일반논문 (Regular Paper)
방송공학회논문지 제24권 제3호, 2019년 5월 (JBE Vol. 24, No. 3, May 2019)
https://doi.org/10.5909/JBE.2019.24.3.495
ISSN 2287-9137 (Online) ISSN 1226-7953 (Print)

딥 러닝 및 칼만 필터를 이용한 객체 추적 방법

김 기 철[a], 손 소 희[a], 김 민 섭[a], 전 진 우[b], 이 인 재[b], 차 지 훈[b], 최 해 철[a]‡

Object Tracking Method using Deep Learning and Kalman Filter

Gicheol Kim[a], Sohee Son[a], Minseop Kim[a], Jinwoo Jeon[b], Injae Lee[b], Jihun Cha[b], and Haechul Choi[a]‡

요 약

딥 러닝의 대표 알고리즘에는 영상 인식에 주로 사용되는 CNN(Convolutional Neural Networks), 음성인식 및 자연어 처리에 주로 사용되는 RNN(Recurrent Neural Networks) 등이 있다. 이 중 CNN은 데이터로부터 자동으로 특징을 학습하는 알고리즘으로 특징 맵을 생성하는 필터까지 학습할 수 있어 영상 인식 분야에서 우수한 성능을 보이면서 주류를 이루게 되었다. 이후, 객체 탐지 분야에서는 CNN의 성능을 향상하고자 R-CNN 등 다양한 알고리즘이 등장하였으며, 최근에는 검출 속도 향상을 위해 YOLO(You Only Look Once), SSD(Single Shot Multi-box Detector) 등의 알고리즘이 제안되고 있다. 하지만 이러한 딥러닝 기반 탐지 네트워크는 정지 영상에서 탐지의 성공 여부를 결정하기 때문에 동영상에서의 안정적인 객체 추적 및 탐지를 위해서는 별도의 추적 기능이 필요하다. 따라서 본 논문에서는 동영상에서의 객체 추적 및 탐지 성능 향상을 위해 딥 러닝 기반 탐지 네트워크에 칼만 필터를 결합한 방법을 제안한다. 탐지 네트워크는 실시간 처리가 가능한 YOLO v2를 이용하였으며, 실험 결과 제안한 방법은 기존 YOLO v2 네트워크에 비교하여 7.7%의 IoU 성능 향상 결과를 보였고 FHD 영상에서 20 fps의 처리 속도를 보였다.

Abstract

Typical algorithms of deep learning include CNN(Convolutional Neural Networks), which are mainly used for image recognition, and RNN(Recurrent Neural Networks), which are used mainly for speech recognition and natural language processing. Among them, CNN is able to learn from filters that generate feature maps with algorithms that automatically learn features from data, making it mainstream with excellent performance in image recognition. Since then, various algorithms such as R-CNN and others have appeared in object detection to improve performance of CNN, and algorithms such as YOLO(You Only Look Once) and SSD(Single Shot Multi-box Detector) have been proposed recently. However, since these deep learning-based detection algorithms determine the success of the detection in the still images, stable object tracking and detection in the video requires separate tracking capabilities. Therefore, this paper proposes a method of combining Kalman filters into deep learning-based detection networks for improved object tracking and detection performance in the video. The detection network used YOLO v2, which is capable of real-time processing, and the proposed method resulted in 7.7% IoU performance improvement over the existing YOLO v2 network and 20 fps processing speed in FHD images.

Keyword : YOLO, Kalman filter, Object tracking, CNN, Deep learning

Ⅰ. 서 론

최근 무인비행체는 다양한 ICT(Information and Communications Technologies) 분야 중 일정 수준 이상의 상용화를 이룬 주요 기술이며, 무인비행체의 개발이 활발해지고 관련 시장규모도 꾸준히 성장 중이다. 세계 무인비행체 시장 규모는 2020년 115억 달러로 성장할 전망이고, 개인용 무인비행체는 연 10%씩 꾸준히 성장하며 2020년 26억 달러까지 시장이 확대될 것으로 예측된다. 따라서 무인비행체 기술의 발전에 따라 성장 속도는 더욱 빨라질 것으로 보인다[1][2].

하지만 무인비행체는 항공법 이슈, 사고 위험성, 불법적인 활용 등 끊임없는 문제와 논란을 일으키고 있다. 무인비행체는 레이더 상에 잡히지도 않을뿐더러, 전파를 송수신하는 기기도 없어 추적이 어렵다. 또한, 항공기와 달리 시리얼 번호나 공식 등록 절차도 필요치 않기 때문에 문제 발생 시 책임자를 찾기도 힘들다. 아직 항공기 운항 중에 실제 충돌 사례는 없으나, 충돌할 경우 파편이 기체와 부딪치거나 내부로 들어갈 수 있어 소형 무인비행체라 하더라도 비행에 심각한 문제를 일으킬 수 있다. 따라서 무인비행체가 허가되지 않는 곳에서 운행할 경우 많은 위험이 따르게 되고 인적, 물적 피해가 발생할 수 있다. 이러한 환경에서 무인비행체의 허가되지 않은 장소에서의 사용을 적절히 통제하기 위해 관리 및 감시가 필요하다.

이러한 연구 필요성에 기초하여 무인비행체가 지표면이 보이지 않는 하늘에서 비행할 때 카메라를 통해 무인비행체를 탐지하는 방법들이 주로 연구되어 왔다[3][4]. 그러나 무인비행체가 저고도에서 비행할 때는 영상의 배경에 산악지형 혹은 도심 지형과 같이 복잡한 환경을 갖기 때문에 무인비행체를 정확하게 탐지하기는 쉽지 않다. 또한, 무인비행체는 다양한 크기, 빠른 움직임을 갖고 움직이는 새와 혼동을 일으킬 수 있으므로 더욱 어려운 문제이다. 본 논문에서는 이러한 복잡한 환경에서 무인비행체를 탐지하고 추적하기 위해 YOLO(You Only Look Once) v2와 칼만 필터(Kalman filter)를 결합한 탐지 및 추적 방법을 제안한다. YOLO v2는 딥 러닝(Deep learning) 기반 탐지 네트워크 중에서 높은 탐지와 정확성뿐만 아니라 실시간 처리가 가능한 결과를 보인다[8][9]. YOLO v2는 입력 영상을 격자 셀로 나누어 객체 존재 여부와 종류에 대한 확률값을 계산하고, 이를 바탕으로 신뢰도를 산출하여 객체를 탐지한다. GPU를 사용했을 때 처리 속도는 40-90 fps 수준으로 실시간 객체 탐지에 적합하다. 하지만 마지막 특징 맵에서만 객체를 탐지하기 때문에 구조가 단순하고 탐지속도가 빠른 장점이 있지만, 소형 객체에 대한 탐지 정확성은 매우 떨어지는 결과를 보인다. 따라서 본 논문에서는 탐지가 실패하는 부분에서 성능을 향상할 방법을 제안한다. 제안 알고리즘에서는 무인비행체 탐지를 위해 딥 러닝 기반 CNN(Convolutional Neural Networks)[5][6][7] 네트워크인 YOLO v2를 적용하고, 이에 나오는 결과에 추가로 추적 기능을 더하기 위해 칼만 필터[10][11][12][13] 알고리즘을 이용하였다.

만약 객체 탐지 단계에서 검출하지 못했을 경우, 이전의 탐지 내용에 대한 정보를 얻기 위해 칼만 필터를 기반으로 현재 객체 탐지를 다시 진행한다. 따라서 YOLO v2 방법의 장점을 유지하면서 탐지가 실패하는 경우 성능을 향상할 수 있도록 칼만 필터를 수정하여 무인비행체의 위치를 더 정확하게 찾을 수 있도록 결합하였다.

본 논문의 구성은 다음과 같다. 2장에서는 무인비행체 탐지를 위한 CNN 기반 객체 탐지 알고리즘과 추적을 위한 칼만 필터 기반 객체 추적 알고리즘을 설명한다. 3장에서는 딥 러닝 및 칼만 필터를 이용한 객체 추적 방법을 제안하고, 4장에서는 실험 결과를 기술하였다. 마지막으로 5장에서는 결론을 맺는다.

a) 한밭대학교 정보통신전문대학원 멀티미디어공학과(Information of Departments, Hanbat National University)
b) 한국전자통신연구원(ETRI)
‡ Corresponding Author : 최해철(Haechul Choi)
E-mail: choihc@hanbat.ac.kr
Tel: +82-42-821-1149
ORCID: http://orcid.org/0000-0002-7594-0828
※ 본고는 2019년도 정부(경찰청)의 재원으로 국토교통과학기술진흥원의 지원을 받아 수행된 연구임(No. 19PCRD-C139687-03, 무인비행장치의 불법 비행 감지를 위한 EO/IR 연동 레이다 개발 및 실증시험).
※ This research was supported by a grant from Police Science and Technology R&D Program funded by Korean National Police Agency. [No.19PCRD-C139687-03, Development and Field Demonstration Test of Surveillance System using radar and EO/IR for detecting illegal Flight of UAVs]
· Manuscript received April 15, 2019; Revised May 24, 2019; Accepted May 24, 2019.

Ⅱ. 관련 연구

1. 딥 러닝 기반 객체 탐지 방법

딥 러닝[14][15]은 신경망 네트워크로 많은 수의 계층을 만들어 학습하는 기계학습 분야이다. 신경망 네트워크는 오래전인 1950년대부터 인간의 뇌에서 영감을 얻어 시작되었지만 낮은 컴퓨팅 성능 문제와 XOR 연산 문제 등 탓으로 잠시 사라졌다가 최근 컴퓨팅 성능, 빅 데이터[16], RBM (Restricted Boltzmann Machine)[17]으로 오버 피팅 문제 해결 등을 통해 부활하였다. CNN의 특징 추출(Feature Extraction)은 컨볼루션 레이어와 풀링 레이어를 차례로 쌓은 형태로 되어있다. 컨볼루션 레이어는 필터와 같이 컨볼루션 연산을 통해 입력 영상을 변환하며, 풀링 레이어는 입력을 다운 샘플링하기 위해 적용되는 레이어로써 입력 값의 불필요한 정보를 없애고 압축하는 데 이용한다. 즉, 풀링 레이어는 이미지의 차원을 축소하는 역할을 한다. 컨볼루션 신경망인 CNN은 뇌의 시각 피질이 이미지를 처리하고 인식하는 원리를 이용한 신경망이다. 주로 영상 인식 분야에서 적용되며 여러 분야에서 탁월한 성능을 보여주고 있는 인공 신경망이다.

CNN 기반 객체 탐지 알고리즘 중 가장 잘 알려진 방식은 2014년에 발표된 R-CNN(Region-CNN)이다[18]. R-CNN 알고리즘이 발표되기 이전에는 객체 탐지를 위해 주로 SIFT (Scale Invariant Feature Transform)[19], HOG(Histogram of Oriented Gradient)[20], Optical Flow[21], haar-like features[22] 알고리즘 등이 자주 사용되었으며, 이는 영상에 존재하는 low-level feature에 기반을 두기 때문에 성능에 한계가 존재한다. 반면 R-CNN은 입력 영상에서 후보 영역을 만들고, 각 후보 영역에 대해 컨볼루션 신경망을 사용하여 해당 영상에 대해 분류를 하는 방법을 적용하여 기존 연구 대비 매우 높은 성능 향상을 보였다. R-CNN이 발표된 이후 R-CNN을 개선하기 위한 많은 연구가 진행되었는데, 대표적으로 SPP-Net[23], Fast R-CNN[24], Faster R-CNN[25] 등이 있다. 하지만 R-CNN 계열의 객체 탐지 네트워크는 높은 탐지 성능을 보이지만 처리 속도가 매우 느린 단점을 가지고 있다. 후보 영역을 생성하고 각 후보에 대해 분류를 진행하고, 분류 결과 정보를 후처리를 통해 중복 탐지 결과를 제거하는 등 복잡한 파이프라인 구조로 되어 있고, 개별 요소들이 분리되어 학습되므로 연산 양도 많고 최적화하기 어렵다. 실시간 처리가 가능한 객체 탐지 네트워크를 위해 복잡한 파이프라인 구조를 단일망에 해결하도록 설계된 SSD(Single Shot Multi-box Detector)[26], YOLO[8], YOLO v2[9] 등의 탐지 네트워크가 개발되었다. 본 논문에서는 탐지 네트워크 중 정확성과 속도 측면에서 우수한 알고리즘인 YOLO v2를 사용한다. YOLO v2는 전체 영상에 대해 단일 신경망을 통해 경계 박스의 좌표와 각 클래스의 확률을 계산할 수 있다. YOLO v2는 일반적으로 입력 영상의

표 1. 탐지 시스템 네트워크 구성
Table 1. The detection system network

Layer		Filters	Size	Input	Output
0	conv	32	3×3/1	416×416×3	416×416×32
1	max		2×2/2	416×416×32	208×208×32
2	conv	64	3×3/ 1	208×208×32	208×208×64
3	max		2×2/2	208×208×64	104×104×64
4	conv	128	3×3/1	104×104×64	104×104×128
5	conv	64	11/1	104×104×128	104×104×64
6	conv	128	3×3/1	104×104×64	104×104×128
7	max		2×2/2	104×104×128	52×52×128
8	conv	256	3×3/1	52×52×128	52×52×256
9	conv	128	1×1/1	52×52×256	52×52×128
10	conv	256	3×3/1	52×52×128	52×52×256
11	max		2×2/2	52×52×256	26×26×256
12	conv	512	3×3/1	26×26×256	26×26×512
13	conv	256	1×1/1	26×26×512	26×26×256
14	conv	512	3×3/1	26×26×256	26×26×512
15	conv	256	1×1/1	26×26×512	26×26×256
16	conv	512	3×3/1	26×26×256	26×26×512
17	max		2×2/2	26×26×512	13×13×512
18	conv	1024	3×3/1	13×13×512	13×13×1024
19	conv	512	1×1/1	13×13×1024	13×13×512
20	conv	1024	3×3/1	13×13×512	13×13×1024
21	conv	512	1×1/1	13×13×1024	13×13×512
22	conv	1024	3×3/1	13×13×512	13×13×1024
23	conv	1024	3×3/1	13×13×1024	13×13×1024
24	conv	1024	3×3/1	13×13×1024	13×13×1024
25	route	16			
26	conv	64	1×1/1	26×26×512	26×26×64
27	reorg		/2	26×26×64	13×13×256
28	route	27 24			
29	conv	1024	3×3/1	13×13×1280	13× 13×1024
30	conv	30	1×1/1	13×13×1024	13×13×30
31	detection				

크기가 작을수록 수행 속도는 빨라지지만, 인식 성능은 줄어드는 경향을 보이며, 416×416을 입력 영상으로 사용하는 경우 기존 방식들보다 성능은 유사하거나 높고 처리 속도도 빠른 것을 확인할 수 있다. 본 논문에서 사용하는 YOLO v2의 네트워크 구성은 표 1과 같다. 최종 출력 모양은 13×13×30으로 13은 그리드 셀의 개수이고, 30은 다음과 같이 구성된다. 4개의 경계 박스값과 1개의 객체의 확률값, 그리고 객체 종류 개수는 1이므로 총 6이 된다. 마지막으로 각 그리드 당 5개의 박스를 예측하므로 6×5=30이 된다. 따라서 최종 출력인 (13×13×30) 모양이 구성된다.

2. 칼만 필터

칼만 필터는 잡음이 포함된 선형 역학계의 상태를 추적하는 재귀 필터로써 루돌프 칼만이 개발하였다. 칼만 필터는 컴퓨터 비전, 로봇 공학 등의 여러 분야에서 사용되고 있으며, 시간에 따라 진행한 측정을 기반으로 한다. 따라서 해당 순간에만 측정한 결과만 사용한 것보다는 좀 더 정확한 결과를 기대할 수 있다. 또한, 잡음까지 포함된 입력 데이터를 재귀적으로 처리하는 필터로써 현재 상태에 대한 최적의 통계적 예측을 진행할 수 있다. 칼만 필터는 선형 시스템을 기반으로 설계된 필터로써 정상상태 기준의 모델을 수행하였으므로, 선형 시스템이라고 가정할 수 있다. 칼만 필터의 알고리즘은 예측과 업데이트를 반복적으로 계산하여 시스템 출력 값의 잡음 영향을 최소화하고 출력 값을 예측한다. 예측은 현재 상태의 예측을 말하고, 업데이트는 현재 상태에서 관측된 측정까지 포함한 값을 통해서 더 정확한 예측을 할 수 있는 것을 말하며, 칼만 필터의 과정은 그림 1과 같다. 예측에서는 직전에 추정한 값이 어떻게 변할 것인지 추정하는 과정으로써 칼만 필터의 계산 과정을 살펴보면, 먼저 그림 1의 과정 1과 같이 직전 추정값($\hat{x}_{k-1}$)과 오차 공분산(P_{k-1})을 가지고 예측값($\hat{x}_{\bar{k}}$) 및 예측값 오차 공분산(P_k^-)을 계산하는 예측 과정을 수행한다. 예측 과정의 계산 식은 시스템 모형화를 통해 구성되며 칼만 필터의 성능을 결정하고 설계자가 변경할 수 있는 설계값이다. 예측 과정에 이어서 추정 과정에서 과정 2와 같이 칼만 이득을 계산한다. 계산된 칼만 이득은 과정 3 추정값 계산 및 과정 4 추정값 오차 공분산 계산에 사용된다. 그림 1의 과

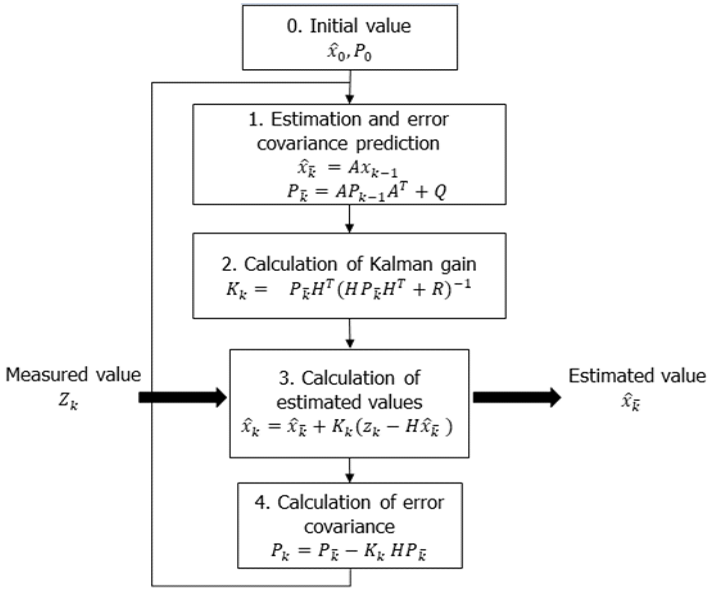

그림 1. 칼만 필터의 흐름도
Fig. 1. Overall flowchart of Kalman filter

정 2의 칼만 이득 계산 식에서 행렬 H와 치환 행렬 H^T는 계산을 위한 변환 행렬이므로 단순화시켜 전개하면 식 1과 같이 표현되며, 전개한 수식을 통해 칼만 이득(K_k)은 측정값 오류(R) 대비 추정값 오류($P_{\bar{k}}$)의 비율임을 알 수 있다. 따라서 측정값 오류가 추정값 오류와 비교하면 상대적으로 크면 칼만 이득은 작아지고, 측정값 오류가 추정값 오류보다 상대적으로 작으면 칼만 이득은 커지게 된다.

$$K_k = P_{\bar{k}}H^T(HP_{\bar{k}}H^T+R)^{-1} = \frac{P_{\bar{k}}H^T}{HP_{\bar{k}}H^T+R} = \frac{P_{\bar{k}}}{P_{\bar{k}}+R} \quad (1)$$

과정 3의 추정값 계산 식을 전개하면 식 2와 같이 표현할 수 있다. 칼만 이득(K_k)은 0에서 1 값을 가지며, 예측값($\hat{x}_{\bar{k}}$)과 측정값(Z_k)에 곱해지는 가중치를 의미한다. 즉, 예측값과 추정값 중 상대적으로 신뢰할 수 있는 값에 큰 가중치 두고 계산하여 단위시간의 추정값을 계산한다. 칼만 필터는 순환적으로 반복되어 수행되므로, 필터가 잘 작동되어 예측값 계산의 신뢰성이 높아질수록 칼만 이득은 점점 작아지며, 측정값 대비 상대적으로 높은 가중치가 예측값에 부여되게 된다.

$$\begin{aligned}\hat{x}_k &= \hat{x}_{\bar{k}} + K_k(z_k - H\hat{x}_{\bar{k}}) = \hat{x}_{\bar{k}} + K_k z_k - K_k H\hat{x}_{\bar{k}} \\ &= (1-K_kH)\hat{x}_{\bar{k}} + K_k z_k\end{aligned} \quad (2)$$

과정 1에서 예측값 계산을 위해 예측시스템 오류 분산 값이 사용되며, 또한 과정 2 칼만 이득 계산 식에서 예측시스템 오류 분산 값과 측정값 오류 분산 값이 이용된다. 과정 4에서 계산된 칼만 이득으로 예측시스템 오류 분산 값을 최적화한다.

Ⅲ. 딥 러닝 및 칼만 필터를 이용한 객체 추적 방법

1. 딥 러닝 기반 객체 탐지 방법의 문제점

본 절에서는 기존 딥 러닝 기반 객체 탐지 방법의 문제점에 대해 분석한다. 일반적으로 딥 러닝 기반 탐지 알고리즘은 단일 프레임을 대상으로 객체 탐지를 수행한다. 하지만 이러한 딥 러닝 기반 탐지 알고리즘은 여러 단계의 레이어를 거쳐서 나온 특징 맵을 기반으로 경계 박스를 예측하기 때문에 위치 예측이 부정확할 수 있고, 단일 프레임에서만 탐지를 수행하므로 잡음에 의한 탐지 실패 확률이 높아질 수 있다.

이를 보완하기 위해 본 논문에서는 동영상 내 연속 프레임을 이용하여 영상 간의 정보를 활용하는 방법을 제안한다. 우선 객체와 카메라의 움직임을 선형이라고 가정하여 무인비행체의 움직임을 궤적 정보로 사용한다. 따라서 탐지가 실패하더라도 연속 프레임 간의 객체 움직임 정보를 반영하고 잡음을 제거함으로써 무인비행체 탐지 및 추적 성능을 높일 방법을 제안한다.

2. 제안 방법

본 논문에서는 무인비행체 탐지 및 추적의 정확도를 높이기 위해 객체 탐지에 주로 활용되는 CNN 기반 탐지 네트워크인 YOLO v2와 추적을 위한 칼만 필터를 결합한 방법을 제안한다. 제안한 시스템의 전체 구성도는 그림 2과 같다. 입력 영상과 사전에 학습된 데이터를 입력받고, 이는 무인비행체 탐지를 위해 YOLO v2 시스템에 입력으로 들어간다. 원하는 대상 객체가 성공적으로 탐지된다면 출력값으로 탐지 결과의 좌표값이 나오게 된다. 그 후, 칼만 필

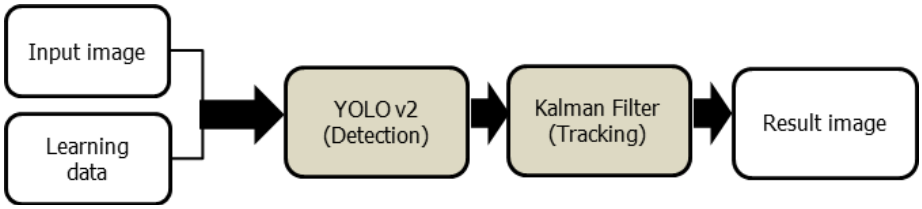

그림 2. 시스템 전체 흐름도
Fig. 2. System flowchart

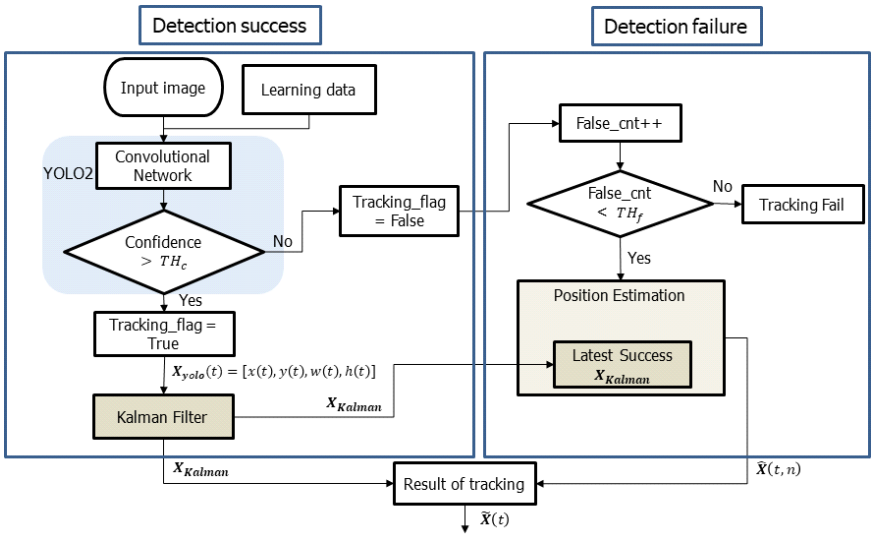

그림 3. 제안 방법의 흐름도
Fig. 3. Flowchart of proposed method

터로 보정을 하고 최종 추적 결과를 출력한다. 만약 탐지 단계에서 실패한다면 이전에 성공했던 탐지 결과 값을 기반으로 칼만 필터 예측을 통해 나온 궤적 정보를 받아와 이전의 탐지 결과와 더해져 최종 추적 결과를 출력한다. 제안 방법의 상세 흐름도는 그림 3과 같다. 탐지 성공은 YOLO v2 네트워크에서 나온 결과인 confidence 값이 TH_c 보다 더 크면 탐지가 성공했다고 판단하고 $Tracking\ flag$ 값을 $True$로 설정한다. TH_c 값은 실험을 통해 0.4로 설정했다. 그리고 경계 박스의 값인 $X_{yolo}(t)$ 값을 칼만 필터에 입력으로 넣는다. 칼만 필터에서 나온 결과를 토대로 추적 결과인 X_{Kalman} 값이 나오게 되고, 성공 값은 변수에 저장해 놓는다. 탐지가 실패하는 경우에는 먼저 $Tracking\ flag$ 값을 $False$로 설정하고 $False\ cnt$ 값을 증가시킨다. $False\ cnt$ 값이 TH_f 값보다 작을 때만 이전 프레임에서 마지막에 추적에 성공한 궤적 정보인 X_{Kalman} 값을 받아 더해주고, 최종 추적 결과를 출력한다.

Ⅳ. 실험 결과 및 분석

본 논문에서는 YOLO v2 네트워크를 훈련시키기 위해 건물, 산, 하늘, 구름 등의 배경을 갖는 무인비행체 데이터를 약 4,000장 사용하였고, 테스트 세트에 이용된 이미지 개수는 약 500장이다. 그림 4는 다양한 배경을 갖는 무인비행체 학습 데이터의 예를 보이며 해상도는 FHD이고, 무인비행체는 DJI 사의 Phantom 4 Professional 모델을 이용하였다. 실험은 우분투 16.04 운영체제에서 Intel i7-6850K 3.6GHz CPU와 32GB RAM 그리고 NVIDIA geforce GTX

그림 4. 다양한 배경의 학습 데이터
Fig. 4. Examples of training data

1080 Ti GPU를 가진 환경에서 진행하였으며, 라이브러리는 python 3.5.5 버전, 기계 학습과 딥 러닝을 위해 구글에서 만든 오픈 소스 라이브러리인 tensorflow[28] 1.10.0 버전, openCV 3.4.0[29] 버전 라이브러리를 사용하였다.

본 논문에서는 IoU(Intersection over Union) 값을 사용하여 제안 방법의 성능을 비교 분석한다. IoU란, 그림 5와 같이 두 영역의 교차 영역의 넓이를 합 영역의 값으로 나눈 값을 뜻한다. 객체 탐지에서 예측된 경계 박스의 정확도를 평가하는 지표 중 하나로 사용되며, 예측된 경계 박스와 실제 참값(ground truth) 경계 박스를 비교해 IoU 값을 구하게 된다. IoU 계산은 다음 식 3과 같으며, 일반적으로 그림 5와 같이 IoU 값이 0.5 이상이면 탐지 및 추적 성공으로 간주한다.

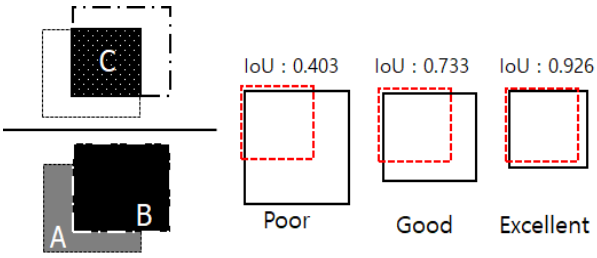

그림 5. IoU 계산 및 다양한 IoU 값에 대한 평가
Fig. 5. Calculation of IoU and evaluation of the various IoU values

$$IoU = \frac{C}{A \cup B} \tag{3}$$

그림 6은 기존 YOLO v2의 실험 결과를 나타내며, 평균 IoU 값은 0.595이고 속도는 FHD((Full High Definition)에서 24 fps이다. 그림 7은 제안방법과 YOLO v2 네트워크의 IoU 실험 결과를 나타낸다. 탐지가 성공한 부분에서는 기존의 YOLO v2 네트워크와 비슷한 결과를 보인다. 테스트를 위해 500장을 사용했고, 실패 횟수는 YOLO v2 단독으로 사용했을 때 75회이며, 제안한 방법을 사용했을 때 36회로 확인되었다. 두 알고리즘 모두 탐지 실패하는 경우에

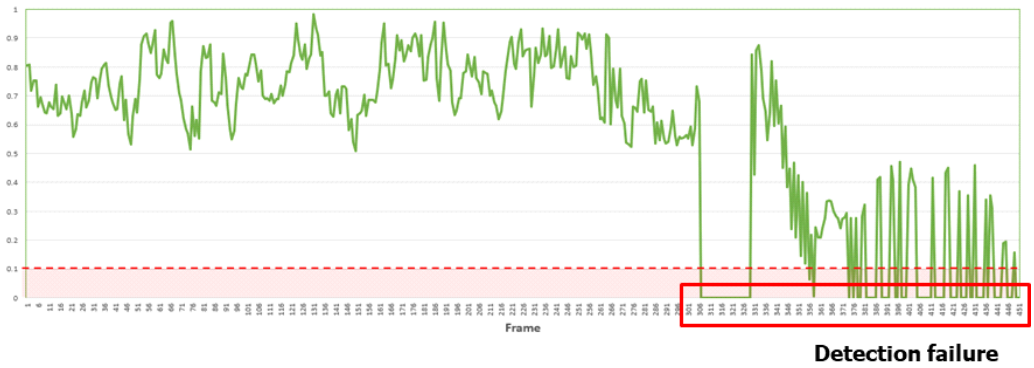

그림 6. YOLO v2 네트워크 IoU 값 그래프
Fig. 6. IoU results of YOLO v2 network

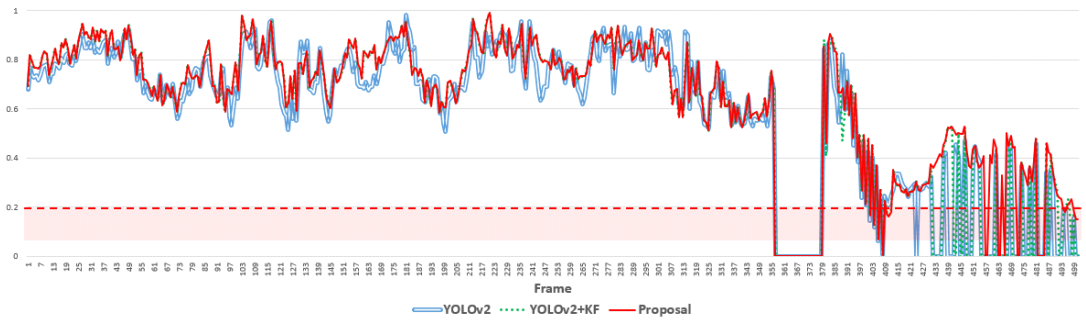

그림 7. YOLO v2 네트워크와 제안 방법의 IoU 값 비교 그래프
Fig. 7. Comparison of proposed method and YOLO v2 with IoU results

그림 8. YOLO v2의 탐지 실패 결과와 제안 방법의 추적 성공 결과의 예
Fig. 8. Examples of detection failure (YOLO v2) and tracking success (Proposed method)

대한 원인을 분석한 결과 무인비행체와 배경이 비슷한 색상이거나 잡음이 많이 포함되어 있다. 그러나 전체적으로 YOLO v2 네트워크 단독 사용 시에는 탐지가 실패해도 제안한 방법은 무인비행체를 계속해서 추적하고 있음을 그래프를 통해 확인할 수 있다. 제안 방법의 평균 IoU 값은 0.672이고 속도는 FHD 기준 20 fps이다. 그림 8은 YOLO v2와 제안 방법의 실험 결과의 예를 나타낸다. 이를 통해 YOLO v2 단독 사용 시에는 탐지가 실패하더라도 YOLO v2와 칼만 필터를 결합한 제안 방법은 추적에 성공함을 확인할 수 있다.

V. 결 론

본 논문에서는 CNN 기반 탐지 네트워크인 YOLO v2와 칼만 필터를 결합하여 동영상에서 단일 객체의 추적 및 탐지 성능을 향상하는 방법을 제안하였다. 제안 방법의 구성은 다음과 같다. 먼저 YOLO v2 네트워크를 이용해 객체 검출을 수행한다. 만약 딥 러닝 기반 알고리즘을 이용한 객체 탐지 단계에서 대상 객체를 검출하지 못했을 경우, 이전의 객체 탐지 결과를 기반으로 칼만 필터를 적용하여 객체 탐지를 진행한다. 딥 러닝 기반 탐지 네트워크는 성능이 우수한 편이지만, 학습 데이터에 기반을 두어 최적의 판단이나 예측값을 찾아내기 때문에 학습데이터와 다른 환경에서 촬영된 입력 영상에 대해서는 성능이 떨어져 추적이 실패할 수도 있다. 이를 보완하기 위해 탐지 실패 시 칼만 필터를 통해 예측된 궤적 정보를 이용하여 추적 성능을 향상했으며, 실험 결과 기존 YOLO v2 대비 7.7%의 탐지 및 추적 성능 향상 효과를 보였다. 처리 속도는 FHD 해상도에서 기

존 YOLO v2 속도 대비 4 fps 감소한 20 fps 결과를 보였다. 하지만 제안한 방법은 기존의 YOLO v2 단독으로 사용하는 경우보다 다소 부정확해지는 것을 보였다. 이는 칼만 필터의 재귀적 특징으로서 반복적으로 업데이트 및 예측 과정에서 생기는 현상이다. 제안 방법은 예측이 실패한 경우에서 성능 향상을 목표로 설정해 발생한 것으로 파악된다. 향후 탐지 및 추적 성능 향상을 위해 첫 번째로 다양한 크기와 배경의 무인비행체 학습 데이터 DB 구축이 필요하다. 본 논문에서는 단일 객체를 추적하기 위해 제한된 모델과 크기의 무인비행체 학습데이터를 수집하였지만, 향후 다양한 기종과 다양한 크기의 학습데이터 및 복잡한 배경 기반의 데이터를 보강하여 학습시켜야 할 것이다. 두 번째로 본 논문에서는 YOLO v2에 최적화시키기 위하여 특징 맵 크기를 32×32로 특정시켰는데, 실제 실험에서 특징 맵보다 작은 소형 객체(일례로 1km 이상 떨어진 무인비행체)를 추적할 때 어려움을 보였다. 따라서 특징 맵의 크기를 다양하게 추가한다면 성능 향상에 효과적일 것이다.

제안한 알고리즘은 무인비행체를 더 빠르고 정확하게 탐지 및 추적하는 데 활용할 수 있으며, 딥 러닝을 이용한 무인비행체 자동 인식 기술과 연계하여 안티 드론 시스템에 효과적으로 활용될 수 있다. 마지막으로 복수 영상을 이용하고 복잡도에 영향을 최소화하기 위해 칼만 필터를 사용하였는데, 앞으로의 방향은 딥 러닝 기법 분석 및 적용 방법에 관한 연구가 필요하다.

참 고 문 헌 (References)

[1] Teal Group, 2014 Market Profile and Forecast, World Unmanned aerial Vehicle Systems, 2014

[2] Choi Youngchul, Ahn Hyosung. (2015). Dron's current and technology development trends and prospects. The world of electricity, 64(12), 20-25.

[3] Eric N. Johnson, Anthony J. Calise, Yoko Watanabe, Jincheol Ha, and James C. Neidhoefer, 2007, “Real-Time Vision-Based Relative Aircraft Navigation,” *Journal of Aerospace Computing, Information, and Communication*, Vol.4, pp.707-738

[4] John Lai, Luis Mejias, and Jason J. Ford, 2011, “Airborne Vision-Based Collision-Detection System,” Journal of Field Robotics, Vol.28, Issue 2, pp.137-157.

[5] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." *Advances in neural information processing systems*. 2012.

[6] Schmidhuber, Jurgen. "Deep learning in neural networks: An overview." *Neural networks* 61 (2015): 85-117.

[7] Gidaris, Spyros, and Nikos Komodakis. "Object detection via a multi-region and semantic segmentation-aware cnn model." *Proceedings of the IEEE International Conference on Computer Vision*. 2015.

[8] Redmon, Joseph, et al. "You only look once: Unified, real-time object detection." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016.

[9] Redmon, Joseph, and Ali Farhadi. "YOLO9000: better, faster, stronger." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017.

[10] Brown, Robert Grover, and Patrick YC Hwang. Introduction to random signals and applied Kalman filtering. Vol. 3. New York: Wiley, 1992.

[11] Ristic, Branko, Sanjeev Arulampalam, and Neil Gordon. "Beyond the Kalman filter." *IEEE Aerospace and Electronic Systems Magazine* 19.7 (2004): 37-38.

[12] Haykin, Simon. *Kalman filtering and neural networks*. Vol. 47. John Wiley & Sons, 2004.

[13] Peterfreund, Natan. "Robust tracking of position and velocity with Kalman snakes." IEEE transactions on pattern analysis and machine intelligence 21.6 (1999): 564-569.

[14] LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton. "Deep learning." *nature* 521.7553 (2015): 436.

[15] Deng, Li, and Dong Yu. "Deep learning: methods and applications." *Foundations and Trends® in Signal Processing* 7.3-4 (2014): 197-387.

[16] Mayer-Schönberger, Viktor, and Kenneth Cukier. *Big data: A revolution that will transform how we live, work, and think*. Houghton Mifflin Harcourt, 2013.

[17] Nair, Vinod, and Geoffrey E. Hinton. "Rectified linear units improve restricted boltzmann machines." *Proceedings of the 27th international conference on machine learning* (ICML-10). 2010.

[18] Girshick, Ross, et al. "Rich feature hierarchies for accurate object detection and semantic segmentation." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2014.

[19] Lowe, David G. "Object recognition from local scale-invariant features." *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*. Vol. 2. Ieee, 1999.

[20] Dalal, Navneet, and Bill Triggs. "Histograms of oriented gradients for human detection." *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*. Vol. 1. IEEE, 2005.

[21] Bouguet, Jean-Yves. "Pyramidal implementation of the affine lucas kanade feature tracker description of the algorithm." *Intel Corporation* 5.1-10 (2001): 4.

[22] Lienhart, Rainer, and Jochen Maydt. "An extended set of haar-like features for rapid object detection." *Proceedings. International Conference on Image Processing*. Vol. 1. IEEE, 2002.

[23] He, Kaiming, et al. "Spatial pyramid pooling in deep convolutional networks for visual recognition." *IEEE transactions on pattern analysis and machine intelligence* 37.9 (2015): 1904-1916.

[24] Girshick, Ross. "Fast r-cnn." *Proceedings of the IEEE international*

conference on computer vision. 2015.
[25] Ren, Shaoqing, et al. "Faster r-cnn: Towards real-time object detection with region proposal networks." *Advances in neural information processing systems.* 2015.
[26] Liu, Wei, et al. "Ssd: Single shot multibox detector." *European conference on computer vision. Springer, Cham,* 2016.
[27] Everingham, Mark, et al. "The pascal visual object classes (voc) challenge." *International journal of computer vision* 88.2 (2010): 303-338.
[28] Abadi, Martín, et al. "Tensorflow: A system for large-scale machine learning." *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16).* 2016.
[29] Bradski, Gary, and Adrian Kaehler. *Learning OpenCV: Computer vision with the OpenCV library."* O'Reilly Media, Inc.", 2008.

저 자 소 개

김 기 철

- 2017년 2월 : 한밭대학교 전파공학과 (학사)
- 2019년 2월 : 한밭대학교 멀티미디어공학과 (공학석사)
- ORCID : https://orcid.org/0000-0002-2091-4841
- 주관심분야 : 비디오 부호화, 영상처리, 컴퓨터 비전, 딥 러닝

손 소 희

- 2015년 2월 : 한밭대학교 멀티미디어공학과 (학사)
- 2017년 2월 : 한밭대학교 멀티미디어공학과 (공학석사)
- 2017년 9월 ~ 현재 : 한밭대학교 정보통신전문대학원 멀티미디어공학과 박사과정
- ORCID : http://orcid.org/0000-0003-2499-492X
- 주관심분야 : 비디오 부호화, 영상처리, 패턴인식, 기계학습

김 민 섭

- 2018년 2월 : 한밭대학교 정보통신공학과 (학사)
- 2018년 3월 ~ 현재 : 한밭대학교 정보통신전문대학원 멀티미디어공학과 석사과정
- ORCID : https://orcid.org/0000-0003-4877-6388
- 주관심분야 : 영상처리, 컴퓨터 비전, 딥 러닝

전 진 우

- 2015년 : 한국과학기술원 전기및전자공학과 학사
- 2017년 : 한국과학기술원 전기및전자공학과 석사
- 2017년 ~ 현재 : 한국전자통신연구원 연구원
- ORCID : https://orcid.org/0000-0001-9934-1187
- 주관심분야 : 영상처리, 기계학습, 패턴인식, 안티드론

저 자 소 개

이 인 재

- 1997년 : 성균관대학교 전자공학과 학사
- 2001년 : 성균관대학교 전기전자및컴퓨터공학과 석사
- 2001년 ~ 현재 : 한국전자통신연구원 책임연구원
- ORCID : https://orcid.org/0000-0002-1975-1838
- 주관심분야 : 안티드론, 영상처리, 기계학습, 멀티미디어

차 지 훈

- 1993년 : 명지대학교 전자계산학과 학사
- 1996년 : 플로리다공과대학교 전자계산학과 석사
- 2002년 : 플로리다공과대학교 전자계산학과 박사
- 2003년 ~ 현재 : 한국전자통신연구원 무인자율운행연구그룹 책임연구원/그룹장
- ORCID : http://orcid.org/0000-0002-5257-014X
- 주관심분야 : 드론 자율비행, 안티드론, 영상처리, 패턴인식

최 해 철

- 1997년 : 경북대학교 전자공학과 학사
- 1999년 : 한국과학기술원 전기및전자공학과 석사
- 2004년 : 한국과학기술원 전기및전자공학과 박사
- 2004년 9월 ~ 2010년 2월 : 한국전자통신연구원(ETRI) 방송미디어연구부 선임연구원
- 2010년 3월 ~ 현재 : 한밭대학교 정보통신공학과 교수
- ORCID : http://orcid.org/0000-0002-7594-0828
- 주관심분야 : 영상처리, 비디오 부호화, 패턴인식, 기계학습