# 동적계획법을 활용한 자동 키프레임 추출 알고리즘에 대한 연구

황주헌, 박예승, 장민규, 이정행, 동호석, 이상훈
연세대학교
{consecrated.h, pys940617, jmg1002, leedoright, hstong09, slee}@yonsei.ac.kr

# Implementation of Keyframe extraction using Dynamic Programming

Hwang Juheon, Park Yeseung, Jang Mingyu, Lee Jeonghaeng, Tong Hoseok, Lee Sanghoon
Yonsei Univ.,

요 약

키프레임 추출은 비디오 디자이너들과 개발자들에게 있어 중요한 업무 중 하나이다. 이를 위해 다양한 알고리즘들이 제시되어 왔고 그 중 가장 최첨단을 달리는 알고리즘은 Salient Pose라는 알고리즘이다. 그러나, 이 알고리즘은 높은 시간 복잡도를 가지며, 키프레임 개수를 매개변수로 필요로 한다는 단점이 존재한다. 본 논문에서는 주성분분석과 유사도전파 알고리즘을 활용한 두 알고리즘을 추가하여 앞선 알고리즘을 개선하였다. 결과적으로, 기존 알고리즘보다 더 빠르고, 키프레임 개수까지 자동으로 찾아주는 알고리즘을 제안하였다.

## Ⅰ. 서 론

최근 휴먼 애니메이션은 영화, 게임, AR/VR 등 많은 산업 분야에서 사용된다. 하지만 휴먼 애니메이션은 많은 차원과 많은 프레임 속도를 요구하기 때문에 재활용을 하는 것이 어렵다. 따라서 재활용을 위해 압축, 저장, 전송, 검색, 복원 등의 기술이 필요한데, 이를 위해 키프레임 추출에 대한 연구가 많이 이뤄지고 있다. 여기서 키프레임 추출이란 비디오, 동작 등의 시계열 데이터에서 필요한 부분만 취하는 것을 의미한다.

키프레임을 통해 휴먼 애니메이션의 전체 데이터를 확인하지 않고도 어떠한 동작을 취하고 있는지 확인할 수 있다. 이러한 특성으로 인하여, 키프레임 추출은 비디오 디자이너와 개발자들의 효율적인 작업을 돕는다.

키프레임 추출에서 가장 좋은 성능을 나타내는 Salient Pose(SP)[1]는 높은 시간복잡도와 키프레임 개수를 매개변수로 한다는 한계점을 갖는다. 이러한 한계점을 해결하기 위해 본 논문은 Salient Pose를 바탕으로 주성분 분석(PCA)[2]과 유사도 전파 알고리즘 Affinity Propagation(AP)[5]을 이용한 시계열 휴먼 스켈레톤의 자동 키프레임 추출 방법을 제안한다.

## Ⅱ. 본론

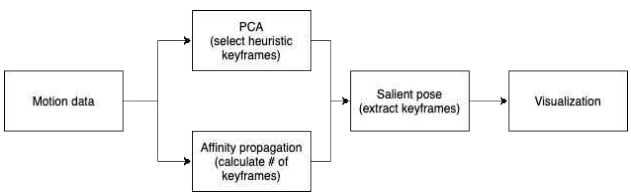



그림 1 제안된 자동 키프레임 추출 알고리즘

기존의 동적계획법을 사용한 SP의 한계를 해결하기 위해, 그림 1과 같은 자동 키프레임 추출 알고리즘을 제안한다. 시계열 휴먼 스켈레톤 데이터(CMU Motion Capture Dataset)를 입력으로 하여 이를 주성분 분석(PCA)과 AP에서 처리하여 각각 휴리스틱 키프레임과 최적의 키프레임의 개수를 찾는다.

본 논문에서 사용한 시계열 휴먼 스켈레톤 데이터는 CMU Graphics Lab에서 제공하는 데이터로, 4MP의 해상도로 초당 120 프레임을 촬영할 수 있는 vicon infrared MX-40 카메라 12대를 사용하여 취득되었다. 스켈레톤은 38개의 관절로 구성이 되어있으나, 본 논문에서는 15개의 관절만을 사용하였다.

주성분 분석(PCA)은 차원을 줄이는 방법 중 하나로, 높은 차원의 데이터를 주성분이라고 하는 낮은 차원으로 사영한다. 본 논문에서는 첫 번째 주성분에서 가장 편차가 큰 값을 휴리스틱 키프레임으로 선정하여 많은 프레임을 가지는 데이터를 200 프레임 단위로 나누도록 하였다.

AP는 비지도 군집화 방법 중 하나로, 군집의 개수를 지정할 필요가 없고, 군집들이 공간을 동일한 크기로 분할하는 것을 강제하지 않는다. 이는 데이터 간의 유사도를 계산하여 군집을 대표할 데이터들을 찾는다. 이때 AP 행렬의 대각선의 값을 다른 전체 값의 최솟값, 중간값, 75% 값을 넣는 것을 통해 키프레임의 개수를 조절하였다. 본 논문에서는 AP 알고리즘을 통해 얻은 최적의 키프레임 개수를 통해 데이터를 대표하는 군집들의 개수를 자동으로 얻을 수 있게 하였다.

## Ⅲ. 분석

본 논문에서 제안한 알고리즘에 대한 실험은 Ryzen 3700X 8-core, 32GB RAM, Ubuntu 18.04의 환경에서 진행되었다. 본 논문은 실험의 결과를 보여주기 위해, 우리는 키프레임을 빨간색으로, 그렇지 않은 것을 회색으로 나타내었다. 그리고 각 키프레임에는 나타낸 스켈레톤 위에 해당 키프레임의 색인을 달았다. 마지막으로 더 큰 색인을 가지는 키프레임이 더 진한 빨간색을 나타내도록 하였다.

Run around in a circle (759 frames)

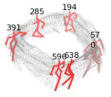
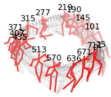
| Algorithm | SP | SP + PCA | SP + PCA + AP |
|---|---|---|---|
| Motion |  |  |  |
| Computation time | 262.3s | 68.9s | 268.5s |
| Keyframes | [8]<br>0, 60, 161, 366, 472, 557, 695, 758 | [8]<br>0, 110, 190, 371, 474, 570, 700, 758 | [17]<br>0, 25, 101, 145, 190, 219, 277, 315, 371, 407, 435, 513, 570, 636, 671, 711, 758 |

그림 2 알고리즘을 통해 얻은 결과 비교

먼저 기존의 SP와 SP+PCA, SP+PCA+AP의 동작 시간과 키프레임의 개수 및 그 형태를 그림2와 같이 비교하였다. 입력된 데이터는 여러번 원을 그리며 걷는 데이터이다. SP와 SP+PCA의 경우 키프레임의 개수를 직접 정하기 때문에 여러번 원을 도는 것을 시각화를 통해 알 수 없는 반면, SP+PCA+AP의 경우 최적의 키프레임 개수를 구할 수 있어 시각화를 통해 이를 알 수 있다. 이때 AP의 경우 중간값을 사용하여 키프레임의 개수를 구하였다.

Two jumps (175 frames)

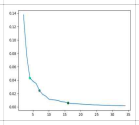
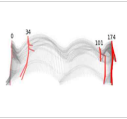
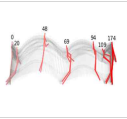
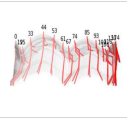
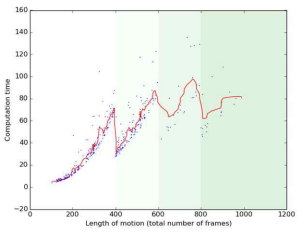
| Category | Cost - # of Keyframe graph | 'min' for AP | 'median' for AP | 'three-quarter' for AP |
|---|---|---|---|---|
| Result |  |  |  |  |
| Cost | - | 0.043 | 0.025 | 0.006 |
| Keyframes | - | [4]<br>0, 34, 101, 174 | [7]<br>0, 20, 48, 69, 94, 109, 174 | [16]<br>0, 19, 25, 33, 44, 53, 61, 67, 74, 85, 93, 100, 106, 116, 130, 174 |

그림 3 AP 알고리즘에서 최솟값, 중간값, 75% 값을 사용하여 얻은 키프레임 추출 결과 및 cost - 키프레임 개수 그래프

그림 3을 통해 AP 알고리즘이 적절한 키프레임의 개수를 자동으로 찾아주는지 확인하였다. 그림3의 Cost - 키프레임 개수 그래프에서 왼쪽 점부터 차례대로 최솟값, 중간값, 75% 값을 사용한 지점이다. 먼저 최솟값의 경우 추출된 키프레임의 개수가 적어 모든 움직임을 표현하기 부적절한 것을 확인할 수 있다. 다음으로 75% 값의 경우 키프레임 개수가 많아 효율적으로 압축했다고 볼 수 없다. 중간값의 경우 두 번의 점프를 키프레임을 통해 확인할 수 있으며 도약하기 전과 착지하는 순간을 확인할 수 있다. 따라서 AP 알고리즘의 중간값을 통해 적절한 키프레임의 개수를 구할 수 있다는 것을 확인하였다.



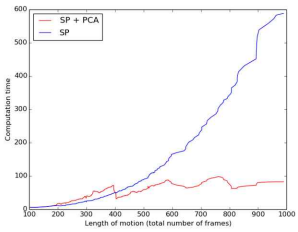



그림 4 SP+PCA의 프레임수-동작 시간 그래프(왼)와 SP와 SP+PCA의 프레임수-동작 시간 비교 그래프(오)이다.

앞서 설명한 것과 같이, 많은 프레임을 가지는 데이터를 200 프레임 단위로 나누어 키프레임을 추출하였기 때문에, 그림3의 왼쪽 그래프와 같이 200 프레임 단위마다 동작 시간이 줄어드는 것을 확인할 수 있다. 오른쪽의 그래프는 SP와 SP+PCA의 동작 시간을 비교한 그래프로 주성분 분석으로 휴리스틱 키프레임을 얻는 방식이 동작 시간을 줄이는 것에 매우 효과적임을 확인할 수 있다.

## Ⅳ. 결론 및 향후 연구 방향

본 논문은 동적계획법을 이용한 Salient Pose를 기반으로 하여 키프레임 추출 알고리즘의 시간복잡도를 줄이고, 최적의 키프레임 개수를 자동으로 찾아 활용하기 쉽게 구현되었다. 향후 추가적인 연구로 사용자 주관적 평가를 통해 다양한 동작에서의 최적의 키프레임 개수를 찾는 평가 방법을 통해 시각적으로 데이터 손실이 적은 키프레임 추출 알고리즘에 대한 연구로 확장시킬 수 있을 것이라 생각한다.

## ACKNOWLEDGMENT

이 성과는 2020년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (No. 2020R1A2C3011697)

## 참 고 문 헌

[1] Roberts, R/, Lewis, J. P, Anjyo, K., Seo, J., & Seol, Y. (2018). Optimal and interactive keyframe selection for motion capture. In SIGGRAPH Asia 2018 Technical Briefs (pp. 1-4).

[2] Miura, T., Kaiga, T., Shibata, T., Katsura, H., Tajima, K., & Tamamoto, H. (2014). A hybrid approach to keyframe extraction from motion capture data using curve simplification and principal component analysis. IEEJ Transactions on Electrical and Electronic Engineering, 9(6), 697-699.

[3] Liu, X. M., Hao, A. M., & Zhao, D. (2013). Optimization-based key frame extraction for motion capture animation. The visual computer, 29(1), 85-95.

[4] Zhang, Q., Zhang, S., & Zhou, D. (2014). Keyframe extraction from human motion capture data based on a multiple population genetic algorithm. Symmetry, 6(4), 926-937.

[5] Frey, B. J., & Dueck, D. (2007). Clustering by passing messages between data points. science, 315(5814), 972-976.

[6] Bulut, E., & Capin, T. (2007, June). Key frame extraction from motion capture data by curve saliency. In Computer animation and social agents (p. 119).

[7] Comaniciu, D., & Meer, P. (1999, September). Mean shift analysis and applications. In Proceedings of the Seventh IEEE International Conference on Computer Vision (Vol. 2, pp. 1197-1203). IEEE.

[8] BEster, M., Kriegel, H. P., Sander, J., & Xu, X. (1996, August). A density-based algorithm for discovering clusters in large spatial databases with noise. In Kdd (Vol. 96, No. 34, pp. 226-231).