. 사용자의 저글링 동작 정보를 이용하여 아바타의 움직임을 제어할 뿐만 아니라 가상 공의 궤적을 실시간으로 표현할 수 있다. 이 과정에서 사용자의 손위치 정보를 모두 활용하는 것이 아닌, 한 쪽 손의 데이터를 기반으로 다른 쪽 손의 궤적을 합성한다. 또한 계산량이 큰 물리 기반 최적화 과정이 아닌, 상대적으로 경량화된 기법인 포물선 운동을 활용해 가상 공의 궤적으로 실시간으로 표현할 수 있는 결과를 보여준다.

키워드: 대칭(Symmetry), 데이터-드리븐(Data-driven), 전처리(Preprocessing), 아바타(Avatar), 애니메이션(Animation), 저글링(Juggling), 가상환경(Virtual environment)

I. Introduction

가상증강현실 시장은 점점 많은 분야에서 적용되며 계속 성장하는 추세이며, 앞으로는 데이터와 인공지능 기반의 새로운 고품질 콘텐츠들이 지속해서 요구될 것으로 전망된다. 사용자의 몰입을 증폭시킬 수 있는 가상현실 기반의 체험형 콘텐츠는 다양한 시장에서 그 중요성이 점점 높아지고 있다. 현재 가상현실 콘텐츠들에서 사용되는 컨트롤러는 대부분 물리 기반의 동작과는 다소 차이가 있기 때문에 사용자의 몰입을 떨어뜨리는 주원인이 된다. 가상현실 컨트롤러를 이용해 상호작용을 할 때는 물리 동작과는 다른 형태이기 때문에 그 직관성이 떨어지게 되며, 이에 따라 몰입 또한 낮아지게 된다. 이를 보완하기 위해 본 논문에서는 사용자의 대칭 동작을 이용하여 아바타의 동작을 제어하는 효율적인 방법을 제안한다.

사용자가 움직이는 대로 아바타와 객체의 상호작용이 자연스럽게 이루어진다면, 사용자는 콘텐츠에 몰입 할 수 있게 되며, 만족스러운 경험을 즐길 수 있게 된다. 이러한 콘텐츠는 게임, 시뮬레이션, 체험형 콘텐츠 등 몰입감이 필요한 콘텐츠뿐만 아니라 사용자의 움직임을 이용해 애니메이션을 생성하기 때문에 반복적인 움직임이 필요한 재활 치료 등에서도 활용이 가능하다. 본 논문에서는 가상환경에서

사용자의 대칭 동작에 따라 아바타의 움직임을 데이터-드리븐 기법을 이용해 처리하는 방법을 제안한다.

II. Preliminaries

1. Related works

기존 접근법들에서는 강체, 유체, 및 변형체를 제어하기 위해 다양한 기법들을 사용하고 있지만, 대부분 지배방정식에 따라 계산이 되기 캐릭터의 움직임이 아닌 재질에 영향을 많이 받는다[1,2,3]. *Chemin et al.*은 캐릭터의 상반신 움직임을 기반으로 저글링 모션을 학습 접근법을 통해 생성했다. 이를 학습하기 위해 다양한 모션을 강화학습에 적용했지만, 실제 사람의 모션을 이용하지 않았기 때문에 결과적으로 학습된 움직임에 대한 차이가 발생한다. *Choi et al.*은 사람의 팔 움직임을 데이터-드리븐 기법을 통해 계산했지만[4], 양쪽 팔 데이터와 공 궤적을 계산하기 위해 최적화 함수를 사용하기 때문에 비효율적이며, 영상으로부터 추출된 속도에 노이즈가 많이 포함되기

때문에 시뮬레이션이 불안정해지기 경우가 발생한다. Hong *et al.*은 물리 기반 캐릭터 애니메이션 기법을 통해 발목의 움직임을 세밀하게 조정하였고, 이를 통해 캐릭터로 축구공을 보다 정확하게 전달할 수 있는 방법을 제시했다[5].

III. The Proposed Scheme

1. Algorithm overview

본 논문에서는 데이터 기반으로 사용자의 동작 데이터를 이용한 아바타의 움직임을 합성한다. 가상환경에서 사용자의 상호작용과 몰입에 초점을 맞춰 움직임과의 상호작용이 자연스럽게 합성될 수 있도록 한다. 데이터-드리븐을 이용해 저장된 움직임 데이터를 기반으로 애니메이션을 생성함으로써, 기존에 사용했던 수작업 애니메이션과 달리 이미 저장된 데이터를 사용하기 때문에 효율적인 애니메이션 생성이 가능하다.

본 논문에서는 사용자의 움직임 데이터를 얻기 위해 림모션(Leap Motion) 기기를 이용하였으며, 이를 통해 저글링 동작을 얻었다 (Fig. 1 참조). 이 과정에서 사용자의 오른손 움직임이 캡처되고, 이를 CSV 파일 형식으로 저장해 데이터화한다. 저장된 데이터와 새로운 입력 데이터간의 유사도를 계산하여 가장 유사한 모션 클립을 선택한다. 오른손 데이터만을 사용하므로, 데이터-드리븐 기법을 활용해 대칭적인 왼손 데이터를 생성한다. 이후, 오른손과 왼손 데이터에 맞춰 가상환경에서 움직이는 공을 생성해 아바타의 움직임과 동기화된 저글링 애니메이션을 합성한다.



Fig. 1. Hand position tracking using Leap Motion.

2. Motion data generation

이번 장에서는 사용자의 손 움직임을 저장하고, 움직임 데이터를 생성하는 방법에 대해 설명한다. 사용자는 림모션에서 오른손을 움직여 데이터를 얻는다. 이때, 유니티3D의 StreamWriter을 이용해 입력 받은 사용자 오른손 손목, 손바닥, 중지의 위치 좌표를 CSV파일로 저장한다. 이후, 저장된 위치 데이터를 이용해 아바타가 공을 던지고 받을 때 기준이 되는 수직속도를 계산하게 된다. 수직속도는 손목, 손바닥, 중지 각각의 Y축 변화량을 계산한 값이다. 이렇게 구한 수직속도를 기준으로 데이터를 나눠 저장한다. 수직 속도의 변화량을 그래프로 나타냈을 때 파랑색은 손목, 주황색은 손바닥, 회색은 중지를 의미한다. (Fig. 2 참조) 공을 위로 던질 때를 기준으로 수직속도가

음수일 때에는 공을 잡는 프레임이 되고, 양수일 때에는 공을 던지는 프레임이 된다.

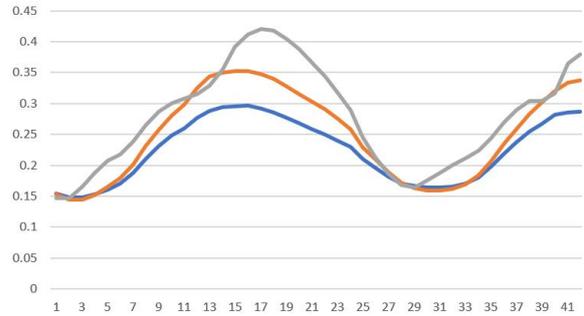


Fig. 2. A chart showing the change in vertical velocity.

3. Calculating similarity

3.1 Maximum speed frame

입력된 움직임과 사전에 저장된 움직임 데이터 사이의 유사도를 계산하기 전에 움직임에 데이터에서 잡기와 던지기 프레임에서 각각 최대 속력이 되는 프레임을 계산한 뒤 각각 두개의 구간을 선정한다.

- 첫번째 프레임~최대 속도 프레임
- 최대속도 프레임~마지막 프레임

이렇게 계산이 되면 총 4개의 구간이 생기게 된다. 이 4개의 구간에서 모션 벡터와 이동 시간을 구하여 유사도를 계산한다.

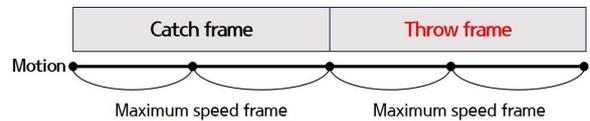


Fig. 3. Structure of maximum speed frame.

Choi *et al.*은 유사도를 구하는 식을 다음과 같이 제시했다[4](수식 1 참조) 식의 첫 번째 항은 모션 벡터와 이동시간의 유사도를 계산한다. 두 번째 항은 회전 유사도를 계산한다.

$$D = w_1 D_M(M_i, M_j) + w_2 D_T(M_i, M_j) \quad (1)$$

3.2 Motion vector and travel time similarity

이번 장에서는 입력으로 사용된 움직임 데이터 간의 벡터와 이동시간의 유사도를 계산하는 방법에 대해 설명한다. 앞서 최대속력을 이용해 계산한 4개의 구간을 이용하여 유사도를 계산한다. 유사도는 각 구간에서 입력으로 사용한 움직임 데이터 간의 차이를 이용해 계산한다. 입력으로 사용한 움직임의 이동 시간에서 움직임 데이터의 이동시간을 뺀 시간 차이를 계산한다. 이후 4개의 구간을 더한 뒤 가중치를 곱하면 첫 번째 항인 D_M 이 계산된다 (수식 2 참조). 이 과정을 오른손의 손목, 손바닥, 중지 데이터에 따라 각각 반복한다.

$$D_M(M_i, M_j) = w_v \sum_{i=1}^4 \|v_i - v'_i\| + w_d \sum_{i=1}^4 |d_i - d'_i| \quad (2)$$

여기서 v_i 와 d_i 는 다음과 같이 계산한다 : $v_i = p_{i+1} - p_i$, $d_i = t_{i+1} - t_i$. 좀 더 자세한 설명은 이전 논문을 읽어보길 권장한다 [4].

3.3 Rotation similarity

이번 장에서는 입력으로 사용한 움직임 데이터간 위치와 방향에 관계없이 포즈의 회전 유사도를 계산하는 방법에 대해 설명한다. 이 과정은 입력으로 사용한 움직임 데이터간의 회전 유사도를 비교해 가장 작은 값을 계산하는 과정이다 (수식 3 참조).

$$D_T(M_i, M_j) = \min_{\theta, x_0, z_0} \sum_k w_k \| p_{i,k} - T_{\theta, x_0, z_0} p_{j,k} \|^2 \quad (3)$$

여기서 θ 는 각도를 나타내는 수식이며 다음과 같이 계산한다 :

$$\theta = \arctan \frac{\sum_i w_i (x_i' z_i' - x_i' z_i') - (\bar{x} z' - \bar{x}' z)}{\sum_i w_i (x_i x_i' - z_i z_i') - (\bar{x} x' - \bar{x}' z)}$$

또한 x_0 와 z_0 는

다음과 같이 계산한다 : $x_0 = (\bar{x} - \bar{x}' \cos \theta - \bar{z}' \sin \theta)$, $z_0 = (\bar{z} - \bar{z}' \cos \theta - \bar{x}' \sin \theta)$. 위 수식에서 T_{θ, x_0, z_0} 는 Y축 대해 θ 만큼 회전하고 x_0, z_0 만큼 이동시키는 선형변환이다. 또한 $p_{i,k}$ 와 $p_{j,k}$ 는 입력 움직임 데이터 중 i 번째 프레임에서 생성된 포인트와 j 번째 포인트이다. 본 논문에서는 위 수식을 통해 계산된 회전 차이에서 가장 작은 값을 찾는다. 이 과정을 손목, 손바닥, 중지 데이터에 따라 각각 반복한다.

4. Create virtual objects with data-driven

4.1 Create virtual hand and ball

본 논문에서는 오른손 데이터만 저장하기 때문에 양손 저글링 애니메이션을 적용하기 위해서는 왼손의 움직임을 실시간으로 합성해야 한다. 저글링 동작 시 왼손은 오른손의 반대로 회전하기 때문에 (Fig. 4 참조) 유니티3D의 StreamReader를 이용해 모션 데이터를 반대로 읽어와 생성하고, 자연스러운 저글링 움직임이 나타나도록 시간 차이를 두어 데이터를 합성하였다. 또한 왼손은 합성 데이터이기 때문에 그 속도는 오른손 데이터의 속도에 의존하는 형태로 나타난다. 오른손과 왼손 데이터를 이용해 아바타의 양 손을 이동하는 가상의 공을 생성한다. 사용자가 공을 던지기 전과 후에는 아바타가 공을 잡고 있기 때문에 공은 아바타의 손과 함께 움직이게 된다. 이 과정은 Boolean 변수를 이용해 공이 어느 손에 있을 것인지 결정한다.



(a) Right rotation direction (b) Left rotation direction

Fig. 4. Virtual hand formation.

4.2 Virtual ball trajectory based on parabolic motion

본 논문에서의 가상 공의 움직임을 결정하기 위해 움직임 데이터에서 수직속도가 상승할 때 반대 손의 위치로 공이 포물선 운동을 하도록 설정한다. 손의 위치를 실시간으로 입력받은 뒤, 시작 위치와 도착 위치를 자동으로 결정하도록 포물선 운동을 계산한다. 시간과 높이에 따른 사인(Sine)을 계산해 포물선값에 따라 공이 포물선으로 이동하도록 한다. Fig. 5는 오른손과 왼손의 위치를 기반으로 공의 궤적을 시각화한 결과이다.

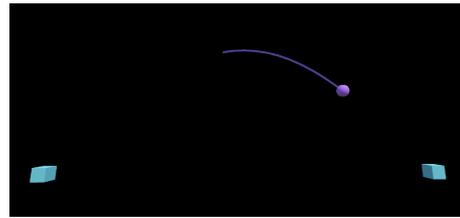


Fig. 5. Virtual ball formation

5. Controls to improve immersion

본 논문에서는 아바타의 자연스러운 관절 움직임을 표현하기 위해 아바타에 IK(Inverse kinematics)를 적용한다. IK는 역운동학으로, 목표 위치에 따라 관절 회전을 자동으로 계산하게 된다.



Fig. 6. Creating scenes with juggling motion control.

또한 애니메이션에 손쉽게 원하는 장면을 제작하기 위해 애니메이션 재생 시 변수를 이용해 손의 속도와 손의 간격을 제한한다.

IV. Conclusions

본 논문에서는 가상환경에서 캐릭터의 저글링 움직임 제어를 효율적으로 처리하기 위한 데이터-드리븐 기법을 제안한다. 사용자의 움직임을 저장해 프리프로세싱으로 수직 속도에 따라 움직임 데이터를 저장한다. 사용자의 움직임을 입력받으면 데이터 유사도를 비교해 입력 움직임과 유사한 애니메이션을 합성한다. 또한 오른손의 데이터를 이용해 가상 왼손의 궤적을 생성하고, 양손의 데이터로 아바타가 저글링을 할 수 있는 가상의 공을 생성하는 결과를 보여준다. 이후, 아바타에 IK를 적용해 아바타의 자연스러운 관절 움직임을 구현한다. 아바타가 자연스럽게 공과 상호작용하며 저글링형태로 움직임을 자제를 합성하는 것을 확인할 수 있다 (Fig. 7 참조).



Fig. 7. Virtual juggling motion with our method.

향후, 공이 포물선 운동을 할 때, 실시간으로 아바타의 손이 움직여 도착 위치가 변경되더라도 변경된 도착 위치로 보간하여 공이 움직이도록 방법을 확장할 계획이다.

REFERENCES

- [1] C. Twigg and D. James, "L.: Backward steps in rigid body simulation," *ACM Trans. Graph.*, vol. 27, 08 2008. DOI: 10.1145/1399504.1360624.
- [2] R. Fattal and D. Lischinski, "Target-driven smoke animation," *ACM Transaction on Graphics*, vol. 23, 06 2004. DOI:10.1145/1015706.1015743.
- [3] J. Barb[~]i and J. Popov[~]i, "Real-time control of physically based simulations using gentle forces," *ACM Transactions on Graphics*, vol. 27, 12 2008. doi: 10.1145/1409060.1409116.
- [4] J. I. Choi, S. J. Kang, C. H. Kim, and J. Lee, "Virtual ball player," *The Visual Computer*, vol. 31, 05 2015. doi: 10.1007/s00371 -015-1116-9.
- [5] S. Hong, D. Han, K. Cho, J. S. Shin, and J. Noh. 2019. Physics-based full-body soccer motion control for dribbling and shooting. *ACM Trans. Graph.* vol 38, no 4, Article 74 (July 2019), pp. 1-12. DOI:10.1145/3306346.3322963.