Journal of KIIT. Vol. 19, No. 10, pp. 1-9, Oct. 31, 2021. pISSN 1598-8619, eISSN 2093-7571
http://dx.doi.org/10.14801/jkiit.2021.19.10.1

SORT와 DeepSORT의 혼합을 이용한 실시간 다중객체 추적

양수진*[1], 정인화*[2], 강동화*[3], 백형부**

Real-Time Multi-Object Tracking using Mixture of SORT and DeepSORT

Sujin Yang*[1], Inhwa Jung*[2], Donghwa Kang*[3], and Hyeongboo Baek**

이 논문은 인천대학교 2021년도 자체연구비 지원에 의하여 연구되었음.

요 약

DeepSORT(Deep Simple Online and Real-time Tracking)는 SORT의 정확도를 개선한 다중 객체 추적 기술로써, SORT에서 연속된 프레임간의 객체들의 관계를 판별할 때 CNN(Convolution Neural Network)의 특징 맵(Feature map)을 이용한다는 특징이 있다. 이러한 특징은 정확도를 비약적으로 개선시킨다는 장점이 있지만, 필요한 추가 컴퓨팅 연산의 양이 많아 FPS(Frame Per Second)를 떨어뜨린다는 단점이 있다. 따라서 자율주행 자동차와 같은 실시간 시스템의 경우 DeepSORT보다는 SORT가 더욱 적합하다고 간주되어 왔다. 본 논문에서는 기존의 Deep SORT의 단점을 제거하기 위하여 DeepSORT에 SORT의 기술을 혼합시키는 기법을 제안한다. 다중 객체 추적의 성능을 위해 사용 널리 사용되는 데이터 셋인 MOT-16을 대상으로 하는 실험 결과, 제안 된 방법은 정확도에 대한 손실에 거의 없이 FPS를 비약적으로 상승시키는 것을 확인 할 수 있었다.

Abstract

Deep Simple Online and Real-time Tracking(DeepSORT) is a multi-object tracking technology that improves the accuracy of SORT, and has the characteristic of using a feature map of Convolution Neural Network(CNN) when determining the relationship between objects between consecutive frames in SORT. This feature has the advantage of dramatically improving the accuracy, but has a disadvantage in that the frame per second(FPS) is lowered due to the large amount of additional computing operations required. Therefore, for real-time systems such as autonomous vehicles, SORT has been considered more suitable than deep SORT. In this paper, we propose a method of mixing SORT technology with DeepSORT to remove the shortcomings of the existing DeepSORT. As a result of the experiment targeting MOT-16, a widely used data set used for multi-object tracking performance, it was confirmed that the proposed method dramatically increased the FPS with little loss in accuracy.

Keywords

multi-object tracking, real-time system, real-time tracking

* 인천대학교 컴퓨터공학부 학사과정(*[1,2] 공동 1저자)
- ORCID[1]: https://orcid.org/0000-0003-3960-3442
- ORCID[2]: https://orcid.org/0000-0002-3662-9463
- ORCID[3]: https://orcid.org/0000-0003-2183-3290
** 인천대학교 컴퓨터공학부 조교수(교신저자)
- ORCID: https://orcid.org/0000-0001-9518-3556

· Received: Jul. 14. 2021, Revised: Aug. 18, 2021, Accepted: Aug. 21, 2021
· Corresponding Author: Hyeongboo Baek
Incheon National University, Republic of Korea
Tel.: +82-32-835-8493, Email: hbbaek359@gmail.com

Ⅰ. 서 론

다중 객체 추적 기술은 비디오 영상에 존재하는 사람, 자동차와 같은 객체들의 위치를 연속적으로 추적하는 기술로써, 자율주행 시스템, 무인 감시 시스템(CCTV, Closed-Circuit Television) 등과 같은 다양한 분야에서 활용되고 있다[1]-[3]. 다중객체 추적 기술은 먼저 영상의 각 프레임(단일 이미지)에서 각 객체들의 위치를 파악하는 단계인 객체 탐지(Object detection) 단계와 연속된 두 프레임에 존재하는 객체들의 관계를 파악하여 각 객체들의 위치 변화를 추적하는 객체 추적(Object tracking) 단계로 나눌 수 있다.

현대의 객체 탐지 기술은 인공지능의 발전과 함께 비약적인 발전을 이루었다. 딥러닝을 이용하여 하나의 이미지에서 여러 객체를 탐지하는 방법으로써 R-CNN(Regions with Convolution Neural Network features)[4], Fast R-CNN[5], YOLO(You Only Look Once)[6] 등의 방법이 제안되었다. 각 기술마다 탐지성능과 속도 측면에서 장단점이 존재하는데, 예를 들어 YOLO의 경우 정확도 보다는 탐지 속도에 강점을 가지는 기법이다. 이미지를 여러 개의 그리드 영역으로 분할 한 후, 각 그리드 영역을 CNN 기법을 이용하여 학습 된 이미지 중 어떠한 이미지에 속하는지 분류하게 된다. 마지막으로 이를 이용하여, 각 그리드들을 통합하여 원본 이미지에서 객체가 존재하는 위치와 그 객체의 크기를 도출하게 된다.

객체 추적 기술 또한 매우 빠른 속도로 발전하고 있는데, 대표적으로 FairMOT(Fair Multi-Object Tracking)[7], SORT[8], DeepSORT[9] 등의 기법들이 제안되었다. 예를 들어, SORT의 경우 컴퓨팅 연산의 양이 많지 않아 속도 측면에서 빠르다는 장점이 있다. 앞서 묘사한 YOLO등의 기법을 통해 각 프레임의 객체들을 탐지하고, 이 결과를 이용하여 연속된 프레임간의 객체들이 겹치는 영역의 크기를 계산한다. 겹치는 영역이 일정이상 일 경우 같은 객체로 간주하게 된다. 더욱 자세한 설명은 다음 섹션에서 다룬다.

한편, 현대의 드론, 자율주행자동차의 같은 시스템은 인공지능을 탑재한 시스템인 동시에 주어진 작업을 정해진 시간 내에 끝마쳐야 하는 실시간 시스템(Real-time system)이다[10][11]. 예를 들어, 자율주행자동차의 경우 보행자의 위치를 탐지하고 정해진 시간 내에 브레이크 시스템이 동작하지 않을 경우 큰 인명 피해를 야기할 수 있다. 자율주행자동차의 카메라, Lidar(Light wave detection and ranging), Rader와 같은 물체를 감지하는 센서들은 정해진 주기(Period)를 가지며 반복적으로 센싱 작업을 수행하게 되는데, 이러한 작업들이 마감시간(Deadline)이라고 불리는 정해진 시간 내에 끝마치지 못할 경우 사람, 자동차와 같은 물체를 마감시간 내에 감지하지 못하게 되어 큰 인적, 물적 피해가 야기 되게 된다[12]-[15].

따라서 실시간 시스템에서는 하나의 태스크가 수행하는 반복적인 작업의 정확도도 중요하지만, 각 작업의 마감시간 준수여부가 훨씬 중요하다. 다중객체 추적 기술 역시 자율주행자동차에서 중요한 역할을 하는데, 주행 중에 차량의 전방에 위치하는 차량들의 주행 진로를 추적하거나, 횡단보도에서 보행자의 진로를 파악하는데 쓰인다. 이러한 다중 객체 추적 기술의 성능을 평가할 때 대부분 MOTA (Multiple Object Tracking Accuracy), MOTP(Multiple Object Tracking Precision)[16]와 같은 성능지표를 사용하지만, 실시간 시스템과 같이 각 작업이 정해진 시간 내에 동작하기를 요구하는 시스템에서는 FPS와 같은 성능지표가 더욱 중요한 성능지표이다.

Deep SORT는 SORT의 정확도를 개선하기 위해 CNN의 특징 맵을 사용하여, 연속 된 프레임간의 객체들의 관계를 판별한다. 해당 인공지능 기법은 정확도를 개선시킨다는 장점이 있는 반면, 추가적인 컴퓨팅 연산을 요구하여 FPS를 감소시킨다는 단점도 존재한다. 이러한 특징 때문에 SORT가 Deep SORT보다 자율주행 자동차와 같은 실시간 시스템에 더욱 적합한 기법으로 인식되어 왔다.

본 논문에서는 Deep SORT의 단점을 보완하기 위하여 Deep SORT와 SORT의 기술을 혼합시키는 새로운 기법을 제안한다. 성능의 검증을 위해 다중객체 추적의 성능을 위해 사용 널리 사용되는 데이터 셋인 MOT-16을 사용하였고, DeepSORT, SORT, DeepSORT에 일정 비율의 프레임에 추적 기술을 적용시키지 않는 기법을 비교대상으로 선정하였다. 해당 실험을 통해, 제안 된 방법이 정확도에 대한 손

실에 거의 없이 FPS를 비약적으로 상승시키는 것을 확인 할 수 있었다.

II. 기존 연구

2.1 딥러닝을 이용한 객체 탐색

딥러닝은 기계학습 알고리즘의 한 종류로, 많은 수의 층을 쌓아 학습을 진행하는 신경망 네트워크이다. 이러한 딥러닝의 한 종류로 CNN이 있다. CNN은 주로 이미지와 같은 영상을 처리하는 딥러닝의 한 종류이다. 크게 데이터의 특징을 추출하는 컨볼루션 레이어와 데이터의 차원을 줄여주는 풀링 레이어로 구성되어 있다. 이러한 CNN은 지금까지 다양한 형태로 발전되어 왔으며, 단순한 이미지 분류를 넘어 현재는 객체 탐지와 분할(Segmentation) 등 다양한 영역에서 많은 영향력을 발휘하고 있다.

CNN을 기반으로 하는 객체 탐색에서 잘 알려진 것 중 하나는 2014년에 발표된 R-CNN이다. R-CNN은 Selective Search 알고리즘을 이용해 영상에서 객체가 있을 만한 곳을 고른 후 CNN을 통해 해당 지역에 객체가 존재하는지 판단하는 방식으로 작동한다. 처음 R-CNN이 발표된 이후 현재까지 다양한 방식으로 발전해왔고, 그 결과 Fast R-CNN, Faster R-CNN 등 다양한 CNN 기반 객체 탐색 모델들이 등장하게 되었다. 하지만 이러한 R-CNN 종류의 방식들은 정확도는 비교적 높지만, 속도가 많이 느리기에 이와는 다른 방향으로 빠른 속도를 자랑하는 YOLO와 같은 기법들도 제안되었다.

2.2 딥러닝과 칼만 필터를 이용한 객체 추적

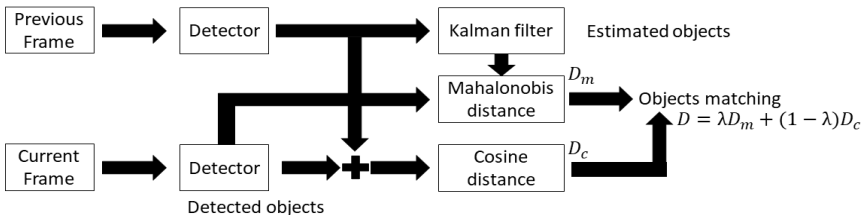
객체를 추적하는 작업은 객체에 ID를 부여하는 작업과 같다고 볼 수 있다. 추적 알고리즘은 영상에서 각 객체에게 그 객체만의 고유한 ID를 부여해준다. 이렇게 부여받은 ID는 프레임이 흘러감에도 이 고유한 ID값이 변하지 않고 유지하는 것이 추적 알고리즘의 핵심이라고 할 수 있다. 이러한 추적 알고리즘은 지금까지 보통 칼만 필터를 이용한 방식이 주를 이뤘다. 칼만 필터는 잡음을 포함하여 선형 역학계의 상태를 추적하는 재귀 필터이다.

이러한 칼만 필터를 이용한 대표적인 알고리즘에는 SORT가 있다. SORT는 딥러닝을 포함한 다양한 객체 탐지 알고리즘을 이용해 탐지한 객체들의 위치 정보를 기반으로, 이후 프레임에 올 객체의 위치 정보를 잡음과 속도 정보 등을 이용하여 구한다. 그리고 이렇게 구한 정보들과 기존의 정보를 IOU Distance 및 Hungarian 알고리즘을 이용하여 객체를 추적한다. 이러한 SORT는 계산과정이 많이 복잡하지 않기에 속도가 매우 빠르다는 장점이 있지만, 고려하는 것이 위치 정보뿐이라는 점에서 정확도가 다소 떨어진다는 단점이 존재한다.

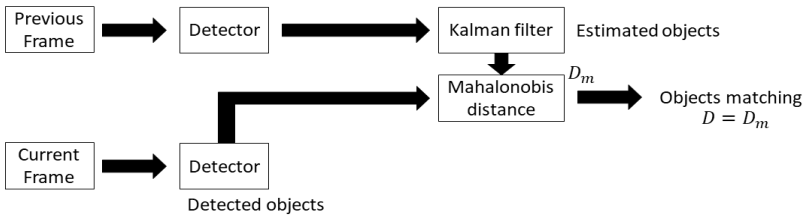
이러한 단점을 보완하기 위해 등장한 것이 DeepSORT이다. DeepSORT는 기존 SORT의 방식에 딥러닝의 CNN을 추가하여 추적의 정확도를 높인 방식이다. DeepSORT도 기본적으로는 SORT와 같이 칼만 필터 등을 사용하기에 객체의 위치나 속도 등의 정보를 이용한다. 하지만 DeepSORT는 여기에 더해 객체의 시각적 정보도 받는다. CNN을 이용해 각 객체의 이미지 정보를 받아와 특징값을 가져온다. 여기에 Cosine Distance를 이용하여 객체들의 유사도를 구해 객체 ID를 부여한다. 이렇듯 DeepSORT는 SORT보다 많은 정보를 이용해 추적하기에 SORT에 비해 시간이 오래 걸린다는 단점이 있지만, 정확도는 더 높다는 장점이 있다.

III. 제안하는 방법

그림 1(a)는 이러한 DeepSORT의 작동 방식을 간단히 나타낸 것이다. 그림에서 보는바와 같이 이전 프레임에서 추출한 객체 정보들에 칼만 필터를 이용하여 각 객체들의 현재 값을 추정하고, 이 정보와 현재 프레임에서 추출한 객체들의 정보를 이용하여 Mahalaonobis Distance(Dm)를 구한다. 동시에 이전 프레임의 객체들의 특징값과 현재 프레임의 특징값을 이용하여 Cosine Distance(Dc)를 구한다. 이전 프레임과 현재 프레임의 모든 객체 쌍들에 대해 $D(=\lambda Dm+(1-\lambda)Dc)$ 값을 구하여 D값이 일정 이상이 되면 서로 다른 프레임에 존재하는 두 객체는 같은 객체로 인식하게 된다. 그림 1(b)에서 보는바와 같이 SORT의 경우 Cosine Distance 값의 계산 없이 Mahalonobis Distance 값만을 이용하여 객체들의 일치성을 판단하게 된다.



(a) DeepSORT의 동작원리
(a) Procedure of DeepSORT



(b) SORT의 동작원리
(b) Procedure of SORT
그림 1. DeepSORT와 SORT의 동작 원리
Fig. 1. Procedure of DeepSORT and SORT

본 논문에서는 주어진 비율에 따라 주어진 프레임에 DeepSORT와 SORT를 혼합하여 적용시키는 기법을 제안한다. DeepSORT와 SORT가 적용되는 프레임의 비율은 실험 환경에 따라 여러 가지로 주어질 수 있는데, 예를 들어 1 대 5의 비율일 경우 다섯 프레임 당 한번씩 SORT가 적용 되고, 나머지 프레임의 경우 DeepSORT가 적용된다. 즉, 그림 1(a)와 그림 1(b)를 이용하여 다시 설명하면, 이전 프레임과 현재 프레임의 객체들 간의 매칭을 진행 할 때, 다섯 프레임 당 한번씩 D = Dm의 식을 이용하고(즉, SORT의 경우) 나머지 프레임의 경우 D = λ Dm+(1-λ)Dc의 식을 이용한다 (즉, DeepSORT의 경우). 둘 모두의 경우에서 D값이 정해진 임계값을 넘으면 각각 다른 프레임에 존재하는 해당 두 객체는 같은 객체로 인식된다.

정확도에 대한 성능지표로써 MOTA와 MOTP를 사용하고, 처리속도에 대한 성능지표로써 FPS를 사용한다. 먼저 MOTA는 식 (1)에 의해 구해진다[16].

$$MOTA = 1 - \frac{\sum_t (m_t + fp_t + mme_t)}{\sum_t g_t} \quad (1)$$

g_t는 현재 시간 t에 해당하는 프레임에 존재하는 실제 객체들의 수를 나타낸다. m_t는 현재 시간 t의 프레임에 적용한 추적기법이 미탐지한(즉, 실제로 존재하지만 탐지하지 못한) 객체들의 수를 나타내며, fp_t는 부정오류(즉, 존재하지 않은 객체들을 탐지함)의 수를 나타내며, mme_t은 미스매치(즉, 객체들이 서로 교차하였으나 이를 인지하지 못함)의 수를 나타낸다. MOTP는 식 (2)에 의해 구해진다[16].

$$MOTP = \frac{\sum_{i,t} d_t^i}{\sum_t c_t} \quad (2)$$

여기서 C_t는 현재 시간 t를 기준으로 현재 프레임과 이전 프레임의 통해 계산된 매치된 객체의 수를 나타낸다. d_t^i는 매치된 i번째 객체들에 대하여 두 객체의 거리를 나타낸다. 즉, MOTP는 매치된 객체들이(즉, 두 객체가) 얼마나 떨어져 있는지를 성능을 지표로 삼는다.

그림 2는 제안한 기법들의 성능을 나타낸다. Original은 DeepSORT가 단독으로 적용된 기법을 나타내며, 1/N SORT는 N 프레임 당 한번씩 SORT가 적용된 것을 나타낸다. accuracy는 MOTA를 나타낸다. 해당 그림에서 볼 수 있듯이, SORT가 적용되는 비율이 증가하여도 정확도의 감소는 거의 없으며, FPS는 적용된 비율만큼 상승한 것을 볼 수 있다.

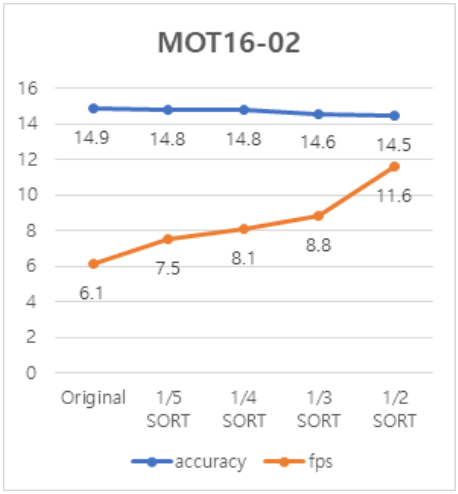


그림 2. 제안한 기법들의 성능 비교
Fig. 2. Comparison of performance of the proposed methods

다음 장에서 여러 데이터 셋에 대한 실험을 토대로 실험결과를 분석한다.

Ⅳ. 실 험

본 연구에서는 제안한 기법의 성능을 측정하기 위해 사실상 표준의 데이터 셋인 MOT16[17]을 사용하였다. 실험환경은 Intel(R) Xeon(R) Silver 4215R CPU @ 3.20GHz 32 CPUs, 251.5GB Memory에서 수행하였고, 성능 지표로써, 이전 장에서 설명한 MOTA와 함께, MOTP를 사용하였다. 실험대상으로써 제안 된 기법에 대해 먼저 실험을 진행하였고, 비교대상으로써 제안 된 기법에서 SORT가 적용되는 프레임에는 어떠한 추적기법도 적용하지 않는 기법을 고려하였다. 또한, 탐지 기법으로는 YOLO, FAST-RCNN 등 현대의 널리 사용되고 있는 어떠한 기법이라도 적용 가능하므로, 수행 된 실험은 탐지 기법이 포함 된 FPS는 측정하지 않았다.

표 1은 제안한 기법에 대한 실험결과를 나타낸다.

표 1. 제안한 기법과 DeepSORT, SORT와의 성능 비교
Table 1. Performance comparison of proposed technique, DeepSORT, and SORT

	VIDEO	MOTA	MOTP	FP	FN	IDSW	FPS
DeepSORT	MOT16-02	14.9	76.7	318	14830	31	6.1
	MOT16-04	33.2	80	1125	30572	104	2.6
	MOT16-10	30.7	76.4	181	8340	24	5.8
	MOT16-11	46.7	79.6	293	4580	21	6.8
	OVERALL	31.4	78.2	1917	58322	180	5.3
1/5 SORT	MOT16-02	14.8	76.57	280	14887	28	7.5(22.9% increase)
	MOT16-04	32.5	79.90	1037	30939	120	3.2(23.1% increase)
	MOT16-10	29.4	76.06	174	8483	35	7.0(20.7% increase)
	MOT16-11	46	79.29	294	4628	30	8.3(22.0% increase)
	OVERALL	30.7	78.9	1,785	58937	213	6.5(22.6% increase)
1/4 SORT	MOT16-02	14.8	76.5	269	14907	25	8.1(32.8% increase)
	MOT16-04	32.3	80	1041	31052	119	3.4(30.8% increase)
	MOT16-10	29.1	76.1	172	8532	29	7.6(31.0% increase)
	MOT16-11	45.9	79.2	299	4638	31	9.0(32.3% increase)
	OVERALL	30.6	78	1781	59129	204	7.1(34.0% increase)
1/3 SORT	MOT16-02	14.6	76.4	263	14935	30	8.8(44.3% increase)
	MOT16-04	31.9	79.9	990	31280	116	3.8(46.1% increase)
	MOT16-10	28.9	75.8	163	8569	26	8.3(43.1% increase)
	MOT16-11	45.9	79.2	273	4658	32	10.0(47.0% increase)
	OVERALL	30.3	77.8	1,689	59,442	204	7.7(45.3% increase)
1/2 SORT	MOT16-02	14.5	76.2	248	14978	36	11.6(90.2% increase)
	MOT16-04	31.3	79.8	978	31547	168	5.0(92.3% increase)
	MOT16-10	27.6	75.6	203	8668	51	11.0(89.6% increase)
	MOT16-11	44.9	78.8	310	4698	47	13.0(91.2% increase)
	OVERALL	28.8	78.7	1739	59891	302	10.1(90.6% increase)
SORT	MOT16-02	14.17	76.19	1320	13846	140	251.8
	MOT16-04	23.55	79.25	8907	27015	434	125.9
	MOT16-10	26.69	75.88	1102	7778	150	233.9
	MOT16-11	40.21	79.19	1167	4243	75	327.0
	OVERALL	26.15	77.63	12496	52882	799	113.2

첫 번째 열은 실험대상이 되는 서로 다른 기법들의 이름을 나타내는데, 각 기법의 이름은 그림 2에서 설명한 기법들과 동일한 것을 뜻한다. MOT16 데이터 셋은 7가지의 데이터 셋을 포함하는데, 본 실험에서는 MOT16-02, MOT16-04, MOT16-10, MOT16-11의 네 가지 데이터 셋을 고려하였다. 이 데이터 셋들이 가지는 가장 큰 차이는 밀집도이며, 이것은 각 데이터의 셋의 한 프레임 당 존재하는 평균 객체 수를 의미한다. MOT16-02, MOT16-04, MOT16-10, MOT16-11는 29.7, 45.3, 18.8, 10.2의 밀집도를 가지며, 해상도는 1920x1080픽셀로서 동일하다.

표 1의 첫 번째 행에서 FP(False Positive), FN (False Negative), IDSW(Id Switch)는 해당 영상 전체에서 확인된 총 부정오류 수, 미 탐지 수, 미스매치 수를 나타낸다. OVERALL은 MOTA, MOTP, FPS의 경우는 고려한 네 개의 데이터 셋에 대한 평균을, FP, FN, IDSW의 경우 합계를 나타낸다.

그림 3과 4는 각각 표 1(표 2도 마찬가지)의 MOT16-02와 MOT16-04의 다중객체추적의 정답을 나타낸다. 즉, 100%의 성능을 가지는 다중객체추적 기술이 적용된 결과를 나타내는 것으로 이 결과와 제안되는 기법들의 결과를 비교하여, FP, FN, IDSW를 측정하게 된다.

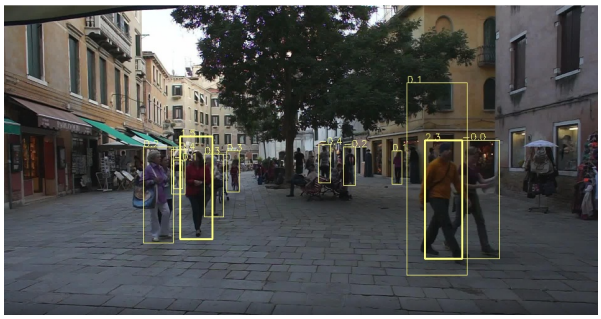


그림 3. MOT16-02에 대한 다중객체추적 결과
Fig. 3. Multi-object tracking result for MOT16-02

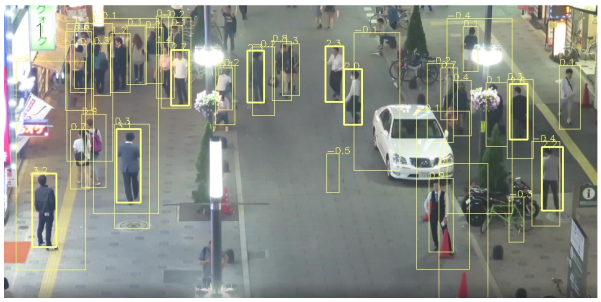


그림 4. MOT16-04에 대한 다중객체추적 결과
Fig. 4. Multi-object tracking result for MOT16-02

표 2. 1/N SKIP과 DeepSORT간의 성능 비교
Table 2. Performance comparison of 1/N SKIP and DeepSORT

	VIDEO	MOTA	MOTP	FP	FN	IDSW	FPS
DeepSORT	MOT16-02	14.9	76.7	318	14830	31	6.1
	MOT16-04	33.2	80	1125	30572	104	2.6
	MOT16-10	30.7	76.4	181	8340	24	5.8
	MOT16-11	46.7	79.6	293	4580	21	6.8
	OVERALL	31.4	78.2	1917	58322	180	5.3
1/5 SKIP	MOT16-02	12.1	76.8	213	15435	29	7.5(22.9% increase)
	MOT16-04	26.5	80.2	866	34001	106	3.1(19.2% increase)
	MOT16-10	24.3	76.3	143	9155	30	7.0(20.7% increase)
	MOT16-11	36.9	79.5	248	5513	31	8.3(22.0% increase)
	OVERALL	24.9	78.2	1,470	64,104	196	6.4(20.7% increase)
1/4 SKIP	MOT16-02	11.3	76.8	222	15567	33	8.0(31.1% increase)
	MOT16-04	24.7	80.2	852	34829	109	3.3(26.9% increase)
	MOT16-10	22.6	76.3	133	9376	28	7.5(29.3% increase)
	MOT16-11	34.5	79.4	211	5767	33	9.0(32.3% increase)
	OVERALL	23.3	78.2	1,418	65,539	203	6.9(30.2% increase)
1/3 SKIP	MOT16-02	10.0	77.0	188	15837	30	9.0(47.5% increase)
	MOT16-04	22.1	80.2	714	36250	97	3.8(46.1% increase)
	MOT16-10	20.5	76.3	122	9699	29	8.3(43.1% increase)
	MOT16-11	31.0	79.5	167	6132	29	10.0(47.0% increase)
	OVERALL	20.9	78.2	1,191	67,918	185	7.7(45.3% increase)
1/2 SKIP	MOT16-02	7.4	76.9	116	16361	28	11.6(90.2% increase)
	MOT16-04	16.5	80.3	522	39112	94	5.0(92.3% increase)
	MOT16-10	14.9	76.2	102	10432	37	11.0(89.6% increase)
	MOT16-11	22.9	79.2	119	6928	27	13.0(91.2% increase)
	OVERALL	15.4	78.1	859	72,833	186	10.1(90.6% increase)

표 2는 비교대상기법인 1/N SKIP의 성능을 나타낸다. 1/N SKIP은 매 N번째 프레임은 추적기법을 적용하지 않는 것을 뜻한다. 따라서 추적기법이 적용되지 않은 프레임의 모든 객체는 미 탐지로 판단된다. 표 1에서 DeepSORT 기법이 단독으로 적용되었을 때의 경우에서 주목 할 것은 MOT-16-02의 경우 MOT16-04와 동일한 해상도를 가지며 오히려 밀집도가 낮음에도 불구하고, 정화도가 MOT16-04의 반도 되지 않는다는 것이다. 이는 그림 3과 그림 4에서 보는바와 같이 MOT16-02의 영상이 훨씬 어둡고 원거리의 작은 크기의 객체들이 많아 다중객체 추적 기법이 객체 탐지를 더욱 어렵기 때문이다.

본 논문에서 제안하는 기법인 1/N SORT을 DeepSORT와 비교하였을 때, 정확도인 MOTA는 거의 감소하지 않는 반면, FPS는 SORT가 포함 된 비율만큼 증가하는 것을 확인할 수 있다. 예를 들어, MOT16-04에 대한 DeepSROT와 1/2 SORT의 경우, MOTA는 1.9만큼 떨어진 반면, FPS는 2배 가까이 증가하였다. 해당 데이터 셋에 대해 SORT 단독으로 적용되었을 때의 MOTA가 23.55인 것에 비해 정확도가 비약적으로 상승한 것을 볼 수 있다. 이것은 해당 표에서 볼 수 있듯이, 부정오류(즉, 존재하지 않는 객체를 탐지)에 대한 값이 SORT의 경우 1320인 반면, 1/2 SORT의 경우 248로써 매우 크게 줄었기 때문이다.

이러한 현상은 SORT에 비해 높은 정확도를 가지는 DeepSORT가 두 프레임마다 한번씩 SORT에 비해 높은 정확도로 객체의 위치를 알려주고 SORT가 다음 프레임에서 이전 프레임에서 DeepSORT가 알려 준 정보를 참고하기 때문이다.

반면에, 1/N SKIP의 경우 표 2에서 보는바와 같이 FPS가 추적기법을 적용하지 않은 프레임의 수의 비율대로 줄어드는 것을 볼 수 있다. 주목할 부분은 FP가 오히려 감소하고, FN이 비약적으로 증가한 것인데, 예를 들어 MOT16-04에 대한 DeepSROT와 1/2 SKIP의 경우, FP이 반 이하로 감소하였고, FN이 9000가량 증가하였다. 이것은 1/N의 경우 N프레임마다 한번 씩 어떠한 추적기법도 적용되지 않았기 때문에, 해당 프레임에는 부정오류가 발생하지 않고 모두 미 탐지에 해당되는 FN으로 고려되기 때문이다. FPS의 경우 1/N SORT와 1/N SKIP이 거의 차이가 없는 것을 볼 수 있다. 따라서 본 논문에서 제안한 기법인 1/N은 정확도의 손실이 거의 없이 FPS를 비약적으로 상승 시킬 수 있다는 것을 확인하였다.

Ⅴ. 결론 및 향후 과제

본 논문에서는 기존의 Deep SORT의 단점인 낮은 FPS를 개선하기 위하여 Deep SORT에 SORT의 기술을 혼합시키는 기법을 제안하였다. 다중 객체 추적의 성능을 위해 사용 널리 사용되는 데이터 셋인 MOT-16을 대상으로 제안한 방법에 대한 실험을 진행하였고, 비교대상으로 DeepSORT, SORT, DeepSORT에 일정 비율의 프레임에 추적 기술을 적용시킨 기법을 선정하였다. 실험 결과, 제안 된 방법은 정확도에 대한 손실에 거의 없이 FPS를 비약적으로 상승시키는 것을 확인 할 수 있었다. 향후 연구로는 일정한 순서대로 프레임에 다른 기법을 적용시키는 것이 아닌, 프레임의 워크로드 정보를 사전에 파악하여 적응적으로 다른 기법을 적용시키는 기법을 개발하고자 한다.

References

[1] Margrit Betke, Haritaoglu Esin, and Larry S. Davis, "Real-time multiple vehicle detection and tracking from a moving vehicle", Machine Vision and Applications, Vol. 12, No. 2, pp. 69-83, Aug. 2000.

[2] Weiming Hu, Tieniu Tan, Liang Wang, and S. Maybank, "A survey on visual surveillance of object motion and behaviors", IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), Vol. 34, No. 3, pp. 334-352, Aug. 2004. https://doi.org/10.1109/TSMCC.2004.829274.

[3] Wei-Lwun Lu, Jo-Anne Ting, James J. Little, and Kevin P. Murphy, "Learning to track and identify players from broadcast sports videos", IEEE Transactions on Pattern Analysis and Machine

Intelligence, Vol. 35, No. 7, pp. 1704-1716, Jul. 2013. https://doi.org/10.1109/tpami.2012.242.

[4] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation", IEEE Conference on Computer Vision and Pattern Recognition, pp. 580-587, Jun. 2014. https://doi.org/10.1109/CVPR.2014.81.

[5] Ross Girshick, "Fast r-cnn", IEEE International Conference on Computer Vision, pp. 1440-1448, Dec. 2015. https://arxiv.org/abs/1504.08083.

[6] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadiet, "You only look once: Unified, real-time object detection", IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, pp. 779-788, Jun. 2016. https://doi.org/10.1109/CVPR.2016.91.

[7] Yifu Zhang, et al., "FairMOT: on the fairness of detection and re-identification in multiple object tracking", arXiv preprint arXiv:2004.01888, Sep. 2020. [accessed: Aug. 12. 2021]

[8] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, "Simple online and realtime tracking" IEEE International Conference on Image Processing (ICIP), Phoenix, AZ, USA, pp. 3464-3468, Sep. 2016. https://doi.org/10.1109/ICIP.2016.7533003.

[9] Nicolai Wojke, Alex Bewley, and Dietrich Paulus, "Simple online and realtime tracking with a deep association metric", IEEE International Conference on Image Processing (ICIP), Beijing, China, pp. 3645-3649, Sep. 2017. https://doi.org/10.1109/ICIP.2017.8296962.

[10] C. Liu and J. Layland, "Scheduling algorithms for multiprogramming in a hard-real-time environment", Journal of the ACM, Vol. 20, No. 1, pp. 46-61, Jan. 1973. https://doi.org/10.1145/321738.321743.

[11] S. Baruah, N. K. Cohen, C. G. Plaxton, and D. A. Varvel, "Proportionate progress: a notion of fairness in resource allocation", Algorithmica, Vol. 15, No. 6, pp. 600-625, Jan. 1996. https://doi.org/10.1145/167088.167194.

[12] H. Baek and J. Lee, "Improved schedulability analysis of the contention-free policy for real-time systems", Journal of Systems and Software, Vol. 154, pp. 112-124, Aug. 2019. https://doi.org/10.1016/j.jss.2019.04.067.

[13] H. Baek, "Multi-level contention-free policy for rate monotonic scheduling algorithm on real-time systems", Journal of Korea Institute of Information Technology, Vol. 16, No. 2, pp. 29-38, Feb. 2018. http://dx.doi.org/10.14801/jkiit.2018.16.2.29.

[14] S. Vestal, "Preemptive scheduling of multi-criticality systems with varying degrees of execution time assurance", IEEE Real-Time Systems Symposium, pp. 239-243, Dec. 2007. http://dx.doi.org/10.1109/RTSS.2007.47.

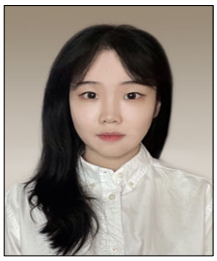
[15] H. Chwa, K. G. Shin, H. Baek, and J. Lee, "Physical-state-aware dynamic slack management for mixed-criticality systems", IEEE Real-Time and Embedded Technology and Applications Symposium, pp. 129-139, Apr. 2018. https://doi.ieeecomputersociety.org/10.1109/RTAS.2018.00023.

[16] Bernardin Keni and Rainer Stiefelhagen "Evaluating multiple object tracking performance: the CLEAR MOT metrics" EURASIP Journal on Image and Video Processing, Vol. 2008, No. 1, pp. 1-10, May 2008. https://doi.org/10.1155/2008/246309.

[17] Milan Anton, et al., "MOT16: A benchmark for multi-object tracking", arXiv preprint arXiv: 1603.00831, May. 2016. [accessed: Aug. 12. 2021]

저자소개

양 수 진 (Sujin Yang)



2016년 3월 ~ 현재 : 인천대학교 컴퓨터공학부 학사과정
관심분야 : 다중객체 추적, 딥러닝

정 인 화 (Inhwa Jung)

2019년 3월 ~ 현재 : 인천대학교 컴퓨터공학부 학사과정
관심분야 : 다중객체 추적, 딥러닝

강 동 화 (Donghwa Kang)

2019년 3월 ~ 현재 : 인천대학교 컴퓨터공학부 학사과정
관심분야 : 다중객체 추적, 자율주행, 딥러닝

백 형 부 (Hyeongboo Baek)

2010년 2월 : 건국대학교 컴퓨터공학과(공학사)
2012년 2월 : 한국과학기술원 전산학과(공학석사)
2016년 8월 : 한국과학기술원 전산학과(공학박사)
2018년 5월 : 성균관대학교 박사 후 연구원
2019년 2월 : 국방과학연구소 선임연구원
2019년 3월 ~ 현재 : 인천대학교 컴퓨터공학부 조교수
관심분야 : 실시간 시스템, 사이버 물리 시스템