차로별 좌표계 변환에 의한 차량 속도 추정 기반 사고 위험 상황 분류

Classification of accident risk situations based on vehicle speed estimation through lane-specific coordinate transformation

김승현, 강한빈, 강종규, 배창석[1)]
Seung Hyun Kim, Han Bin Kang, Jong Gyu Kang, Changseok Bae

(34520) 대전광역시 동구 대학로 62, 대전대학교 정보통신공학과
{ryan7115, strong_b7326, dkffk149, csbae}@edu.dju.ac.kr

요 약

일반적인 도로 환경에 비해 도심 이면 교차로는 사고 위험이 특히 높은 것으로 알려져 있다. 본 논문에서는 이러한 도심 이면 교차로에 설치한 CCTV 영상으로부터 위험 상황을 인지하는 모델을 제안한다. 제안하는 모델은 크게 다음 2가지 특징을 가진다. 먼저, 교차로의 경우 차로별로 카메라 시점에 따른 왜곡 방향이 다르기 때문에 진행 차량의 정확한 속도 추정을 위해서는 차로별로 독립적인 BEV (Bird's Eye View)로의 좌표계 변환이 필요하다. 다음으로는 기존의 동영상 분류 방법보다 우수한 성능으로 위험 상황을 인지하기 위해 속도 추정 결과를 바탕으로 교차로에서의 위험 상황을 판단한다. 제안하는 모델의 성능을 검증하기 위해 교차로 모의 실험 환경을 구축하고 교통사고 영상 데이터셋을 수집한다. 영상 데이터는 '안전', '주의', 그리고 '위험'으로 총 세 가지 교차로 상황의 클래스로 구분하고, 학습 및 성능 검증을 위해 각 클래스 별로 72개의 영상 데이터를 선별하여 사용한다. 논문에서 제안하는 위험 상황 인지 판단 모델 학습을 위해 YOLO v8 모델을 사용하여 영상 속 차량의 이동 궤적을 추적하고, 카메라 화각에 따른 왜곡을 보정하기 위해 차로별 독립 BEV 변환을 수행하고 차량의 위치와 속도를 추정한다. 제안하는 모델의 현실 세계 적용 가능성을 확인하기 위해 실제 도로에서의 차량 주행 영상을 바탕으로 속도를 추정한 실험 결과를 함께 제시하고 있다. 본 논문에서는 추정한 차량의 속도와 위치를 기반으로 MLP (Multi-Layer Perceptron) 분류기를 적용한 위험 상황 인지 모델을 제안한다. 또한, 제안하는 모델의 성능 평가를 위한 비교군으로 추정한 차량의 속도와 위치를 기반으로 GRU (Gate Recurrent Unit) 분류기를 적용한 모델과 속도 추정없이 Inception v3 기반의 비디오 분류 모델을 사용하여 제안 모델을 평가한다. 제안하는 모델과 기존 모델의 성능을 비교한 본 논문의 실험 결과 YOLO와 MLP를 사용한 모델이 89%의 정확도와 5ms의 추론 속도를 보여주어 정확도와 실시간성 측면에서 다른 비교 모델인 YOLO와 GRU 기반의 모델과 비디오 분류 모델에 비해 가장 뛰어남을 확인할 수 있다.

※ 본 과제(결과물)는 2024년도 교육부의 재원으로 한국연구재단의 지원을 받아 수행된 지자체-대학 협력기반 지역혁신 사업의 결과입니다.(2021RIS-004)

1) 교신저자

논문투고일자 : 2024. 01. 15 심사일자 : 2024. 01. 26 게재확정일자 : 2024. 12. 27 http://doi.org/10.23019/kingpc.21.1.202502.001

Abstract

Compared to typical road environments, urban backstreet intersections are known to have a particularly high risk of accidents. This paper proposes a model for recognizing risk situations from CCTV video installed at such urban backstreet intersections. The proposed model has two main features. First, since the distortion direction due to the camera viewpoint varies by lane at intersections, lane-specific independent coordinate transformations to Bird's Eye Views (BEV) are necessary for accurate vehicle speed estimation. Second, to detect risk situations with better performance than existing video classification methods, the model determines risk situations at intersections based on the estimated speed results. To validate the performance of the proposed model, a simulated intersection environment was constructed, and a traffic accident video dataset was collected. The video data are classified into three intersection situations: 'safe', 'caution', and 'danger', with 72 video samples selected per class for training and performance validation. For training the risk detection model proposed in this paper, YOLO v8 is employed to track vehicles in the videos. Then, lane-specific independent BEV transformations are performed to correct distortion based on the camera angle. After that, the model can estimate vehicle positions and speeds. To confirm the real-world applicability of the proposed model, experimental results based on actual road driving video are presented, where vehicle speed estimation is demonstrated. Utilizing the estimated speeds and positions, a hazard recognition model based on a Multi-Layer Perceptron (MLP) classifier is proposed. For performance evaluation, the model is benchmarked against a Gate Recurrent Unit (GRU)-based model and an Inception v3-based video classification model. The experimental findings reveal that the YOLO and MLP-based model achieves an accuracy of 89% and an inference speed of 5ms, demonstrating superior performance in both accuracy and real-time processing compared to the GRU-based and video classification models.

키워드: 교차로, 교통 위험, 단일 카메라, 동적 객체 검출, 동적 객체 속도 추정, 위험 상황 인지
Keyword: Intersection, Traffic risk, Single camera, Dynamic object detection, Dynamic object speed estimation, Risk situation recognition

1. 서론

현대 사회는 다양한 교통수단으로 편리함을 추구하고 있다. 그러나 이러한 교통수단의 치명적인 문제점은 교통사고로 인해 매년 30만 명에 가까운 사람이 다치고 3천 명 규모의 사람이 목숨을 잃고 있다는 사실이다.

경찰에서 조사, 처리한 교통사고 정보 [1]에 따르면 2023년 교통사고 건수는 196,836건이고, 이 중 교차로에서 발생한 사건 수가 95,354건으로 약 48.44%로 절반을 차지했다. 교차로도 교차로 형태에 따라서 몇 가지 종류로 분류할 수 있는데, 그 중 삼지 (三枝), 사지 (四枝) 교차로에서 발생한 사건 수가 92,168건으로 약 46.82%이다. 그 이외의 회전이나 오지 (五枝) 이상의 교차로에서 발생한 사건이 약 1.61%이고, 나머지는 단일로에서 발생한 교통사고이다.

교통사고를 예방하기 위한 인공지능과 빅데이터를 활용한 사고 예측 및 예방 기술이 주목받고 있으며 [2, 3], 스마트 교차로 시스템 [4], 운전자 보조

시스템 [5] 등 다양한 방법들이 연구되고 있다 [6-8]. 특히 운전자 보조 시스템의 한계를 인지함에 따라 교통 인프라를 활용하여 이를 보조하는 해결 방안을 찾고 있다 [9, 10]. 이에 따라서 교통을 관망하는 센서나 카메라를 설치하고 교통정보를 수집하여 운전자들에게 제공하는 연구까지 이어지고 있다 [2, 4, 9, 10].

본 논문에서는 이러한 현재 상황에 주목하여 교차로에서의 교통사고를 경감하는 문제를 다루고자 한다. 이러한 문제를 해결하기 위해 교차로 환경에 설치한 카메라로부터 입력되는 각 차로별 영상을 분석하여 위험 상황을 인지하는 모델을 제안한다. 교차로에 진입하는 차로별 차량의 속도로부터 위험 상황을 인지할 수 있다는 가정에서 위험 상황 인지 모델을 제안하고 있다. 먼저 차로별로 각각 입력되는 영상으로부터 이동 차량을 탐지하고 객체의 이동 궤적을 추적하여 차량의 이동 속도를 추정한다. 교차로에 진입하는 차량의 속도에 따라 단순한 분류기 모델로도 사고 위험도를 우수한 성능으로 예측할 수 있다. 본 논문에서는 사고 위험도를 “안전”, “주의”, 그리고 “위험”의 3가지 단계로 구분하여 예측한다.

본 연구를 기반으로 교통 인프라 서비스를 개발하면 시야 제한과 운전자의 운전 미숙으로 인해 발생하는 사고를 예방하여 사고율 및 사망률을 줄이는데 기여할 수 있다. 또한, 2차 사고를 예방하거나 차량 흐름을 원만하게 하는 등 통합 지능형 교통 관제 시스템의 주요한 요소로 활용 가능하다.

2장에서는 관련된 영상 분석 기술과 본 논문에서 적용한 딥러닝 기술에 대해 알아보고, 3장에서 제안하는 위험 상황 인지 모델 구조를 설명한다. 4장에서 실험 결과에 따른 모델 성능을 보여주고, 마지막으로 5장에서는 결론 및 향후 연구 방향에 대해 논의한다.

2. 관련 연구

본 장에서는 먼저, 교차로 내에서의 상황을 분석하는 시스템과 지능형 영상 분석 기술을 살펴본다. 이후 관련 기술을 활용하여 제안하는 도심 이면 교차로 내 위험 상황 인지 모델을 설명한다.

2.1 도로 환경 영상 분석 시스템 관련 기술

영상 기반의 분석 기술을 통한 도로 내 위험 상황 분석 시스템으로는 크게 두 가지 방법이 있다.

첫 번째로는 객체 인식 및 추적 기반의 위험 상황 판단 알고리즘 [11-13]이다. 기존 객체 인식 기반의 위험 분석 시스템 [14]에서는 차량의 위치나 속도에 대한 추정은 없이, GNSS (Global Navigation Satellite System)에서 변환된 좌표를 이용하여 위험 상황을 판단한다. 하지만 이런 시스템은 GNSS의 성능에 의지해야 하며, 성능이 저하되는 골목길이나 도심에서는 GNSS 외의 다른 방안이 필요하다. 또한 다른 연구 [15]에서는 CCTV (Closed-Circuit Television) 영상의 화각 및 설치 방향이 교통량 측정 및 방범을 목적으로 설치하였기 때문에 교차로에서 다방향의 이동 객체의 인식 및 추적과 상황 판단이 어려운 문제를 가진다.

두 번째로는 CNN (Convolution Neural Networks) 기반의 네트워크로 영상의 특징을 추출하고, 추출된 프레임별 특징의 시계열 분석 기반의 방법이 있다 [16, 17]. 이 방법에서 사용하는 영상 데이터는 사고 판단을 주목적으로 하여 블랙박스 영상으로부터 대부분 제작되었으며, 실시간 영상은 입력으로 사용하지 않는다.

2.2 차량 검출, 추적 및 위험 상황 인지 관련 기술

지능형 영상 분석 기술은 CCTV와 같은 카메라를 통해 촬영되는 영상을 인공지능으로 영상 속 정보들을 수집하고 파악하는 기술이라 할 수 있다. 본 절에서는 차량의 검출 및 추적과 관련된 딥러닝 기반의 지능형 영상 분석 기술을 살펴보고자 한다.

2.2.1 YOLO (You Only Look Once)

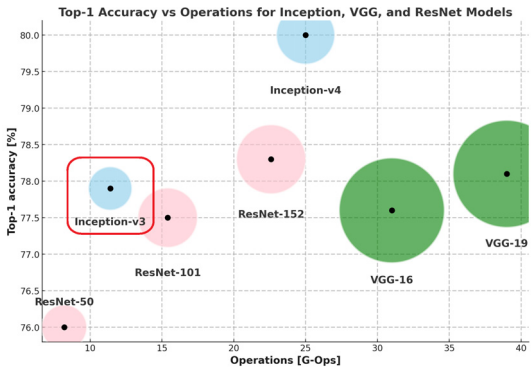
본 논문에서 사용하는 가장 기본적인 영상 분석 기술은 영상 내에 존재하는 객체를 탐지하고 추적

하는 기술이라 할 수 있다. 딥러닝 기반의 객체 탐지 모델은 단계에 따라 크게 두 가지로 분류할 수 있다. 물체의 위치를 발견하는 문제와 발견한 물체가 무엇인지 식별하는 문제를 순차적으로 해결하는 Two-stage 방법과 이 두 가지 문제의 해결을 동시에 수행하는 One-stage 방법이 있다 [18].

각 방법에 따라 장단점이 존재하는데, Two-stage 방법은 높은 정확도를 보여주지만 느린 처리 속도로 인해 실시간 처리에 부적합하며 대표적인 모델로는 R-CNN (Region-based Convolution Neural Network), Fast R-CNN, Faster R-CNN [19-21] 등이 있다. One-stage 방법은 Two-stage 방법에 비해 낮은 정확도를 보여주지만 빠른 처리 속도를 통해서 실시간 처리에 적합하며 대표적인 모델로는 YOLO(You Only Look Once), SSD(Single Shot multi box Detector) [22] 등이 있다. 본 논문에서는 단일 고정 카메라 영상을 통해 동적 객체가 실시간으로 탐지되어야 한다. 따라서 One-stage 방법의 모델 중 하나인 YOLO [23]를 채택하고 있다.

2.2.2 Inception-v3

Inception-v3 [24]는 Google에서 개발한 딥러닝 모델로, 33, 22 합성곱 연산으로 분해하는 것에 비해 13, 31 비대칭 합성곱의 분해를 통해서 연산 파라미터를 효과적으로 감소시켰다. 또한 Pooling layer와 Convolution layer를 병렬로 사용하고, 둘을 연결하여 표현력을 감소시키지 않으면서 연산량을 감소시킬 수 있다. (그림 1)은 여러 특징 추출 모델의 성능을 분석한 연구 결과이다. Inception-v3 모델은 Top-1 Accuracy 부분에서 다층 ResNet과 비슷한 성능을 보여주지만, 처리 속도를 결정하는 Operations는 비교군 대비 낮아 웹캠을 통해 얻은 실시간 영상에서 빠른 속도로 특징을 추출하기에 적합하다.



(그림 1) Inception-v3의 성능 지표 [25]

2.2.3 MLP(Multi-Layer Perceptron)

기계 학습에서 사용되는 전통적인 분류기 모델은 MLP(Multi-Layer Perceptron), SVM(Support Vector Machine), Decision Tree, K-NN(K-Nearest Neighbors) 등 다양하게 있는데, 본 논문에서는 이 중 MLP [26] 분류기 모델을 활용하여 성능을 분석하고, 하이퍼파라미터를 조정하여 최적의 모델을 찾는다.

MLP는 여러 개의 은닉층을 가지고 있는 다층 구조의 인공 신경망의 일종으로 입력층, 은닉층, 출력층, 활성화 함수, 역전파 알고리즘으로 구성되어 있다. ReLU(Rectified Linear Unit)와 같은 비선형 활성화 함수를 사용하여 비선형 문제를 해결할 수 있고, 역전파 알고리즘을 통해 신경망의 가중치를 조절하며 학습할 수 있다. MLP 분류기 모델은 지도학습 모델로 입력값과 출력값을 통해 모델을 학습하고 새로운 데이터에 대한 예측을 할 수 있다. 이를 실험 환경에 적용하여 YOLO 모델을 통해 동적 객체의 데이터를 영상에서 수집하고, 수집한 데이터를 MLP 분류기 모델로 학습하여 위험 상황을 판단할 수 있도록 한다.

2.2.4 GRU(Gate Recurrent Unit)

시퀀스 모델은 자연어 처리와 음성인식, 시계열 데이터 분석, 비디오 처리 분야에서 CNN과 다른 접근법을 통해서 뛰어난 성능을 보여준다. 시퀀스 모델의 뿌리라고 볼 수 있는 네트워크 구조는 RNN

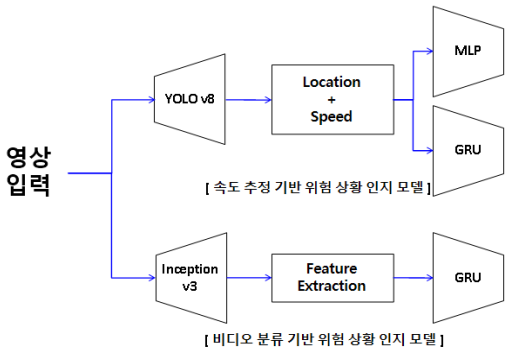
(Recurrent Neural Network) [27]인데, 시퀀스의 길이가 길어지면 기울기 소실이 발생하여 장기 기억에 취약하다.

GRU(Gate Recurrent Unit) [28]는 기존 RNN의 장기 기억 취약 문제를 해결하고자 개발된 LSTM(Long Short-Term Memory) [29]에서 더욱 발전시켜 Cell State와 Memory State를 병합한 시계열 분석 모델이다. Reset Gate는 이전 상태의 정보 유지에 대한 가중치를 설정하고, Update Gate는 이전 상태의 정보와 새로운 정보를 결합하는 역할을 한다. LSTM에 비해 빠른 학습 시간을 가지며 구조가 단순하여 매개변수가 적어 계산에 효과적이다.

3. 위험 상황 인지 모델 제안 구조

관련 연구에서 살펴본 영상 분석 기술을 본 논문의 목적에 맞게 활용 및 보완하여 영상분석 기반의 교차로 상황 인지 모델을 만들고자 한다. 본 논문에서는 교차로 환경을 가정하여 차로별 CCTV 카메라를 설치하고 이들로부터 각각 영상을 입력받는다. 교차로에 진입하는 차로별 차량의 이동 속도를 함께 고려함으로써 교차로에서의 위험 상황을 인지하는 모델을 제안한다. 성능 평가를 위해 비교군으로는 비디오 분류 기반 위험 상황 인지 모델을 둔다.

본 논문에서 사용하는 3가지 상황 인지 모델의 구조를 (그림 2)에서 보여주고 있다. 먼저, 제안하는 속도 추정 기반 위험 상황 인지 모델은 YOLO v8을 통해 객체의 위치와 속도를 추정한다. 이후 추정된 정보를 통해 MLP와 GRU, 두 가지 인공지능 분류기 모델을 사용하여 위험도를 분류한다. 비교군인 비디오 분류 기반 위험 상황 인지 모델은 비디오 분류에서 흔히 볼 수 있는 CNN+RNN의 구조를 활용하여 위험도를 분류한다.



(그림 2) 위험 상황 인지 모델 제안 구조도

단일 카메라를 통해 영상을 받고, 이를 두 가지 접근 방법(속도 추정 기반, 비디오 분류 기반)의 입력값으로 넣는다. 이후 출력되는 특징을 활용하여 인공지능 분류기 모델에 입력값으로 넣어 교차로의 세 가지 상황(안전, 주의, 위험) 중 하나로 분류하여 위험 상황을 인지할 수 있도록 한다.

3.1 속도 추정 기반 위험 상황 인지 모델

동적 객체의 인지 및 추적을 위해 단일 카메라로 들어오는 영상을 YOLO v8에 입력한다. YOLO v8은 기본적으로 MOT(Multiple Object Trackers) 중 하나인 BoT-SORT(Bag of Tricks for Simple Online and Realtime Tracking) 알고리즘을 지원하기 때문에 이를 통해 동적 객체의 특징을 추출한다. 단일 카메라를 통해 차량의 위치와 속도를 추정하기 위해서는 2차원 영상 좌표계로 표현된 객체의 특징을 3차원 좌표계로 변환하여 표현할 수 있어야 한다. 또한 카메라는 거리가 멀수록 객체의 크기와 움직임은 작고, 가까울수록 큰 원근 왜곡 현상이 발생한다. 이러한 문제를 해결하기 위해 Bird's Eye View(BEV) 변환을 활용한다 [30].

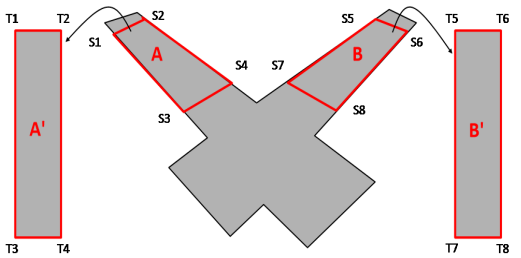
BEV 변환은 카메라의 시야를 통해 얻은 영상을 위에서 내려다보는 것처럼 보이도록 영상을 변환하는 과정으로 원근 왜곡 문제를 해결하고 차원 간의 차이를 최소화할 수 있다. 즉, 움직임이 동일한 물체가 멀리 있을 때와 가까이 있을 때가 같게 표현할 수 있다는 것이다. 3차원 좌표계를 2차원 좌표계로

변환하여 렌즈 왜곡을 보정하는 카메라 캘리브레이션 방법도 있지만, 동적 객체의 정확한 위치와 속도를 구하고자 하는 것이 아닌 실시간으로 객체의 위치와 속도를 추정하여 빠르게 위험 상황을 인지하는 것에 목적을 두고 있기에 BEV 변환을 채택한다. 또한 객체 (차량)가 동일 평면 (도로) 상에서 이동하는 상황을 고려하고 등속 직선 운동을 한다고 가정할 수 있으므로 BEV 변환을 이용하여 동적 객체의 위치와 속도를 추정할 수 있다.

위치와 속도를 기반으로 위험 상황을 인지하기 위해 YOLO를 활용하여 동적 객체를 탐지하고 추적하여 위치와 속도를 구한다. 구해진 위치와 속도를 MLP와 GRU 분류기에 입력하여 세 가지 위험 상황인 '안전', '주의', '위험' 중 하나로 분류한다.

3.1.1 위치 추정 알고리즘

실제 도로에서의 차량의 위치와 속도를 추정하기 위해서는 먼저 카메라 화각에 대한 왜곡을 보정해야 한다. 카메라 화각에 따라 왜곡이 있는 도로 원본 영상에서 BEV에서의 도로 영상으로 변환하는 과정은 (그림 3)과 같이 고려할 수 있다.



(그림 3) 차로별 독립적 BEV 변환

BEV 변환으로 변환 영상 속 객체의 위치를 나타내기 위해서는 원근 변환 행렬 M을 구해야 한다. 이를 위해 도로 원본 영상 속에서 변환하고자 하는 사각형 영역의 4개의 점을 지정하고, 변환된 영역을 보여줄 변환 영상의 4개의 점을 지정한다. 변환 과정 설명을 위해 A 영역을 지정한 4개의 점 (S1, S2, S3, S4)과 A′ 영역을 지정한 4개의 점 (T1, T2, T3, T4) 중 점 S1과 점 T1을 활용한다.

$$\begin{bmatrix} x_1' \\ y_1' \\ w' \end{bmatrix}_{\text{변환}} = \text{원근 변환 행렬}(M) \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix}_{\text{원본}} \quad (1)$$

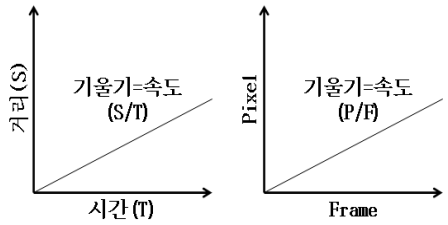
$$M = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & 1 \end{bmatrix} \quad (2)$$

$$\begin{aligned} x_1' &= \frac{h_{11}x_1 + h_{12}y_1 + h_{13}}{h_{31}x_1 + h_{32}y_1 + 1} \\ y_1' &= \frac{h_{21}x_1 + h_{22}y_1 + h_{23}}{h_{31}x_1 + h_{32}y_1 + 1} \end{aligned} \quad (3)$$

식 (1)은 점 S1 (x_1, y_1)과 점 T1 (x_1', y_1')을 동차 좌표계 $((x_1, y_1, 1), (x_1', y_1', w'))$로 변환하여 원근 변환 행렬 M을 구하는 과정이다. 동차 좌표계로 표현하면 변환 행렬을 통해 원근 변환을 쉽게 표현할 수 있다. 원근 변환 행렬 M은 식 (2)과 같이 3×3 행렬로 표현되고, 동차 좌표를 다시 2D 좌표로 변환하면 식 (3)으로 표현된다. A 영역의 점 4 쌍 (S−T)에 대하여 위의 과정을 반복하면 식 (3)과 같은 방정식이 총 8개가 나와 원근 변환 행렬, M_A를 구할 수 있고, 이와 동일한 방법으로 B 영역의 변환 행렬 M_B를 구할 수 있다. 이를 통해 원본 영상 속 모든 좌표 (S1~S8)은 행렬 M_A와 M_B를 활용하여 변환 영상의 좌표 (T1~T8)로 표현할 수 있고, 결과적으로 변환 영상 속 좌표를 통해 동적 객체의 위치를 추정한다.

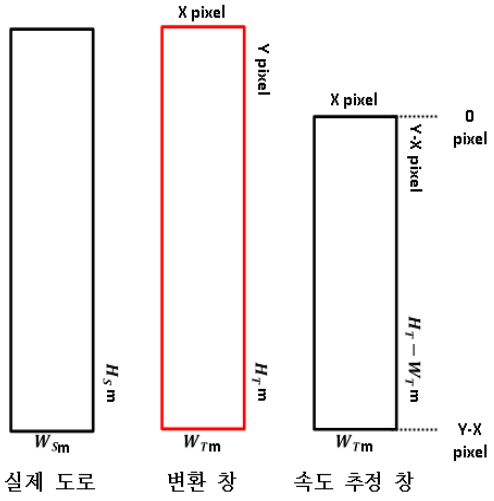
3.1.2 속도 추정 알고리즘

위에서 설명한 BEV 변환을 통해 얻은 동적 객체의 위치 (좌표) 값을 활용하여 속도를 추정할 수 있다. 기본적으로 물체의 속도를 구하기 위해서는 물체가 이동한 거리와 그 거리만큼 이동하는 데 걸린 시간이 필요하다.



(그림 4) S-T 그래프, P-F 그래프

물리의 역학에서는 속도, 가속도를 다루는 과정 중 거리-시간 그래프와 속도-시간 그래프를 활용하여 기본적인 동역학을 이해한다. 이 과정에서 거리-시간 그래프의 기울기는 물체의 속도를 나타낸다. 이 개념을 영상분석에 적용하면, (그림 4)에서 보여주는 바와 같이 거리는 화소(Pixel), 시간은 프레임(Frame)으로 고려할 때, 화소/프레임(Pixel/Frame) 단위로 영상 속 물체의 속도를 추정할 수 있다.



(그림 5) 실제 도로, 변환 창, 속도 추정 창 크기

(그림 5)는 차량의 위치 및 속도를 추정하기 위해 사용되는 세 개의 영역의 크기를 나타낸 것이다. 실제 도로 크기를 $W_S \times H_S$로 가정하면 변환 창에 보여지는 원본 영상 속 실험 도로 크기는 $W_T \times H_T$로 실험 도로와 실제 도로의 비율을 알 수 있다. 이 비율은 실제 도로의 속도를 추정할 때 사용할 수 있다.

속도를 추정하기 위해 변환 창이 아닌 속도 추정 창을 활용한다. 이는 거리가 먼 객체의 경계 사각형(BBox)이 튀거나 객체 인식이 명확하게 되지 않아 불안정한 값이 측정되는 것을 방지하기 위해서이다. 따라서 변환 창에서 $X-$화소 크기의 정사각형 영역을 잘라내어 속도 추정 창을 만든다. 속도 추정 창 내에서 객체 중심의 y 좌표를 통해 객체의 이동 거리 S를 측정한다. 이동 거리는 객체의 중심점이 속도 추정 창에 들어온 순간을 0 이라 할 때, 창에서 벗어나는 순간까지의 거리는 $Y-X$ 화소라 할 수 있다.

$$\frac{\text{객체 중심의 } y\text{좌표}(S)}{\text{처리한 프레임 수}(T)} = \text{속도}(V) \qquad (4)$$

이동 거리는 매 프레임 갱신되기 때문에 객체가 속도 추정 창에 들어온 순간부터 빈 리스트에 이동 거리 값을 추가한다. 따라서 이 리스트의 길이는 속도 추정이 시작되는 순간부터 끝나는 시점까지의 처리한 프레임 수로 해당 구간을 이동하는 데 걸린 시간 T를 구할 수 있다. 이를 통해 이동 거리인 객체 중심의 y 좌표 (S)에서 시간에 해당하는 처리한 프레임 수 (T)를 나눠주면 식 (4)에서 보여주는 바와 같이 속도 (V)를 추정할 수 있다.

$$V_{ppf} = \frac{S}{T} \qquad (5)$$

식 (5)에서 객체의 속도 V_{ppf}는 화소 단위의 객체 간 이동 거리 S와 이때 걸린 시간을 나타내는 프레임 수 T의 비율로 나타낼 수 있으며 속도의 단위는 프레임당 화소 수 (p/f)로 나타낼 수 있다. 이 값으로부터 차량의 속도 실제 속도를 추정하기 위해서는 km/h 단위로 변환할 필요가 있다.

$$V_{mps} = V_{ppf} \times \frac{H_T - W_T}{Y - X} \times R \qquad (6)$$

식 (6)은 프레임 당 이동 화소 수로 구한 속도인 V_{ppf}를 초당 미터 단위의 이동 거리 비율로 속도를 변환하는 과정이다. 이때 (그림 5)에서 보여주는 바와 같이 속도 추정 창에 대한 실제 실험 도로 길이 $(H_T - W_T)$를 창의 높이 $(Y - X)$로 나눠주어 m/f 단위의 값을 구하고 이를 프레임 레이트(frame rate)에 해당하는 값 R을 곱해서 m/s 단위의 속도인 V_{mps}를 구한다.

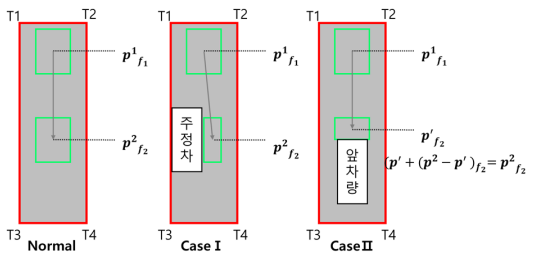
$$V_{kmph} = V_{mps} \times \frac{3600}{1000} \quad (7)$$

다음에는 식 (7)을 이용하여 초당 미터 단위의 이동 거리로 구한 속도인 V_{mps}를 일반적인 차량의 이동 속도 표기 단위인 시간당 킬로미터 단위의 이동 거리 비율로 속도를 변환한다.

$$\hat{V}_{kmph} = V_{kmph} \times \frac{W_S}{W_T} \quad (8)$$

다음에는 식 (8)에서와 같이 실제 도로 길이와 실험 도로 길이의 비율 (W_S / W_T)을 곱하여 실제 도로 환경에서의 속도, $\hat{V}_{kmph}$를 추정한다.

위의 수식들을 활용하여 속도를 추정하기 위해서는 객체를 정확히 인식해서 경계 사각형이 올바르게 검출되어야 한다. 그러나 심한 혼재와 가림으로 인해 경계 사각형이 뒤틀리는 오류가 발생할 수 있다. 따라서 이 오류를 억제하기 위한 방안을 (그림 6)을 활용하여 제시한다.



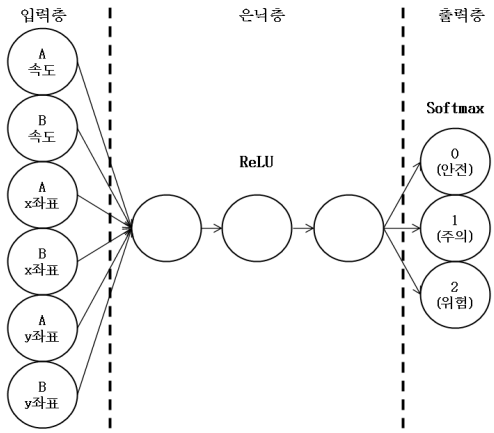
(그림 6) 3가지 경우의 경계 사각형 검출

객체 중심의 y좌표를 p, 그때의 처리한 프레임 수를 f라 하고, 그 순간의 속도를 p_f라 가정한다. 경계 사각형이 정상적으로 검출된다면 Normal과 같이 경계 사각형 모양의 변화가 크게 없어야 한다. 이때 속도는 $p^{1}{}_{f_1}$, $p^{2}{}_{f_2}$로 올바르게 추정되고, 차량이 등속 운동을 한다면 $p^{1}{}_{f_1}$과 $p^{2}{}_{f_2}$의 값은 동일해야 한다. 만약, Case Ⅰ과 같이 주정차들로 인해 속도를 추정하는 객체가 가려진다면 경계 사각형이 세로로 긴 직사각형 형태로 변하게 된다. 이 경우에서는 객체 중심의 x 좌표만 영향을 받기 때문에 객체 중심의 y 좌표의 변화만을 활용하여 속도 추정을 진행한다면 문제가 되지 않는다. 그러나 Case Ⅱ와 같이 앞 차량 혹은 도로 주변 장애물로 인해 객체가 가려져 경계 사각형이 가로로 긴 직사각형 형태를 가지게 된다면 Case Ⅰ의 방법으로 해결할 수 없다. 따라서 경계 사각형의 가로, 세로 비율을 계산하여 비율의 변동 값에 따라 p를 보정한다.

Normal에서 경계 사각형의 세로 비를 8로 고정하고, 경계 사각형의 비가 8:6이라 가정한다. 이때, Case Ⅰ의 비는 8:3, Case Ⅱ의 비는 8:24로 해석할 수 있다. 즉, Normal과 Case Ⅰ에서는 세로 비가 크지만, Case Ⅱ에서는 가로 비가 더 커지게 된다. 경계 사각형의 세로 비가 감소한 만큼 객체 중심의 y 좌표는 '감소한 경계 사각형 세로 길이/2'만큼 작아지게 된다. 작아진 크기만큼 p'값에 더해주어 p^2값과 같아지게 만들어 속도를 추정한다. 이러한 방안을 통해 경계 사각형 검출 오류에 의한 속도 추정 오류 억제를 진행한다.

3.1.3 MLP 활용 위험 상황 인지

위에서 구한 동적 객체의 위치와 속도를 MLP 분류기로 세 가지 위험 상황 중 하나로 분류하여 위험 상황을 인지 한다.



(그림 7) MLP 분류기 모델 구조

위치와 속도를 MLP의 입력값으로 넣어주기 위해서는 정적 데이터 형식을 불러와야 한다. 따라서 객체가 교차로에 진입할 때 객체의 위치와 속도를 CSV 파일 형식으로 저장하고, 저장된 CSV 파일을 불러와 MLP 모델을 학습한다. 본 논문에서 제안하는 MLP 구조는 (그림 7)에서 보여주는 바와 같이 1개의 입력층, 3개의 은닉층, 1개의 출력층을 가지고, 입력층 노드 6개, 출력층 노드 3개로 고정한다.

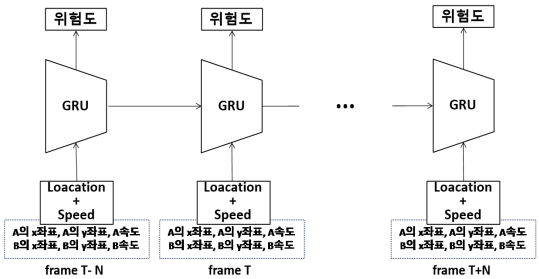
입력층 노드는 A의 속도와 위치, B의 속도와 위치를 넣고, 위치는 x와 y로 나누어 총 6개의 노드를 가진다. 은닉층의 노드 수는 학습을 통해 찾아내고, 활성화 함수는 ReLU를 활용한다. 출력층의 클래스는 3가지(안전, 주의, 위험) 클래스로 다중 분류 문제이기 때문에 출력층의 활성화 함수는 Softmax를 사용한다.

최적의 모델을 빠르게 찾기 위해 하이퍼파라미터 최적화 탐색 기법인 Grid Search [31]를 사용한다. 하이퍼파라미터인 alpha, hidden_layers_sizes, learning_rate_init을 다양하게 설정하여 MLP 분류기 모델을 학습시키고 최고의 성능이 나온 모델의 하이퍼 파라미터를 고정하고, 이 모델에 대한 성능을 평가한다.

3.1.4 GRU 활용 위험 상황 인지

GRU는 교차로에 진입한 객체의 위치와 속도를 프레임 단위로 CSV 파일 형식으로 저장하고, 이 데이터를 통해서 학습한다. 이때 영상의 프레임 길이는 차량의 속도나 상황에 따라서 다를 수 있어 최대 길이의 영상 프레임을 측정하고, 이에 맞게 Padding 기법을 사용하여 모든 데이터의 길이를 통일한다. 또한 이때 Padding으로 생성된 데이터 길이는 학습에 사용하지 않기 위해서 Masking 기법을 사용한다. 학습 하이퍼 파라미터 튜닝을 위해서 optuna 패키지를 사용해서 최적의 모델을 학습한다.

본 절에서는 3.1.1과 3.1.2에서 YOLO 모델을 통해 추출된 위치와 속도를 시계열 데이터 형식으로 CSV 파일에 저장한 뒤 학습을 진행했다. 모델의 구조는 (그림 8)에서 파악할 수 있다.



(그림 8) GRU 분류기 모델 구조

GRU의 입력값으로는 A의 x좌표, A의 y좌표, B의 x좌표, B의 y좌표, A의 속도, B의 속도 총 6개의 데이터를 사용한다. 이때 GRU 모델 특성상 이전 Layer에서 발생한 결과값은 다음 Layer의 입력값으로 들어가 학습의 영향을 준다. 모델의 출력값으로는 '안전', '주의', '위험'으로 위험도가 나온다.

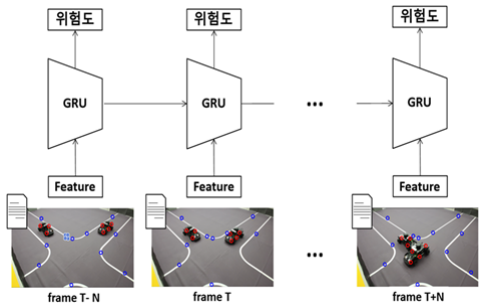
3.2 비디오 분류 기반 위험 상황 인지 모델

앞에서 언급한 교통사고 과실 판단 네트워크인 VTN(Video Transformer Network) 모델은 제한된 축소 환경인 점을 고려하여 모델을 간소화할 필요가 있다. 특징 추출 부분에서는 ImageNet으로 사전 학습된 Inception-v3의 가중치를 사용하고, 분류 부분에서는 비디오 흐름 파악을 위한 시계열 분석 모델 GRU를 사용한다. 정리하자면 Inception-v3 모델을 통해 특징을 시계열 데이터로 추출한 뒤, 영

상이 재생되면서 변하는 특징의 흐름을 GRU 모델로 학습시킨다고 볼 수 있다. 이를 통해 움직이지 않는 정적인 영상 환경에서 동적인 객체의 특징을 효과적으로 추출할 수 있다.

비디오 분류 기반 위험 상황 인지 판단 모델은 영상 데이터 자체를 입력값으로 위험 상황을 3가지로 분류한다. 여기에 사용되는 모델의 구조는 (그림 9)와 같다. 특징을 추출하는 모델로는 Inception-v3, 특징값을 통해 분류하는 모델로는 GRU를 통해서 위험 상황을 분류한다.

차량이 영상에 들어와서 차량이 영상에서 벗어날 때까지 특징 추출 모델의 입력을 받고, 분류를 위해서 일정 시간 간격으로 쌓인 입력값을 기준으로 GRU 모델의 추론을 진행하게 된다. 쌓인 입력값이 너무 적으면 원활한 추론이 되지 않기 때문에 본 연구에서는 시간 간격을 1초로 두고 진행한다.



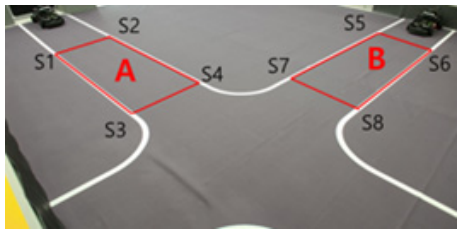
(그림 9) Video Classification 모델

4. 실험 및 성능 평가

실제 도로 환경에서 차량의 사고 영상 데이터를 찾거나 생성하기에는 어려움이 있으므로 실제 도로 환경과 유사한 실험 도로 환경을 만들어 실험을 진행한다. 실험 환경에서의 차량은 자율 주행이 가능한 자동차 로봇을 활용한다. 또한 실제 도로 환경에서 수집한 영상을 대상으로 실제 환경에서의 적용 가능성을 검증한다. 실험 도로 환경에서 영상 데이터를 수집하고 수집된 데이터에서 위험 상황을 인지한다. 위험 상황 인지 방법으로는 3장에서 말한 두 가지 접근 방식을 활용한다. 두 가지 접근 방법에서 사용되는 세 가지 모델 각각의 최적의 모델을 찾고, 성능 지표를 활용하여 각 모델의 성능을 평가한다.

4.1 실험 환경

실험 환경의 도로는 평균적인 골목길의 도로 폭과 불법 주정차를 가정하여 도로 폭 기준 (실제 도로:실험 도로=4m:0.4m) 약 10배 축소된 사지(四枝) 교차로로 제작하여 사용한다.



(그림 10) 실험 도로 환경

카메라는 1280×720(30fps)의 해상도로 영상을 촬영한다. 카메라는 (그림 10)에서 보여주는 바와 같이 교차로에 설치하여 2개의 도로 (왼쪽, 오른쪽)에서 오는 차량을 볼 수 있도록 고정한다. 본 논문의 실험에 사용한 컴퓨팅 환경으로는 Cuda 11.8 버전과 파이토치(PyTorch) 2.2.1, 그리고 텐서플로우(Tensorflow) 2.10.0을 사용하고 있다.

실제 차량을 대신할 장비는 (그림 11)에서 보여주고 있는 자율 주행이 가능한 자동차 로봇 'Q-Car'를 사용한다. 해당 장비는 NVidia Jetson TX2 보드를 이용한 차선 인식 주행이 가능하며, 깊이(Depth) 카메라를 활용한 객체 인식 및 전방 추돌 방지를 구현할 수 있다.

(그림 11) Q-Car

4.2 실험 데이터셋 구축

본 논문에서 제안하는 모델의 성능을 검증하기 위한 실험 데이터셋은 실험 환경에 대한 동영상 데이터셋을 가장 기본으로 하고 있다. 동영상 데이터셋을 활용해서는 차로별 차량의 객체 추적 및 이동 속도 추정에 활용한다. 추가로 차로별 차량의 속도와 위험 상황에 대한 데이터셋도 사용한다. 이 데이터셋은 차로별 차량 속도 기반 위험 상황 인지에 사용한다.

4.2.1 실험 동영상 데이터셋

교차로 동영상 데이터 셋은 실험 도로 위에서 차량의 움직임을 카메라로 촬영한 영상(비디오) 데이터로 구축한다. 데이터의 편향(Bias)을 방지하기 위해 차량의 위치, 출발 지점, 속도와 움직임을 <표 1>에서와 같이 다양하게 나누어 영상(비디오) 데이터를 수집한다. 이를 통해 최소한의 데이터로 제안하는 구조의 일반화 결과를 얻도록 한다.

〈표 1〉 차량의 속도, 위치, 출발 시점 분류

영역	차량 속도	차량 위치	출발 시점
A / B	S / M / F	L / C / R	A(t), B(t)

(그림 10)에서 보여주는 바와 같이 영역 A (왼쪽)와 B (오른쪽)는 교차로의 각 차로를 구분하여 나눈 것이다. 모델의 학습에 사용하는 데이터의 다양성을 확보하기 위해서 <표 1>에서 보여주는 바와 같이 차량의 속도, 출발 위치, 그리고 출발 시점 등 3가지 파라미터를 균일하게 분포하도록 조절하여 영상 데이터를 수집하고 실험에 사용한다.

먼저, 차량의 속도는 느림 (S), 보통 (M), 그리고 빠름 (F)의 3가지 종류로 구성한다. 실험에 사용하는 차량은 -0.2~0.2 범위 내에서 소수점 세 자리까지 값으로 설정하여 속도를 조절할 수 있다. 본 논문에서는 실험 차량의 속도 값을 0.05 (S), 0.075 (M), 그리고 0.1 (F)로 설정하여 실험 영상을 수집한다. 최고 속도로 설정한 0.1 (F)은 30 fps(frames per second)의 프레임 레이트(frame rate)로 영상을 입력할 때 객체가 안정적으로 검출될 수 있는 최댓값으로 제한하고 있다. 또한 차량을 제어하기 위한 속도 값을 고정값으로 주기 때문에 등속 직선 운동을 한다고 가정한다.

다음으로 차량의 출발 위치는 차량이 출발할 때 도로의 어느 쪽에 있는지에 따라 왼쪽 (L), 중간 (C), 오른쪽 (R)으로 구성한다. 이를 통해 차량이 도로를 주행할 때 나타날 수 있는 다양한 상황을 고려할 수 있다.

마지막 3번째는 각 차로를 진입하는 차량이 항상 동시에 진입하지는 않으므로 A 차로와 B 차로로 진입하는 차량의 출발 시점을 다양하게 설정한다. A 영역 차량의 출발 시점을 $A(t)$, 그리고 B 영역 차량의 출발 시점을 $B(t)$라고 가정할 수 있다. 처음에는 동일한 시점인 t 초에서 출발하고 이후부터는 A, B 두 영역에 진입하는 차량 중 하나의 출발 시점 t를 조절한다. 실험에 사용한 데이터셋에서 t의 범위는 0에서 3으로 0.5초 단위로 조절하여 영상(비디오) 데이터를 수집하고 실험용 영상 데이터를 구축한다.

본 논문에서 최종적으로 얻고자 하는 교차로의 상황은 사고 위험도에 따른 “안전”, “주의”, 그리고 “위험”의 3가지 상황이다. 따라서 본 논문의 실험을 위한 영상 데이터의 수집도 이와 같은 3가지 상황을 고려하여 수행한다. 또한 데이터 불균형의 문제를 방지하기 위해 3가지 상황을 각각 72개씩 모두 216개의 동영상을 수집하여 실험 영상 데이터셋을 구축하고 있다.

앞에서 논의한 바와 같이 차로별 차량의 속도, 위치, 출발 시점과 같은 3가지 파라미터에 대해 균등하게 분포하도록 실험 데이터를 수집하고 구축하여 사용함으로써 구축한 데이터셋의 편향을 방지하고 일반화를 유지하도록 한다. 또한 인지하고자 하는 각 상황 별로 동일한 수의 데이터를 수집함으로써 불균형의 문제를 해결한다.

위험 상황 인지에 대한 두 가지 접근 방법 (위치 및 속도 기반, 비디오 분류 기반)에서 사용되는 교차로 위험 상황 인지 모델 학습에 사용할 동영상 데이터에 라벨링을 해주기 위해 분류 기준은 다음과 같다.

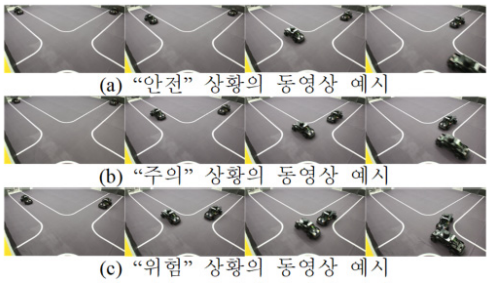
사고/비사고 영상(비디오) 데이터를 나누고 사고 데이터의 상황을 '위험 (2)'으로 분류한다.

비사고 영상에서는 학습한 객체 탐지 및 추적 YOLO v8 모델을 통해 A 영역의 객체 (차량)와 B 영역의 객체 (차량) BBox 중심 좌표(x, y)를 구한다.

math 라이브러리의 dist() 함수를 활용하여 A, B 영역 내 객체 BBox 중심 좌표 사이의 거리 (d)를 계산한다.

거리 (d)가 400 (실제 실험 도로상 거리 0.5m) 미만이면 '주의 (1)', 400 이상이면 '안전 (0)'으로 분류한다.

(그림 12)에서는 교차로 위험도에 따른 "안전", "주의", 그리고 "위험"의 3가지 상황에 대한 실험 동영상 셋의 예시를 보여주고 있다.



(그림 12) 교차로 상황에 따른 실험 동영상 예시

이상과 같이 수집한 실험 동영상 데이터로부터 3.2절의 비디오 분류 기반 위험 상황 인지 모델의 실험에 사용하는 데이터의 수는 <표 2>와 같이 구분하여 사용한다. 각 상황 별로 58개의 동영상을 훈련에 사용하고 검증과 테스트에 각각 7개씩을 사용한다.

〈표 2〉 비디오 분류 기반 위험 상황 인지 모델의 학습을 위한 실험 데이터셋 구성

Ground Truth	훈련 데이터 수	검증 데이터 수	테스트 데이터 수	합계
안전	58	7	7	72
주의	58	7	7	72
위험	58	7	7	72
합계	174	21	21	216

4.2.2 차량의 위치 및 속도 기반 교차로 상황 인지 실험 데이터셋

다음으로는 앞의 4.2.1 절에서 설명한 방법으로 수집한 동영상 데이터셋으로부터 교차로에 진입하는 차량의 위치와 속도를 추출하여 (그림 7)과 (그림 8)의 분류기 모델의 학습에 사용하는 실험 데이터 셋을 구축한다. 교차로에 대한 각 차로별 차량의 위치와 속도는 3.1.1절과 3.1.2절에서 설명한 방법으로 구하고 실험 데이터 셋은 CSV 파일 형태로 구성한다. 본 절에서는 구축한 CSV 형식의 실험 데이터셋의 구조만 살펴보고 4.3.1절과 4.3.2절에서 3장에서 논의한 방법으로 실험을 통해 데이터를 수집하는 과정 및 데이터셋의 정확도 및 성능에 대해 상세히 살펴보고자 한다.

<표 3>에서 실험에 사용한 CSV 파일의 예시를 보여주고 있다. 4.1.1절에서 구축한 각 동영상 파일에 대해 각 프레임별로 차로 A의 차량과 차로 B의 차량의 속도와 위치를 추출하여 모델의 입력 특징으로 사용한다. 각 동영상의 라벨에 해당하는 "안전", "주의", 그리고 "위험"을 그라운드 트루(ground true)로 사용한다.

〈표 3〉 차량 위치 및 속도 기반 교차로 상황 인지 실험 데이터 예시 (CSV 파일)

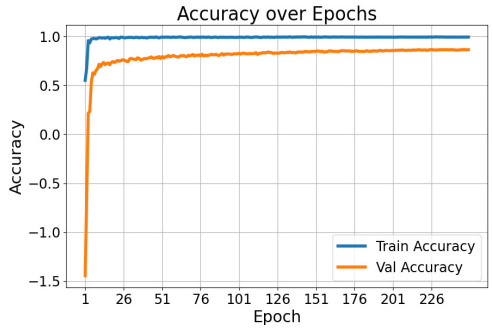
동영상	프레임	차로 A 차량			차로 B 차량			Ground Truth
		속도	x	y	속도	x	y	
1	1	0	0	0	13	58	675	안전 (0)
	2	0	0	0	14	58	497	
	...							
	55	13	100	693	12	53	271	
	56	13	100	542	13	59	89	
	...							
	104	14	100	348	00	0	0	
	105	14	100	154	0	0	0	
73	1	0	0	0	7	57	716	주의 (1)
	2	0	0	0	8	59	711	
	...							
	39	21	100	557	8	59	278	
	40	21	100	271	7	59	211	
	...							
	78	21	99	15	8	57	100	
	79	0	0	0	8	57	5	
145	1	0	0	0	14	57	489	위험 (2)
	2	0	0	0	13	60	385	
	...							
	26	21	100	648	12	58	294	
	27	21	101	399	12	58	124	
	...							
	51	21	100	155	0	0	0	
	52	21	101	12	0	0	0	

4.3 위치 및 속도 추정 기반 위험 상황 인지 모델

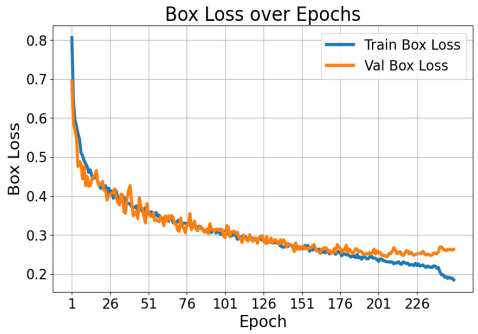
YOLO v8을 사용하여 객체 (차량)의 위치와 속도를 추정하고, 추정된 값을 특징값으로 활용하여 위치 및 속도 추정 기반 위험 상황 인지 모델을 두 가지 만든다.

4.3.1 YOLO 모델 학습과 영상 데이터 분류

YOLO v8을 사용하여 객체 (차량)를 탐지하고 추적하기 위해 4.2에서 수집한 영상 (비디오) 데이터에서 60프레임 단위로 영상 (이미지) 데이터를 추출한다. 추출된 영상 (이미지)에서 개방형 라벨링 도구인 'Labelimg'를 이용하여 영상 내에 존재하는 객체 (차량)에 대해 BBox(Bounding Box)를 입력하고 해당 BBox가 어떤 객체인지 라벨(label)을 붙여준다.



(a) Epoch에 따른 Accuracy 변화



(b) Epoch에 따른 Loss 변화

(그림 13) YOLO 기반 객체 탐지 결과 Accuracy, Loss 그래프

추출된 영상(이미지) 데이터는 1,364장이고, 2,728개의 객체(차량)를 라벨링 하였다. 영상에서 움직이는 차량을 인식하고 추적하기 위해 YOLO v8을 사용하여 객체 인식 및 추적 모델을 학습한다.

모델을 학습시키기 위해 1,364장의 영상(이미지) 데이터를 훈련 데이터(Train Data)와 테스트 데이터(Test Data)의 두 가지로 구분하고 각각 9:1로 나눈다. 학습 결과의 Accuracy와 Loss에 대한 그래프는 (그림 13)과 같다. 이 학습에서 객체 위치 및 속도 추정에서 활용되는 객체 특징값을 명확하게 하기 위해 BBox의 IoU와 정밀도가 높은 수치를 가지도록 해야 한다.

<표 4>에서 mAP와 mAR 수치를 중점적으로 제시하고 있다. 제한된 실험 환경에서 수집된 데이터를 통해 학습했기 때문에 어느 정도의 과적합이 이루어졌다고 생각된다. 결과적으로 작은 오차로 객체 탐지가 되도록 하고, YOLO v8에서 지원하는 BoT-SORT 알고리즘을 활용하여 객체에 ID를 부여한 동적 객체 추적이 되도록 한다.

〈표 4〉 mAR, mAP 지표(Epoch 250)

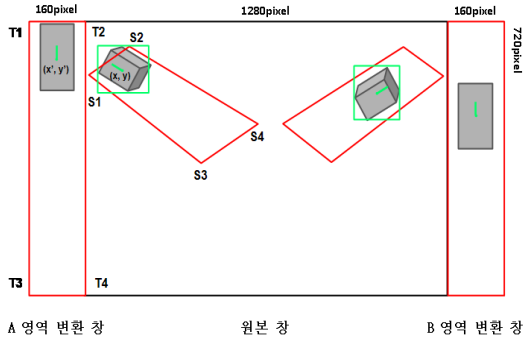
mAP(%)			mAR(%)		
50	75	95	1	10	100
0.98	0.94	0.94	0.93	0.98	0.98

4.3.2 YOLO를 활용한 위치 및 속도 추정

카메라(1280×720(30fps))를 통해 2차원 영상을 받아오고, 영상 속 동적 객체의 위치와 속도를 구하기 위해 YOLO 모델을 활용하여 2차원 영상 좌표계(1280×720) 상 객체의 BBox 중심 위치 (x, y)를 구한다. 이후 3.1에서 언급한 BEV 변환을 이용하여 동적 객체의 좌표 (x, y)를 변환 영상 좌표계 (160×720) 상 객체의 중심 좌표 (x', y')로 표현한다.

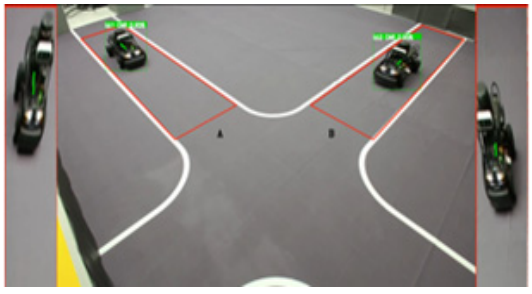
원본 영상 (1280×720)에서 교차로의 도로를 A (왼쪽)와 B (오른쪽)로 나누어 사각형의 영역을 지정한다. 실제 실험 도로의 A, B 영역의 크기는 0.4m×1.8m이다. 이후 각 영역의 꼭짓점 좌표를

정의하고 변환 영상 창 (160×720)의 꼭짓점 좌표에 매핑한다. OpenCV의 함수를 이용하여 원근 변환 행렬 (M)을 구하고, 원본 영상의 좌표를 변환 영상의 좌표로 변환한다. (그림 14)는 실험 환경 속 BEV 변환 과정의 예시를 보여주고 있다. 그림의 내용은 BEV 변환 중 A 영역의 변환 과정을 설명한 것이다.



(그림 14) 실험 환경 속 BEV 변환

- 원본 창에서 변환하고자 하는 영역 4개의 점 (S1, S2, S3, S4)을 지정하고, 변환 창에 표현할 4개의 점 (T1, T2, T3, T4)을 지정한다.
- 지정한 점을 OpenCV의 getPerspectiveTransform 함수를 통해 원근 변환 행렬 (M)을 계산한다.
- perspectiveTransform 함수를 사용하여 원본 창 속 객체의 중심 좌표 (x, y)를 받아 행렬 (M)을 통해 변환 창 속 객체의 중심 좌표 (x', y')로 변환한다.



(그림 15) 원본 영상과 변환 영상

(그림 15)는 실험 환경 속 원본 영상과 BEV 변환이 이루어진 영상을 나타내며, 가운데 원본 영상에 테두리 선으로 표시된 영역이 A와 B 영역이고, 원본 영상 양옆에 붙어있는 영상이 A와 B 영역의 변환 영상이다.

지정한 동적 객체 이외의 객체가 탐지되고 추적되는 것을 방지하기 위해 지정한 동적 객체의 중심 좌표가 A와 B 영역의 내부에 있을 때 특징값이 추출되도록 했다. 여기서 지정된 동적 객체는 앞에서 말한 Q-Car이다. 지정된 동적 객체의 중심점이 영역 내부로 들어오면 원본 영상에 BBox와 중심점 좌표를 표현하고, 중심점 좌표는 원근 변환 행렬 (M)을 통해 변환 영상에도 표시한다. 표시된 변환 영상의 객체 중심점 좌표를 통해 지정된 동적 객체의 위치와 관련된 특징값들을 추출한다.

위치와 관련된 특징값으로는 'A 내부 객체의 x 좌표 (Ax)', 'B 내부 객체의 x 좌표 (Bx)', 'A 내부 객체의 y 좌표 (Ay)', 'B 내부 객체의 y 좌표 (By)'로 총 4개가 추출된다. 각 특징값의 범위는 아래의 <표 5>에서와 같다.

〈표 5〉 위치 특징값 범위

범위	특징값
0~160	A 내부 객체의 x 좌표(Ax) B 내부 객체의 x 좌표(Bx)
0~720	A 내부 객체의 y 좌표(Ay) B 내부 객체의 y 좌표(By)

3.1.2의 속도 추정 알고리즘을 활용하여 실험 환경 속에서 동적 객체의 속도를 추정한다. 실제 교차로 환경 속 A, B 영역의 실제 도로 크기를 4m×18m로 가정하면 변환 창에 나타나는 0.4m×1.8m 크기의 실험 도로는 실제 도로보다 10배 작은 크기이다. 그러면 속도 추정 창에 보이는 실험 도로 크기는 0.4m×1.4m (160×560)로 정해진다. 여기에 카메라의 성능인 30 fps를 3.1.2의 식 (5, 6, 7, 8)에 대입하면 아래와 같은 식 (9, 10, 11, 12)이 나온다.

$$V_{slope} = V_{ppf} = \frac{S}{T} \quad (9)$$

$$V_{mps} = V_{ppf} \times \frac{1.4}{560} \times 30 \quad (10)$$

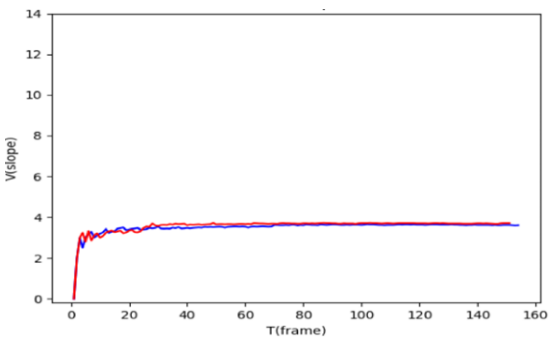
$$V_{kmph} = V_{mps} \times \frac{3600}{1000} \quad (11)$$

$$\hat{V}_{kmph} = V_{kmph} \times \frac{W_S}{W_T} = V_{kmph} \times 10 \quad (12)$$

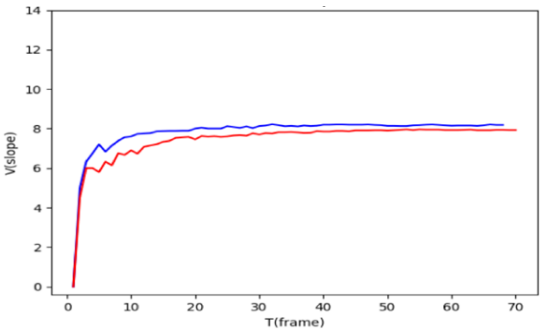
식 (9)에서 구한 V_{slope} 값은 4.2절의 영상 데이터 셋 구축에서 구성한 차량의 속도 (S, M, F)로 매칭하여 분류한다. 분류기 모델의 입력값으로 사용될 객체의 특징값으로는 V_{slope}를 활용한다. 이는 연산량을 줄여 처리 속도를 조금이라도 향상하기 위해서이다. 이를 통해 A 영역 내의 동적 객체의 속도 'A_Slope'와 B 영역 내의 동적 객체의 속도 'B_Slope'를 구하고, 분류기 모델의 입력값으로 사용한다.

〈표 6〉 V_{slope}에 따른 A, B 속 객체 속도 분류

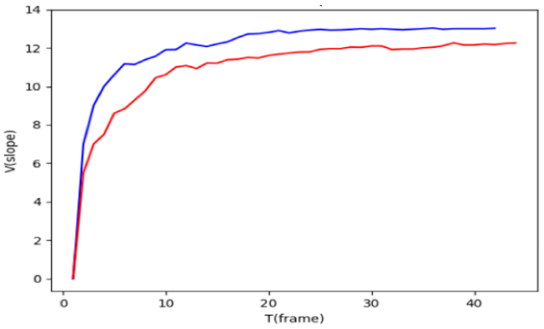
V_{slope}	V_{ppf}		V_{mps}		V_{kmph}		$\hat{V}_{kmph}$	
Area (영역)	A	B	A	B	A	B	A	B
S (느림)	3.60	3.70	0.27	0.28	0.97	1.01	9.70	10.1
M (중간)	8.19	7.93	0.61	0.59	2.20	2.12	22.0	21.2
F (빠름)	13.0	12.2	0.98	0.92	3.53	3.31	35.3	33.1



(a) 속도가 느린 경우



(b) 속도가 중간인 경우



(c) 속도가 빠른 경우

(그림 16) A, B 내 객체의 속도(V)-시간(T) 그래프

<표 6>은 A, B 내의 객체가 각각 영역을 나올 때 V_{slope} 값을 측정하고 그 값으로 실제 속도를 계산한 것이다. V_{slope}의 값은 S (느림)는 약 '4', M (중간)은 약 '8', F (빠름)는 약 '12'의 수치를 보여주고 있다.

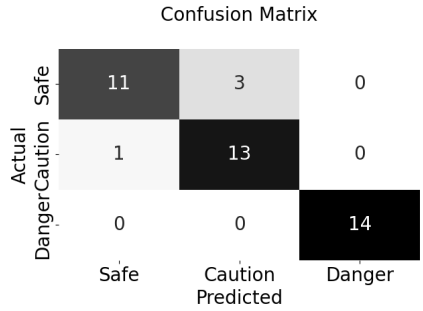
이 수치를 바탕으로 속도(V)−시간(T) 그래프를 그려보면 (그림 16)과 같이 그려진다. 그래프를 보면 기울기 값이 0에 수렴하여 차량이 등속 운동을 한다고 볼 수 있고, 차량의 속도에 따라 V_{slope} 값이 분류되는 것을 볼 수 있다.

이렇게 구한 2개의 속도 값 (A_slope, B_slope)과 앞에서 구한 4개의 위치 값 (Ax, Bx, Ay, By)을 추출하여 총 6개의 특징값을 활용한다. 추출된 특징값을 정적 데이터 형식과 시계열 데이터 형식으로 CSV 파일을 만들고, 이를 인공지능 분류기 모델에 넣어 위험 상황을 인지한다.

4.3.3 YOLO + MLP

앞에서 추정한 위치와 속도에 대한 6개의 특징(A_slope, B_slope, Ax, Ay, Bx, By)값을 이용하여 MLP를 활용한 위험 상황을 인지한다. MLP에 활용되는 정적 데이터 형식의 6개의 특징값은 객체가 A, B 각 영역에서 나올 때 (교차로에 진입할 때) 추출되어 CSV 파일에 저장된다. 4.2.1에서 수집한 216개의 영상 (비디오) 데이터에서 특징을 추출하고, 분류된 위험도에 따라 CSV 파일을 완성한다. 입력된 6개의 객체 특징값과 3개의 위험도를 입력값과 출력값으로 분리하고, 데이터를 '학습:검증:테스트=8:1:1'로 분류한다.

Python의 'sci-kit learn' 라이브러리를 활용하여 MLP 분류기 모델 학습을 진행한다. MLP 분류기 모델의 max_iter를 5,000으로 설정하여 초기화하고, Grid Search를 활용하여 최적의 하이퍼파라미터를 찾는다. 이로써 나온 최적의 모델 하이퍼파라미터는 alpha: 0.1, hidden_layer_sizes: (30, 50, 30), learning_rate_init: 0.001이다. 학습, 검증, 테스트 데이터의 Accuracy는 각각 0.95%, 0.86%, 0.89%로 구해지고, 이때의 epoch는 491이다. 이 중 0.89%의 Accuracy를 가진 테스트 데이터에 대한 혼동 행렬은 (그림 17)과 같다.

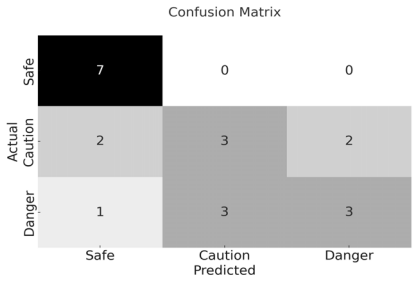


(그림 17) MLP 테스트 데이터 혼동 행렬

위험 영상 데이터에 대해선 정확하게 분류를 하고 있지만, 안전과 주의 영상 데이터에 대해서는 조금의 오분류가 발생하는 것을 알 수 있다.

4.3.4 YOLO + GRU

GRU 모델은 MLP와 달리 특징값을 시계열 데이터로 입력시킨다. GRU 모델은 프레임 별 출력값인 위험도가 출력되게 되는데, 이 위험도는 초당 30번의 변화량을 갖게 되어 너무 잦은 위험도의 변동을 불러온다. 따라서 프레임별로 측정한 위험도 값을 바탕으로 최빈값을 사용함으로써 위험도의 갑작스러운 변동을 무시하고 이상치에 대한 민감성을 감소시킬 수 있다.

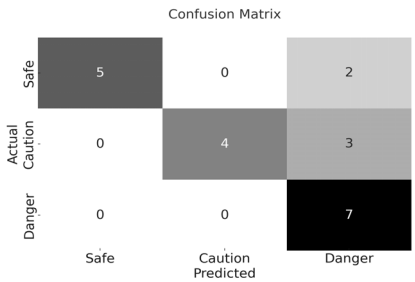


(그림 18) GRU 테스트 데이터 혼동 행렬

(그림 18)은 YOLO+GRU에 대한 테스트 데이터 혼동 행렬이다. 안전 영상 데이터는 완벽하게 분류하지만, 주의와 위험 영상 데이터는 제대로 분류하지 못하는 결과가 나온다. 위험을 주의로 분류하거나, 주의를 위험으로 분류하는 오분류가 발생한다.

4.4 비디오 분류 기반 위험 상황 인지 모델

위치 및 속도 추정 기반 모델의 비교군으로 둔 비디오 분류 기반 위험 상황 인지 모델의 테스트 데이터에 대한 혼동 행렬은 (그림 19)와 같다.



(그림 19) 비디오 분류 테스트 데이터 혼동 행렬

비디오 분류 모델의 영상 데이터는 시퀀스 길이가 가장 긴 차량 충돌 전을 기준으로 추출했다. 위험 영상 데이터에 대한 분류는 완벽하게 했지만, 안전과 주의 영상 데이터에 대한 오분류가 발생한다.

4.5 실험 결과 분석

본 논문에서는 교차로에서의 위험 상황 인지를 위해 YOLO와 MLP를 결합한 모델과 YOLO와 GRU를 결합한 모델, 그리고 Video Classification 모델 3가지를 비교 실험하고 있다. 실험 결과 분석을 위해 각 모델의 테스트 데이터에 대한 성능 지표를 <표 7>에서 보여주고 있다. YOLO+MLP는 YOLO+GRU에 비해 Accuracy가 약 35% 뛰어나며, Video Classification에 비해 추론 시간이 약 224배 빠른 결과를 보인다.

〈표 7〉 제안 모델과 비교 모델에 대한 성능 비교

	YOLO+MLP	YOLO+GRU	Video Classification
Accuracy	0.89	0.64	0.86
F1-score	0.88	0.62	0.86
Recall	0.88	0.59	0.86
Precision	0.89	0.62	0.86
추론시간	5 ms	10 ms	1120 ms

세 가지 모델에 대해 오분류가 발생한 원인으로 영상 데이터의 수가 다소 부족한 결과로 판단되며, 주의 영상 데이터에 대한 분류가 정확하게 이루어지지 않는 것으로 보아 분류 기준이 명확하지 않았을 거라 예상한다. 또한, GRU의 경우 긴 시퀀스 간의 관계를 효과적으로 파악하여 훈련하기 때문에 실험 영상 데이터로는 좋은 성능을 낼 수 없었다고 생각한다. 보다 다양한 환경에서 영상 데이터를 수집하고 모델의 성능 개선에 활용한다면 보다 우수한 결과를 얻을 수 있을 것으로 기대된다.

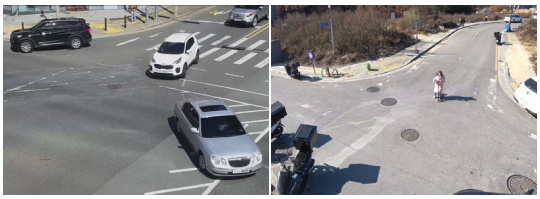
4.6 실세계 영상에 대한 제안 모델의 적용

4.6.1 AI 허브 영상에 대한 분석

제안하는 모델에서의 차량 속도 추정 알고리즘을 실세계 영상에 적용하기 위해서는 2가지 조건이 필요하다. 먼저 실험에 사용하고자 하는 도로의 실제 길이를 알아야 하고 다음으로는 모델에서 추정한 차량의 속도를 검증하기 위해 실제 차량의 속도를 알아야 한다. 실제 AI 허브에 공개된 데이터셋중에 (그림 20)과 같이 도로 교통 정보를 보여주고 있는 데이터가 있다. 일부 차량 속도 추정 태스크를 위한 데이터셋에서 차량의 속도를 그라운드 트루로 제공하고 있다. 하지만, 본 논문에서 필요로 하는 속도와 실제 도로 길이 정보를 모두 가지고 있는 데이터셋은 없는 것으로 확인된다.

본 논문에서는 제안하는 모델을 실제 도로 영상에 대해 적용해보기 위해 별도로 영상을 촬영하여 데이터셋을 확보하고 실험을 진행하고자 한다.

(a) 도심 혼잡 데이터 (b) 차량 이동 데이터



(c) 교통 정보 데이터 (d) 어린이 보호구역

(그림 20) AI 허브에 등록된 데이터 예시

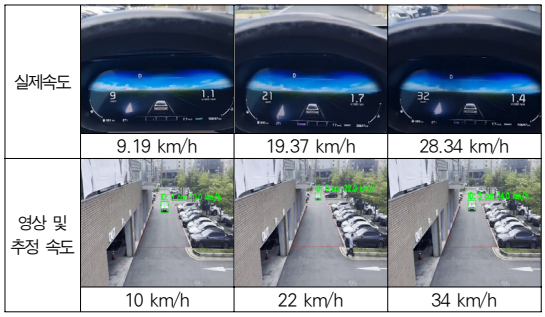
4.6.2 실세계 도로 영상 구축 및 실험 결과 분석

본 논문에서는 제안하는 모델을 실제 도로 환경에 적용하여 성능을 검증하고자 한다. 이를 위해 각각 서로 다른 4 곳의 장소에서 각각 10 km/h, 20 km/h, 그리고 30 km/h의 3 가지 속도를 기준 속도로 하여 영상을 수집한다. 또한 실제 도로의 길이를 50 미터, 40 미터, 30 미터, 그리고 20 미터로 다양하게 하여 제안하는 모델의 일반성을 검증한다.

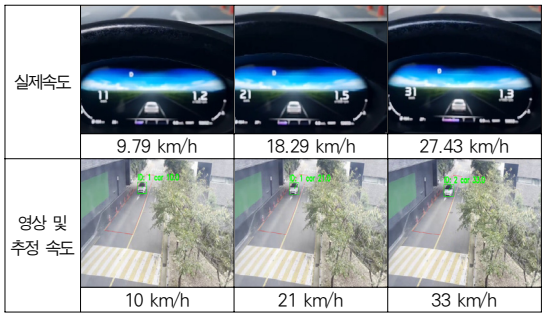
(그림 21)에서는 이와 같이 수집한 영상에 대해 제안하는 모델을 적용하여 차량의 객체 검출 및 추적을 통해 속도를 추정한 결과를 보여주고 있다. 각각의 영상에 대해 실제 속도는 앞에서 언급한 바와 같이 10 km/h, 20 km/h, 그리고 30 km/h를 기준으로 하고 있다. 하지만 사람이 운전하는 관계로 실험에 사용한 도로 구간에 대해 일정하게 맞추기 어렵다. 이러한 점을 고려하여 구간 내의 평균 속도를 제안하는 모델을 이용하는 추정 속도의 그라운드 트루로 사용한다.

기준 속도 10 km/h의 경우 실제 영상에서의 구간 평균 속도에 대한 속도 추정의 결과를 살펴보면, 실제 거리가 50 미터부터 30 미터까지 구간에 대해서는 오차가 1 km/h 이내이다. 하지만 기준 속도 30 km/h인 경우에는 실제 속도를 맞추기 어려워 평균 속도가 기준 속도와 차이가 많이 나며 추정 속도 역시 오차가 크다.

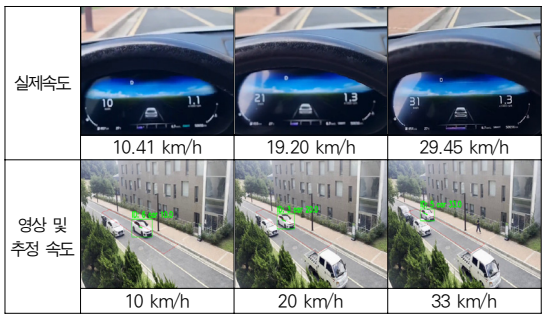
실제 도로의 구간 거리가 가장 짧은 20 미터인 경우를 살펴보면, 각각의 실제 평균 속도가 12.34 km/h, 22.15 km/h, 그리고 32.00 km/h인 경우에 대해 추정 속도가 11 km/h, 22 km/h, 그리고 31 km/h로 오차가 현저히 줄어든다. 전체적으로 오차가 가장 큰 경우를 살펴보면 구간 거리가 50 미터인 도로에서 실제 평균 속도가 28.34 km/h인 경우 추정 속도가 34 km/h로 오차가 5.66 km/h로 약 20% 정도이다. 본 논문에서 제안하는 모델은 차량의 속도를 기반으로 교차로의 위험 상황을 “안전”, “주의”, 그리고 “위험”의 3 가지로 구분한다. 따라서 속도 추정의 정확성이 보장된다면 이후 위험 상황의 판단은 3.1.3절과 3.1.4절에서 논의한 바와 같이 MLP 또는 GRU를 통해 이루어진다. 따라서 제안하는 모델이 실제 도로 환경에서도 우수한 성능을 보일 것으로 기대할 수 있다.



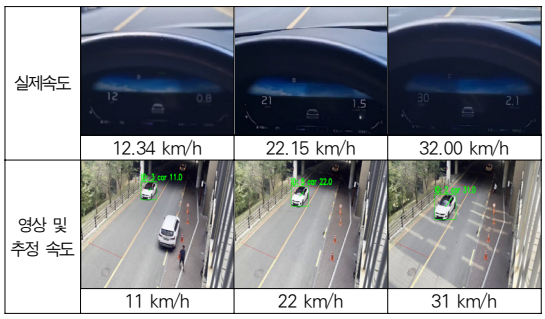
(a) 실제 거리가 50 미터인 도로



(b) 실제 거리가 40 미터인 도로



(c) 실제 거리가 30 미터인 도로



(d) 실제 거리가 20 미터인 도로

(그림 21) 실제 도로 영상에서 속도 추정 결과 예시

5. 결론 및 향후 연구

본 논문은 교차로에서 차량 교통사고 데이터셋을 제작하여 단일 카메라를 통한 위치 및 속도 추정 기반 위험 상황 인지 모델을 제안하고 있다. 제안한 방법은 기존 방법용, 교통정보용 CCTV로 제작된 기존 위험 상황 인지 모델과 달리 추가적인 센서 없이 단일 카메라를 이용하여 위험 상황을 인지한다. 이를 위해 먼저 교차로 상의 이동 객체를 검출하고 프레임 간 객체의 이동 거리를 산출하여 속도를 계산한다. 이동하는 차량의 속도와 위치를 바탕으로 교차로에서의 사고 위험 상황에 대한 인지를 수행했고 학습 데이터셋을 이용하여 성능을 검증하였다. 실험 결과에 의하면 YOLO와 MLP를 결합한 제안 모델이 YOLO와 GRU를 결합한 모델과 비디오 분류 방법에 비해 교차로에서의 위험 상황 인지에서 89%의 정확도를 보여 다른 두 모델의 64%와 86%에 비해 우수한 성능을 보인다. 제안하는 모델은 통합 지능형 교통 제어 시스템의 핵심 요소로 활용 가능하다.

이후 연구에서는 실험 결과를 토대로 영상 데이터를 추가적으로 확보하고, 위험도 분류 기준에 위치와 속도 이외의 특징을 활용하여 위험 상황 인지 모델을 만들고자 한다. 특히, NVidia의 AI City Challenge Dataset [32] 중 교통 안전과 관련된 데이터 등 관련 공개 데이터에 대해 제안하는 모델을 확장하여 적용하고자 한다. 또한, 실시간으로 교차로 내 위험 상황을 인지하고 판단하여 신호 및 차량을 제어할 수 있는 교차로 시스템을 구축하는 연구를 지속적으로 진행하고자 한다.

▌참고문헌

〔1〕 KoROAD, "Traffic Accident Statistics by Road Type", 2023.

〔2〕 N. Behboudia, S. Moosavib, and R. Ramnath, "Recent Advances in Traffic Accident Analysis and Prediction: A Comprehensive Review of Machine Learning Techniques", https://doi.org/10.48550/arXiv.2406.13968, 2024.

〔3〕 S. Ahmed, A. Hossain, S. Ray. M. Bhuiyan, and S. Sabuj, "A study on road accident prediction and contributing factors using explainable machine learning models: analysis and performance", Transportation Research Interdisciplinary Perspectives, vol. 19, May 2023.

〔4〕 A. Abdullayev, O. Allamov, O. Rustamov, R. Urinov, and Y. Kadamovna, "Analysis of existing smart crossroads and a new approach of smart crossroad", Proc. of International Conference on Information Science and Communications Technologies (ICISCT), Nov. 2021.

〔5〕 M. Kim, W. Kim, and G. Choi, "A Drowsiness Detection System Using Multiple Driver Behavior Features Based on SERN", Journal of Korean Institute of Next Generation Computing, vol. 20, no. 3, pp. 47-55, 2024.

〔6〕 A. Thaduri, V. Polepally and S. Vodithala, "Traffic Accident Prediction based on CNN Model," 2021 5th International Conference on Intelligent Computing and Control Systems (ICICCS), Madurai, India, 2021, pp. 1590-1594, doi: 10.1109/ICICCS51141.2021.9432224.

〔7〕 Z. Jin and B. Noh, "From Prediction to Prevention: Leveraging Deep Learning in Traffic Accident Prediction Systems" Electronics 12, no. 20: 4335, 2023.

〔8〕 A. Pourroostaei, X. Liang, K. Mengistu, R. So, X. Wei, B. He, and A. Cheshmehzangi, "Road Car Accident Prediction Using a Machine-Learning-Enabled Data Analysis. Sustainability", 15(7):5939. https://doi.org/10.3390/su15075939,2023.

〔9〕 A. Kotsi, E. Mitsakis, and D. Tzanis, "Overview

of C-ITS Deployment Projects in Europe and USA", https://arxiv.org/pdf/2010.07299, 2020.

〔10〕 S. Ding, K. Muthuswamy, E. Vulpen, and A, Sesai, " A Comparative Assessment of C-ITS Technologies", Project 1-066, La Trobe University, Australia, May 2024.

〔11〕 J. S. Ham, B. K. Kim, and J. Y. Moon, "Prediction of pedestrian crossing intention in child protection zones", in Proc. of Korean Information Science Society Annual Conference, 2022, pp. 305-307.

〔12〕 T. Suzuki, H. Kataoka, Y. Aoki, and Y. Satoh, "Anticipating Traffic Accidents with Adaptive Loss and Large-scale Incident DB", CVPR 2018.

〔13〕 N. Peri, M. Li, B. Wilson, Y. Wang, J. Hays, and D. Ramanan, "An Empirical Analysis of Range for 3D Object Detection", ICCV 2023.

〔14〕 J. W. Baek, J. G. Lee, Y. W. Choi, and G. T. Lim, "Real-time road hazard detection and information provision system based on edge cameras", Journal of Computing Practices of the Information Science Society, vol. 27, no. 8, pp. 394-399, 2021.

〔15〕 J. S. Ham, B. K. Kim, and J. Y. Moon, "Dataset and model for predicting single and multiple pedestrian crossing intentions from CCTV footage", 2023.

〔16〕 S. Lee and Y. G. Lee, "Split liability assessment in car accident using 3D convolutional neural network", Journal of Computational Design and Engineering, vol. 10, no. 4, pp. 1579-1601, 2023.

〔17〕 J, Fang, L. Li, J. Zhou, J. Xiao, H. Yu, C. Lv, J. Xue, and T. Chua, "Abductive Ego-View Accident Video Understanding for Safe Driving Perception", CVPR 2024.

〔18〕 Y. Hong, "A study on the system for AI service production", KIPS Transactions on Computer and Communication Systems, vol. 11, no. 10, pp. 323-332, 2022.

〔19〕 R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation", in Proc. IEEE Conference on Computer Vision and Pattern Recognition, 2013.

〔20〕 R. Girshick, "Fast R-CNN", arxiv.org/pdf/1504.08083, 2015.

〔21〕 S. Ren, K. He, R. Girshick and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks", arxiv.org/pdf/1512.02325.

〔22〕 W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C. Fu, and A. Berg, "SSD: Single Shot MultiBox Detector", Nature, vol. 323, no. 6088, pp. 533-536, 1986.

〔23〕 J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection", in Proc. IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 779-788.

〔24〕 X. Xia, C. Xu, and B. Nan, "Inception-v3 for flower classification", in Proc. 2017 2nd International Conference on Image, Vision and Computing (ICIVC), IEEE, 2017.

〔25〕 A. Canziani, A. Paszke, and E. Culurciello, "An analysis of deep neural network models for practical applications", arXiv preprint arXiv: 1605.07678, 2016.

〔26〕 D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors", Nature, vol. 323, no. 6088, pp. 533-536, 1986.

〔27〕 H. Salehinejad et al., "Recent advances in

recurrent neural networks", arXiv preprint arXiv:1801.01078, 2017.
[28] R. Dey and F. M. Salem, "Gate-variants of gated recurrent unit (GRU) neural networks", in Proc. 2017 IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS), IEEE, 2017.
[29] Y. Yu et al., "A review of recurrent neural networks: LSTM cells and network architectures", Neural Computation, vol. 31, no. 7, pp. 1235-1270, 2019.
[30] "Image transformations", https://docs.opencv.org/4.x/da/d54/group__imgproc__transform.html.
[31] "GridSearchCV documentation", https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html.
[32] S. Wang, D. Anastasiu, Z. Tang, M. Chang, Y. Yao, L. Zheng, M. Rahman, M. Arya, A. Sharma, P. Chakraborty, S. Prajapati, Q. Kong, N. Kobori, M. Gochoo, M. Otgonbold, G. Batnasan, F. Alnajjar, P. Chen, J. Hsieh, X. Wu, S. Pusegaonkar, Y. Wang, S. Biswas, R. Chellappa, The 8th AI City Challenge, The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, June, 2024.

저자소개

◆ 김승현

- 2019년 3월~현재 대전대학교 정보통신공학과 학사과정
- 2023년 3월~현재 DSC공유대학 차세대통신융합전공 학사과정
- 관심분야: 컴퓨터비전, 인공지능, 딥러닝, 모빌리티 시스템, 차세대통신융합 등

◆ 강한빈

- 2019년 3월~현재 대전대학교 정보통신공학과 학사과정
- 2023년 3~현재 DSC공유대학 차세대통신융합전공 학사과정
- 관심분야: 컴퓨터비전, 인공지능, 로보틱스, 모빌리티 시스템, 차세대통신융합 등

◆ 강종규

- 2019년 3월~현재 대전대학교 정보통신공학과 학사과정
- 2023년 3월~현재 DSC공유대학 차세대통신융합전공 학사과정
- 관심분야: 인공지능, 모빌리티 시스템, 차세대통신융합 등

◆ 배창석

- 2003년 8월 연세대학교 전기전자공학과 공학박사
- 1989년 3월~2015년 4월 한국전자통신연구원 책임연구원(SW콘텐츠연구소 빅데이터SW연구부 휴먼컴퓨팅연구실장)
- 2004년 7월~2005년 8월 호주 시드니대학교 정보통신대학 방문연구원
- 2015년 5월~현재 대전대학교 IT융합공학부 정보통신공학과 교수
- 관심분야: 인공지능, 시각지능, 라이프로그, 빅데이터, 널 컴퓨팅, 차세대 컴퓨팅 등