객체인식을 활용한 잔류인원 추정 시스템

이성우[O], 이경형[*], 석진훈[*], 김경섭[*], 전민서[*], 추승오[*], 윤태진(교신저자)[*]
[O]경운대학교 항공소프트웨어공학과,
[*]경운대학교 항공소프트웨어공학과
e-mail: {dnsid28[O], rudgud789456, wlsgns00724, rudtjq9267, jmsparrowk, sam014789}@naver.com[*],
tiyun@ikw.ac.kr[*]

Remaining persons estimation system using object recognition

Seong-woo Lee[O], Gyung-hyung Lee[*], Jin-hoon Seok[*], Kyeong-seop Kim[*], Min-seo Jeon[*],
Seung-oh Choo[*], Tae-jin Yun(Corresponding Author)[*]
[O]Dept. of Aeronautical Software Engineering, Kyungwoon University,
[*]Dept. of Aeronautical Software Engineering, Kyungwoon University

● 요 약 ●

재해, 재난 발생 시에 구조대가 건물 내부나 지하철 등, 특정 구역 내의 대피하지 못한 잔류인원을 제대로 파악하데 어려움을 겪는다. 이를 개선하고자 YOLO와 DeepSORT를 활용하여 통행자를 인식하여 특정 구역의 잔류인원을 파악하고 이를 서버를 통해 확인할 수 있는 시스템을 개발하였다. 실시간 객체인식 알고리즘인 YOLOv4-tiny와 실시간 객체추적기술인 DeepSORT 알고리즘을 이용하여 제안한 방법을 Ubuntu환경에서 구현하고, 실내 상황에 맞춰 통행자 동선을 고려해서 적용하였다. 개발한 시스템은 인식된 통행자 객체 방향으로 출입을 구분하여 데이터를 서버에 저장한다. 이에 따라 재해 발생 시 구역의 잔류인원을 파악하여 빠르고 효율적으로 요구조자 위치와 인원을 예측할 수 있다.

키워드: 딥러닝(Deep Learning), 객체인식(Object Recognition), 객체추적(Object Tracking)

I. Introduction

실내의 재난, 재해 발생 시 많은 인명피해가 지속해서 발생하고 있다. 이런 인명피해를 줄이기 위해 구조자의 위치를 파악하는 여러 기술이 개발, 확대되고 있다. 이에 따라 소방청 등의 기관에서는 구조자 파악을 위해 장소의 특징에 따라 명부를 확인하거나 직접 탐색하거나 드론을 통한 탐색을 시도하고 있지만, 시간이 지연되거나 드론의 경우 활용인력이 부족하고 비용의 문제점이 있다.

딥러닝을 통해 구역 내부의 잔류인원을 주기적으로 파악, 저장하면서 재난 발생 시 이전의 잔류인원을 조회함으로써 구조자의 인원과 대략적인 위치를 파악하여 요구조자 구출에 도움을 줄 수 있다.

II. Preliminaries

제안한 시스템은 Ubuntu 18.04 환경에서 GPU 개발 툴인 Cuda 10.2와 CUDA 딥러닝 라이브러리인 Cudnn 8.6.0, 영상 처리에 사용되는 OpenCV 4.2.0 라이브러리를 통해 개발하였고 실시간 객체 인식 알고리즘인 YOLOv4-tiny와 객체 추적 알고리즘인 DeepSORT를 활용하였다[1,2].

본 논문에서 사용하는 YOLOv4-tiny 알고리즘은 가중치 파일을 요구한다. 가중치 파일 생성을 위해 국내 지하철 내부를 대상으로 촬영된 사람 이미지와 어린이 이미지를 한국지능정보사회진흥원이 운영하는 AI Hub를 통해 12,000장 가량 획득하였고 인식 객체 클래스는 Person과 Child로 구성하였다.

III. Design and Development

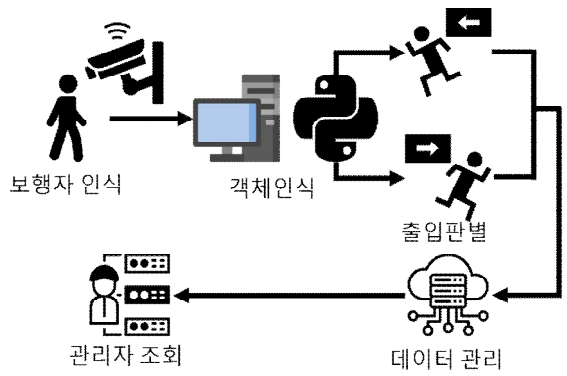

Fig. 1. Proposed system diagram

그림1은 제안한 시스템의 구성으로 카메라를 통해 획득한 영상을 엣지 컴퓨터로 전송하고 객체 인식을 진행한다. 객체가 인식된 후 해당 객체의 출입을 판별하고 인식된 객체 카운트 데이터를 저장한 뒤 관리자가 이를 웹으로 조회할 수 있는 시스템을 설계하였다.

특정 구역을 출입구에 사람의 출입을 판별, 비교하여 traffic count를 통해 남은 인원수를 파악, 조회한다.

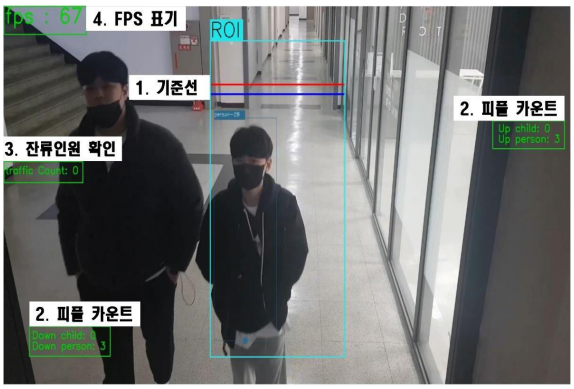

Fig. 2. Proposed system screen

그림2의 표기된 (1) 기준선을 객체가 지났을 때, 어느 기준선에 먼저 닿았는지에 따라 출입을 판별하고 판별된 데이터는 (2) 피플 카운트로 저장되며 화면에 표기된다. 이때 (3) 잔류인원 확인에서 (2) 피플 카운트의 차를 계산하여 잔류인원으로 표기한다. 화면의 (1) 기준선을 통과할 때 인식된 객체의 클래스, 시간, 출입정보를 데이터베이스에 저장한다.

제안한 시스템에서는 YOLO tiny 모델을 선정한 만큼 초기에도 평균 21 FPS(Frame Per Second)의 수치가 나오는 것을 확인하였으나, 인식할 객체의 수나 영상의 이미지 크기 등에 따라 문제가 될 수 있어 시스템 처리 속도를 높여 FPS를 향상시키도록 제안한 방법으로 ROI(Region of Interest), 이미지 resizing, 스레드 구조를 이용한 처리 등을 적용하였다.

ROI는 관심 영역을 설정하는 기술로 그림2와 같이 설정된 영역 밖의 객체는 인식을 진행하지 않아 분석 시간과 데이터양을 줄여 FPS를 높이는 데 기여한다.

이미지 resizing은 영상의 크기가 너무 커 이미지 처리에 지연이 생길 경우를 대비해, 임의로 지정한 비율대로 이미지의 크기와 좌푯값을 조정한다.

스레드는 Video_capture, Inference, Drawing으로 3개로 구성하였다. 차례대로 Video_capture는 영상의 이미지 프레임 생성과 ROI 영역 지정, 이미지 resizing을 진행하고 Inference 스레드는 실질적인 객체 추적과 FPS 계산을 진행한다. 마지막 Drawing 스레드는 객체를 카운팅함과 동시에 객체의 정보를 데이터베이스로 전송한다. 이런 스레드 구조를 통해 한 스레드가 지연되더라도 다른 스레드들이 계속 작업을 진행하게 설계하여 전체적인 지연을 줄였다.

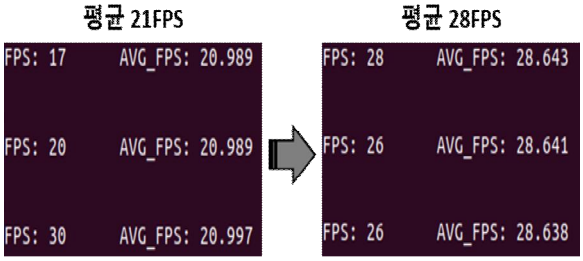

Fig. 3. Improvements of FPS

ROI, 이미지 resizing, 스레드와 같은 기술을 통해 그림3과 같이 평균 21 FPS에서 28 FPS로 성능이 향상된 것을 실험을 통해 확인하였다.

IV. Conclusions

본 논문에서는 특정 구역 내 잔류인원을 추정하고 인원을 주기적으로 파악하여 재난 시 빠르게 잔류인원을 파악할 수 있는 시스템을 제시하였다.

제안한 시스템에서는 향후 여러 개의 출입구에 설치된 카메라들을 연동하여 정보를 공유하는 방법을 고려한 방안을 확장해서 제안할 수 있다. 또한, 이 시스템을 적용하여 유통 매장, 전통시장 등의 방문자 통계를 통해 다양한 통계 정보를 얻어 활용할 수 있다.

REFERENCES

[1] AlexeyAB darknet “https://github.com/AlexeyAB/darknet”
[2] TheAiguysCode deepSORT “https://github.com/theAIGuysCode/yolov4-deepsort”