YOLOv3 알고리즘을 이용한 실시간 낙상 검출

(Real-time fall detection using YOLOv3 algorithm)

이현동, 윤종연, 최수정
동국대학교 컴퓨터공학전공
(Computer Science and Engineering, Dongguk University)

요약

본 논문은 인적이 드문 장소에서 낙상 사고가 발생하는 경우 신고가 어려워 후속 조치가 잘 되지 않는 문제를 해결하고자 한다. 기존에 실시간 낙상 검출 프로그램에서 더 나아가 실시간으로 낙상과 관련된 bbox 정보를 이용해 낙상 상황인지 판단하고 그 후 응급 상황인지 판단하여 알림을 발생시켜 조치를 취할 수 있는 프로그램을 만들고자 하였다. 실험 결과 CCTV 영상에 대해 제시한 알고리즘에 맞게 낙상 상황과 응급 상황을 판별한 후 알맞은 알림을 발생시키는 것을 확인할 수 있었다.

1. 서론

연간 낙상사고로 인한 구급이송 환자는 평균 5만 7천여 건이고, 그 중 50대 이상이 70%에 해당한다.[1] 낙상사고로 인한 응급환자 발생 시 빠른 후속조치를 취할 경우 생존율이 아무 조치를 하지 않았을 때보다 최대 3.3배 이상 나타난다.[2] 이처럼 낙상사고는 빠른 응급처치가 중요하고, 많은 사람들이 이 문제를 풀기 위해 수많은 연구를 진행하였다.[3]

객체인식 알고리즘 RCNN이 등장한 이후부터 지속적으로 실시간 낙상 검출 프로그램을 만들기 위한 노력들이 이루어지고 있다.[4] 이에 본 논문에서는 기존의 낙상 판별 시스템에서 추가적으로 실시간 낙상자 판별과 후속처리 문제를 해결하고자 한다.

2. zinnia 기법

실시간 낙상검출 프로그램 'Zinnia'[5]를 낙상 상황 인식, 응급 상황 판단 및 후속조치 부분으로 나눠 소개한다. 아래 표 1은 본 논문에서 자주 사용되는 용어들에 대한 설명을 정리한 것이다.

표 1. 용어 설명

단어	설명
yolo	딥러닝 기반의 객체검출 알고리즘
bbox	boundary box

아래 그림 1은 해당 기법의 flow chart를 나타낸 것이다. zinna기법은 사람을 인식하고, 해당 사람이 넘어지는 경우 응급 상황인지 판단하여 알림을 주는 방식으로 진행된다.

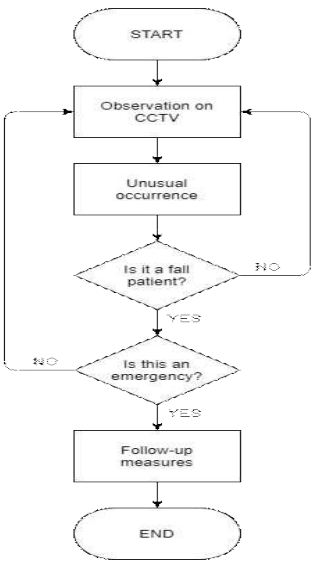

그림 1. Zinnia's flow chart

2.1 낙상자 인식

낙상자를 인식하기 위하여 CCTV에서 실시간으로 데이터를 받아온 뒤, 객체를 검출한다. 사람이 아닌 경우 객체로 인식하지 않고 사람이 나오면 person으로, 넘어지는 경우 falling_person으로, 완전히 넘어지면 fallen_person으로 인식한다. falling_person 상태를 지나 fallen_person이 되면 정상적인 낙상으로 판별하고, falling_person을 지나지 않고 fallen_person이 되는 경우 비정상적인 낙상으로 판별하였다. 두 가지 경우로 나눈 후 후속조치 과정에서 차이를 두었다.

2.2 응급 상황 판단 및 후속조치

낙상자가 인식된 후 스스로 다시 일어나는 상황과 일정시간이 지나도 일어나지 못 하는 응급 상황으로 분류하기 위해 학술 자료 [6]를 참고하여 5초의 시간으로 응급성 여부를 판단하였다. 혼자 있는 낙상자가 5초 동안 일어나지 못하면 응급 환자라고 판단한다. 이를 알고리즘화 하면 fallen_ person 객체가 5초 동안 유지될 경우 응급 환자라고 판단하는 것이다. 응급 환자라고 판단되면 이벤트를 발생시켜 Zinnia를 실행하고 있는 PC에 알림이 준다.

3. 실험

3.1 낙상자 인식 부분

낙상과 관련된 bbox로 인식이 되는 경우 실제 낙상 사고 여부를 잘 판별하는지 2가지 경우로 나누어 실험해보았다.

실험 환경 1) falling → fallen 인 경우

정상적인 낙상으로 판별되어야 하는 falling_person에서 fallen_person으로 넘어가는 영상을 실험해보았다. 그림 2는 객체를 falling_person으로 인식된 상태를 보여준다. 일정 시간 뒤 해당 객체가 그림 3과 같이 fallen_person으로 인식되게 되고 그림 4와 같이 정상적인 낙상으로 판별되는 것을 확인할 수 있다.

그림 2. falling_person 상태

그림 3. fallen_person 상태

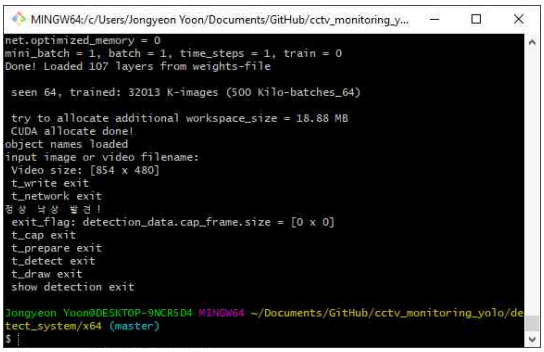

그림 4. 낙상 판별 결과 화면

실험 환경 2) falling X → fallen 인 경우

이번에는 비정상적인 낙상으로 판별되어야 하는 falling_person 상태를 거치지 않고 fallen_person으로 인식되는 경우를 실험해보았다. 그림 5와 같이 person으로 인식되는 상황에서 그림 6처럼 바로 fallen_person으로 인식되게 되면 그림 7과 같이 비정상적인 낙상으로 판별되는 것을 확인할 수 있다.

그림 5. falling_person이 아닌 상태

그림 6. fallen_person 상태

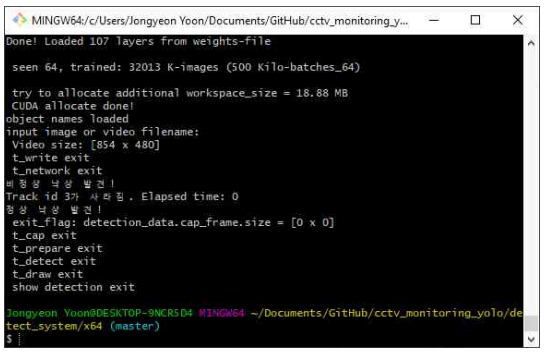

그림 7. 낙상 판별 결과 화면

3.2 응급 상황 판단 및 알림 부분

낙상자가 감지되었을 때, 주위 환경에 맞게 응급 상황 여부를 판단하는지, 응급 상황에서 알림이 전송되는지를 실험해보았다.

실험 환경 1) 정상적 낙상 + 일어남

정상적인 낙상이 일어나는 경우 넘어진 시점부터 지속 시간을 측정한다. 테스트 환경에서의 긴급 상황 기준인 5 초가 되기 전에 객체가 fallen_person 상태를 벗어나는 경우 알림을 주지 않고 넘어간다. 실험 결과는 그림 8에서 알 수 있듯이 처음에 정상적인 낙상임을 안내하고, 기준 시간보다 지속시간이 짧은 경우 응급 상황 안내 없이 해당 객체에 대한 안내를 종료하게 된다.

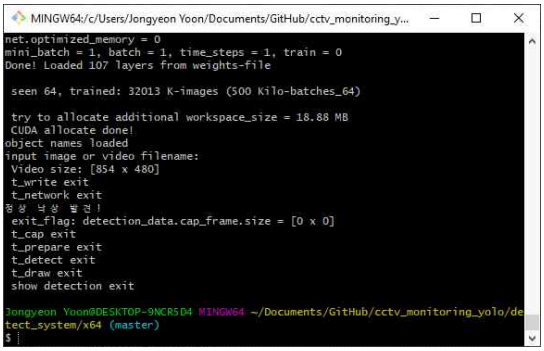

그림 8. 실험 1 결과 화면

실험 환경 2) 정상적 낙상 + 못 일어남

정상적인 낙상이 일어난 후 기준 시간 이상 fallen_person상태가 지속되는 상황을 실험해보았다. 그림 9에서 기준 시간을 넘기는 경우 지속 시간을 표시하며 응급상황임을 안내해 주는 것을 확인할 수 있다.

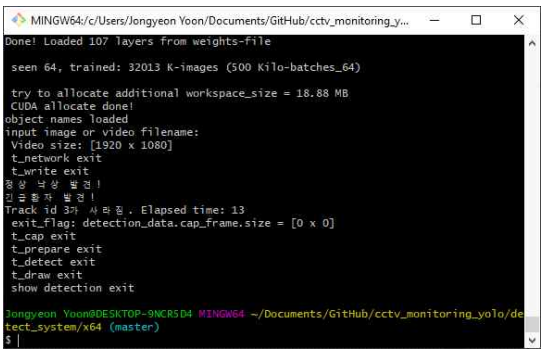

그림 9. 실험 2 결과 화면

실험 환경 3) 비정상적 낙상

비정상적인 낙상 상태에 대한 후속처리를 실험해보았다. 그림 10에서 알 수 있듯이 비정상 낙상 상태 또한 fallen_person으로 인식하고 기준 시간을 넘어가면 응급 상황으로 안내하는 것을 확인할 수 있다.

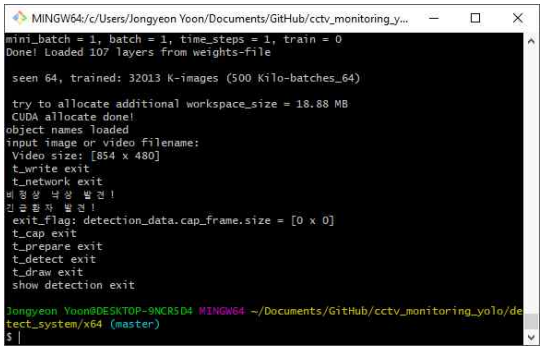

그림 10. 실험 3 결과 화면

4. 결론

Zinnia 프로그램을 통해 CCTV 영상을 확인해본결과, 낙상과 관련된 bbox 정보를 활용해 낙상 여부를 알맞게 판단하고 이에 대한 알림을 발생시키는 것을 확인할 수 있었다. 또한 단순히 낙상만 검출하는 것이 아니라 알고리즘을 통해 낙상이 발생한 후 지속 시간을 기준으로 응급상황인지를 판별하는 결과를 확인할 수 있었고 이에 알맞은 알림을 발생시켜 주는 것을 확인할 수 있었다.

5. Materials and methods

5.1 Method

프로그램 실행 시 낙상자 관련 bbox가 인식되면 정상적인 낙상 상황인지 판단하고 지속 시간을 측정하여 이를 기준으로 이벤트를 발생시키도록 하였다.

5.2 Materials-dataset

YOLOv3를 통해 객체를 검출하기 위해 darknet과 OpenCV를 이용하였다. darknet 솔루션 파일을 빌드하여 .exe 파일을 만들기 위해 darknet 프로젝트의 속성과 PC의 환경변수에 OpenCV를 추가하여 경로를 설정해 주었고, GPU를 사용하여 작업하기 위해 CUDA와 cuDNN도 설치하여 백그라운드를 설정하였다.

학습된 YOLOv3 파일을 VS2015 C++에 import하여 원하는 내용으로 코드를 수정하기 위해 .cfg 파일을 .dll파일로 변환하는 작업을 진행하였고 이를 통해 낙상자가 검출되면 이벤트를 발생시키는 코드를 구현하였다.

5.3 Implementation

Zinnia의 개발을 위해 OpenCV, Darknet, visual studio, Python, YOLO-mark 등의 툴을 사용하였다.

OpenCV와 Darknet에서는 YOLO 알고리즘을 실행하기 위한 백그라운드를 설정하였고, 객체 검출 코드의 작성은 visual studio의 C++언어를 사용하였다.

- 툴 다운로드 링크

1. OpenCV :
https://opencv.org/releases/
2. Darknet :
https://github.com/AlexeyAB/darknet
3. Visual Studio :
https://visualstudio.microsoft.com/ko/vs/?WT.mc_id=DT-MVP-4038234
4. Python :
http://www.python.org/downloads
5. YOLO-mark :
https://github.com/AlexeyAB/Yolo_mark

6. 향후 계획

기존 Zinnia 프로젝트는 자체적인 데이터 학습 과정을 진행한 후 후속 처리까지의 구현을 목표로 했지만, 데이터 학습이 잘 이루어 지지 않아 직접 bbox 정보를 입력하여 프로젝트를 진행하였다. 이에 데이터 학습에 대한 연구를 더욱 진행하여 weight파일을 이용한 객체 검출도 진행할 예정이다.

또한 현재 Zinnia는 Object를 detecting함에 있어서 bbox가 객체를 완벽하게 따라가지 못 하는 문제가 있다. 이에 낙상자 관련 영상의 프레임을 더욱 자세하게 나눠 최대한 bbox가 객체를 잘 따라갈 수 있도록 진행할 계획이다.

7. 프로젝트 링크

Zinnia에 대한 모든 코드와 자세한 프로젝트의 내용은 아래 GitHub에서 확인할 수 있다.

https://github.com/0417yjy/cctv-monitoring-yolo

8. Acknowledgement

8.1 Thanks to

1. 자문 교수 :
 동국대학교 컴퓨터공학전공 신연순 교수님
2. CCTV 영상 협조 :
 동국대학교 스마트 캠퍼스
3. 프로젝트 진행 비용 지원 :
 동국대학교 링크사업단

9. 참고문헌

[1] 김정욱,「낙상 사고 12월에 가장 많아... 낙상 환자 중 50대 이상이 70%」, 서울경제, 2019.12.03, pp. 1.
https://www.sedaily.com/NewsVIew/1VRXJI2624

[2] 전진우,「"심장마비 환자 심폐소생술부터" ... 생존율 최대 3.3배」, 동아닷컴, 2019.11.26, pp. 1.
https://www.donga.com/news/Society/article/all/20191126/98538887/1

[3] 최형진, 김정수,「낙상사고 예방을 위한 국내・외 관련 규정 분석 연구」, RISS 학술연구정보서비스, 2014, pp. 1.
http://www.riss.or.kr/search/detail/DetailView.do?p_mat_type=1a0202e37d52c72d&control_no=7b3887969f528d527f7a54760bb41745

[4] 임세경,「국토교통 R&D Report 오픈소스 하드웨어 기반 실내 환경 모니터링 시스템 개발 및 클라우드를 통한 지속적 자가 학습 딥러닝 시스템 구현」, 국토교통부 (국토교통과학기술진흥원), 2016.06.24, pp. 1.
https://www.kaia.re.kr/portal/landmark/readTskFinalView.do?tskId=114867&yearCnt=2&menuNo=200100

[5] Zinnia는 쌍떡잎식물 백일홍[百日紅]의 영어 이름으로 낙상과 관련된 설화를 갖고 있어 이번 프로젝트의 이름으로 정하였다.

[6] 장덕성, 최승찬, 선주형, 김상현, 한송희,「응급호출기의 개발과 낙상판단 알고리즘」, 한국HCI학회, 2011.01, pp. 1.
http://www.dbpia.co.kr/Journal/articleDetail?nodeId=NODE01880334