영상에서 물체 추적 알고리즘별 성능 비교 연구

이동규, 한동석*
경북대학교 전자공학과
jasmindoe@knu.ac.kr, *dshan@knu.ac.kr

요 약

본 논문은 영상에서 다양한 방식의 추적 알고리즘들의 성능을 실험, 비교하였다. 물체 추적 알고리즘은 물체의 추적 정확도도 중요하지만, 실시간에서는 프레임 성능도 중요하다. 딥러닝 이전의 기존 추적 알고리즘은 물체의 추적을 위해 매 프레임마다 많은 계산량이 필요하다. 딥러닝을 사용하는 추적 알고리즘은 오프라인 방법을 사용한다. 이 방법은 기존의 방법보다는 상대적으로 적은 계산량이 요구된다. 그렇기 때문에 이전의 추적 알고리즘들보다 성능이 낮은 장치에서 더 좋은 성능을 가질 수 있다. 또한 딥러닝을 사용한 추적 알고리즘이 물체 간의 겹침, 추적 물체가 다른 물체에 가려짐, 물체 추적 실패 등 다양한 상황에서 더 좋은 성능을 보여준다. 본 논문에서는 기존의 알고리즘과 딥러닝을 사용한 물체 추적 알고리즘의 프레임 성능과 특정 상황에서 물체 재추적이 가능한지를 테스트하였다.

Ⅰ. 서 론

컴퓨터 비전의 다양한 분야에서 딥러닝 기술들이 사용되고 있다. 위치 추적 기술, SLAM, 객체 검출 및 추적 등에서 딥러닝을 사용하여 큰 성능 향상을 보여주고 있다. 물체 추적 또한 딥러닝을 사용하는 방식이 뛰어난 성능을 보인다. 기존의 물체 추적 알고리즘은 주로 온라인 방식을 사용한다. 온라인 방식은 매 프레임마다 물체의 위치 파악과 추적을 위해 큰 계산량을 필요로 한다. 기존의 방식에는 Boosting, TLD(Tracking, Learning and Detection), CSRT(Channel and Spatial Reliability Tracker), MedianFlow 등이 있다. 이 방식들은 실시간 물체 추적을 위해 높은 기기 사양을 요구한다. 또한 물체가 다른 물체에 가려진 후 추적이나 추적 실패 후 재추적에서 문제점을 가진다. 이러한 문제점을 해결하기 위해 딥러닝을 사용한 오프라인 방식의 물체 추적 알고리즘을 사용할 수 있다. 오프라인 방식은 추적 알고리즘의 네트워크를 미리 학습시키기 때문에 보다 적은 계산량이 요구된다. 그러므로 낮은 사양의 기기에서도 좋은 실시간 프레임 성능을 가질 수 있다. 또한 기존의 방식보다 다양한 상황에서 더 좋은 추적 성능을 보여준다. 딥러닝을 사용한 방식 중 CNN을 사용한 GOTURN(Generic Object Tracking Using Regression Networks) 트래킹 방식[1]과 GOTURN보다 더 좋은 성능을 보여주는 딥러닝 방식의 추적 알고리즘에는 두 개의 CNN을 사용한 Siamese 트래킹 방식이 있다.[2]. 더 나아가 Siamese 네트워크에 RPN(Reigon Proposal Network)을 더해 정확도를 높인 SiameseRPN 트래킹이 있다.[3] 본 논문은 다양한 방식의 물체 추적 알고리즘들 사이에 FPS(Frame Per Second)성능과 특정 상황에서 재추적 성능을 비교하였다.

Ⅱ. 테스트 구성과 실험 결과

추적 알고리즘은 단일 대상에 대해 성능을 테스트하였다. 다양한 방향과 각도로 움직이는 얼굴과 다른 물체에 얼굴이 가려지거나, 겹치는 상황에 대하여 다양한 방식들의 추적 알고리즘들을 실험했다. 본 실험의 주요 목적은 물체가 다른 물체에 가려지는 상황에서 재추적 가능 여부와 추적 알고리즘에 대한 FPS의 성능을 중점적으로 테스트하였다. 기존 방식의 추적 알고리즘은 Boosting, TLD, CSRT를 실험하였고, 딥러닝을 사용한 방식은 GOTURN, Siamese RPN을 사용하였다.

실험에 사용된 장비는 CPU i7-9700k,, GPU 2080super와 로지텍 Webcam 9000을 사용하였다.

웹캠을 사용하여 지정된 방향과 각도로 움직이는 사람의 얼굴을 30초간 녹화하였다. 그리고 특정 상황에서 재추적 여부를 테스트하기 위해 얼굴이 다른 물체에 겹쳐지고 다시 보이는 상황 또한 녹화하였다.

딥러닝 방식의 추적 알고리즘을 위해서 미리 학습된 네트워크들을 사용하였다. GOTURN은 ResNet-50을 기반 네트워크로 사용하였고, Siamese RPN은 ResNet-50과 MobilenetV2 총 2개의 네트워크를 각각 사용하여 테스트하였다.

표. 1. 추적 물체 겹침이나 추적 실패 후 물체 재추적 가능성.

	물체 겹침 후 재추적	추적 실패후 재추적
Boosting	불가능	불가능
TLD	가능	가능
CSRT	가능*	불가능
GOTURN	가능*	가능*
SiameseRPN (Resnet-50)	가능	가능
SiameseRPN (MobilenetV2)	가능	가능

* 는 항상 재추적에 성공하는 게 아닌 간헐적 성공을 의미한다.

Boosting은 추적하던 물체를 놓치고 나면 재추적이 불가능하였다. TLD의 경우 물체를 놓치면 최대한 유사한 물체에 대해 재추적을 시도하면서 원래 추적하던 물체를 재추적할 수 있었다. 하지만 원래 추적하던 물체가 아닌 다른 물체를 추적하는 등의 정확성이 떨어지는 문제점을 보였다. CSRT는 추적 실패 후 재추적에는 실패하였지만, 물체가 겹쳐지고 난 뒤 추적은 어느 정도 성공하였다.

딥러닝 방식을 사용하는 GOTURN과 SiameseRPN은 모두 특정 상황에서 재추적이 가능하였다. GOTURN의 경우 TLD와 CSRT보다는 높은 정확성과 함께 재추적 성공을 보여주었다. SiameseRPN의 경우 테스트한 추적 알고리즘들 중에서 가장 높은 재추적 성공을 보여주었다.

표. 2. 추적 알고리즘별 비실시간 및 실시간 영상에 따른 FPS 성능.

	비실시간 평균 FPS	실시간 평균 FPS
Boosting	21	30
TLD	24	23
CSRT	32	30
GOTURN	97	97
SiameseRPN (Resnet-50)	44	42
SiameseRPN (MobilenetV2)	82	83

비실시간과 실시간 영상에서는 GOTURN 방식이 가장 높은 평균 FPS를 가지며, SiameseRPN의 MobilenetV2 순으로 좋은 성능을 보였다. SiameseRPN은 하나의 추적 물체 입력에 대해 전체 이미지에서 윈도우 슬라이딩(window sliding)과 유사한 방식으로 물체를 찾아 추적하는 방식이고, GOTURN은 이전 프레임에서의 물체 위치 입력과 현재 프레임에서 물체 위치 입력만을 비교하는 방식이다. 그러므로 같은 ResNet-50을 사용하였지만 GOTURN 방식이 SiameseRPN보다 높은 평균 FPS을 가진다. 전체적으로 딥러닝 방식의 추적 알고리즘들이 기존 방식의 추적 알고리즘들보다 높은 FPS를 보였다.

Ⅲ. 결론

본 논문은 특정 상황에서 물체 재추적과 알고리즘별 FPS 성능을 테스트하였다. 딥러닝을 사용한 추적 알고리즘들이 기존 방식의 추적 알고리즘들보다 특정 상황에서 물체 재추적이 가능하면서 높은 평균 FPS를 보여주었다. 딥러닝 알고리즘의 뼈대 네트워크 구조와 학습방식 등의 연구를 통하여 더욱 높은 FPS 성능과 물체 추적 성능을 보여줄 수 있을 것이다.

ACKNOWLEDGMENT

이 연구는 2019년도 산업통상자원부 및 산업기술평가관리원(KEIT) 연구비 지원에 의한 연구임('20003519')

참 고 문 헌

[1] Held D., Thrun S., Savarese S, "Learning to Track at 100 FPS with Deep Regression Networks", European Conference on Computer Vision, pp. 749-765, Oct. 2016.

[2] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. Torr, "Fully-convolutional siamese networks for object tracking", European conference on computer vision, pp. 850-865, Oct. 2016

[3] Li, Bo & Wu, Wei & Wang, Qiang & Zhang, Fangyi & Xing, Junliang & Yan, Junjie, "SiamRPN++: Evolution of Siamese Visual Tracking With Very Deep Networks", Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4282-4291, 2019