디지털산업정보학회 논문지 제18권 제4호-2022년 12월
http://dx.doi.org/10.17662/ksdim.2022.18.4.031

노상 주차 차량 탐지를 위한 YOLOv4 그리드 셀 조정 알고리즘

김 진 호*

YOLOv4 Grid Cell Shift Algorithm for Detecting the Vehicle at Parking Lot

Kim Jinho

〈Abstract〉

YOLOv4 can be used for detecting parking vehicles in order to check a vehicle in out-door parking space. YOLOv4 has 9 anchor boxes in each of 13x13 grid cells for detecting a bounding box of object. Because anchor boxes are allocated based on each cell, there can be existed small observational error for detecting real objects due to the distance between neighboring cells. In this paper, we proposed YOLOv4 grid cell shift algorithm for improving the out-door parking vehicle detection accuracy. In order to get more chance for trying to object detection by reducing the errors between anchor boxes and real objects, grid cells over image can be shifted to vertical, horizontal or diagonal directions after YOLOv4 basic detection process.

The experimental results show that a combined algorithm of a custom trained YOLOv4 and a cell shift algorithm has 96.6% detection accuracy compare to 94.6% of a custom trained YOLOv4 only for out door parking vehicle images.

Key Words : YOLOv4, YOLO Custom Training, Vehicle Detection, Cell Shift Algorithm,

Ⅰ. 서론

야외 노상 주차 차량들을 자동으로 탐지하고 번호판을 인식하는 방식으로 주차 요금을 부과하기 위한 연구들[1, 2]이 많이 진행되고 있다. 최근 다양한 객체들을 실시간으로 탐지할 수 있는 YOLOv4 오픈 소스 라이브러리[3]가 차량 객체 탐지에 많이 활용되고 있다. 차량 객체 탐지를 위한 전용 영상 데이터베이스를 대상으로 YOLOv4를 학습시킨 다음, 학습 결과 구성(configure) 파일과 가중치(weights) 파일들을 이용해서 사용자 특화(customizing)된 실시간 차량 객체 탐지 애플리케이션을 구현할 수 있다.

YOLOv4 알고리즘을 이용하여 도로 주행 차량을 탐지하고 추적하기 위해 연속적으로 입력되는 영상 프레임에서 차량 객체를 탐지하는 경우[4] 한 두 프레임에서 객체 탐지에 실패해도 다음 프레임에서의 객체 위치를 추정하여 객체 탐지를 이어갈 수 있다는 특징이 있다. 노상 주차 차량의 경우 도로상의 주행 차량과 다르게 차량 객체의 위치 및 형태의 변화가 거의 없기 때문에 초기 차량 탐지에 실패한 경우 이어지는 차량 탐지 시도에서도 이전 영상 프레임과 동일한 형태로 차량 영상 프레임이 입력되어 YOLOv4 알고리즘을 반복적으로 적용하면 차량 객체 탐지에

* 경일대학교 전자공학과 교수(단독저자)

연속적으로 실패할 확률이 높아질 수 있다.

YOLOv4 오픈 소스 라이브러리의 객체 탐지 속도 및 성능을 제고하기 위한 연구들[5-7]이 많이 진행되고 있다. 노상 주차된 차량 객체를 탐지하고 차량 번호판을 인식하는 애플리케이션에서는 차량 객체의 탐지 속도 보다 탐지 성능 개선이 더 중요한 요소로 요구된다. YOLOv4의 객체 탐지 성능을 제고하기 위한 대부분의 연구들은 알고리즘 자체의 성능을 높이기 위한 딥러닝 네트워크 구성 및 학습 방법에 대한 연구[8]에 집중하고 있다. 이러한 성능 개선 노력에도 불구하고 입력 영상 프레임에 대해 객체 탐지에 실패할 경우 이어지는 연속 영상 프레임에서도 동일한 위치 및 형태의 차량 객체가 입력될 경우 YOLOv4의 객체 탐지용 그리드 셀 및 Anchor 박스의 위치 및 크기가 고정되어 YOLOv4의 객체 탐지 방식이 이전과 같기 때문에 연속적으로 차량 객체의 탐지에 실패할 수 있다. YOLO[4]에서는 비슷한 형태의 영상이 연속적으로 입력되면 동일한 방식으로 객체 탐지를 반복하게 되어 한 번 탐지에 실패할 경우 연속적으로 객체 탐지에 실패할 수 있다.

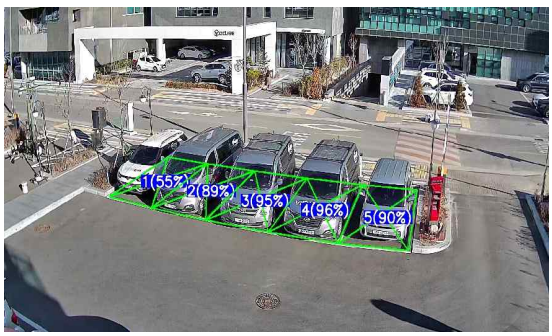
YOLOv4의 객체 탐지 그리드 셀의 위치를 조정(shift)하면 Anchor 박스의 위치도 이동되어 형태의 변화가 거의 없는 객체에 대해서도 다른 방식으로 객체의 종류 및 위치를 추정하는 효과를 얻을 수 있어서 추가로 객체 탐지를 시도하는 것과 같은 효과를 낼 수 있다.

본 논문에서는 야외 노상 주차 차량의 객체 탐지 성능을 제고하기 위해 YOLOv4의 그리드 셀 위치를 이동시켜 고정된 셀들 사이에 발생할 수 있는 탐지 오차 변위를 줄이면서 추가로 탐지 시도 회수를 늘일 수 있는 알고리즘을 제안하였다. YOLOv4에서는 입력 영상을 416x416 크기로 정규화한 다음 가로 및 세로 13x13개의 셀 영역을 구획하고 각 셀 마다 3개의 스케일에 대해 각각 3개의 Anchor 박스를 배정하여 객체의 Bounding 박스를 추론한다. Anchor 박스는 각 셀의 좌표를 기준으로 할당되어 있기 때문에 Anchor 박스의 위치는 셀의 위치에 종속적이다. 따라서 인접한 셀과 셀 사이의 거리에 따라 Bounding 박스 추론 과정에 오차가 발생할 수 있다. 본 연구에서는 셀 중심을 일정 방향과 범위로 이동 시켜 객체 추정 영역을 확장시키는 방식으로 탐지 시도 횟수를 늘여서 객체 탐지 성능을 개선할 수 있도록 하였다.

YOLOv4 라이브러리를 이용하여 노상 주차 차량 데이터베이스를 학습시키고 제안한 그리드 셀 조정 알고리즘을 YOLOv4 기본 알고리즘에 추가로 구현한 다음 야외 노상 주차 차량 객체 탐지 실험을 수행하고 성능을 분석하였다.

II. YOLOv4를 이용한 주차 차량 탐지 알고리즘 개요

노상 주차장에 주차된 차량들을 탐지하여 주차 면의 주차 여부와 함께 점유율을 추정한 예를 <그림 1>에 도시하였다.



<그림 1> 야외 주차 차량 탐지 및 주차면 점유율 추정 예

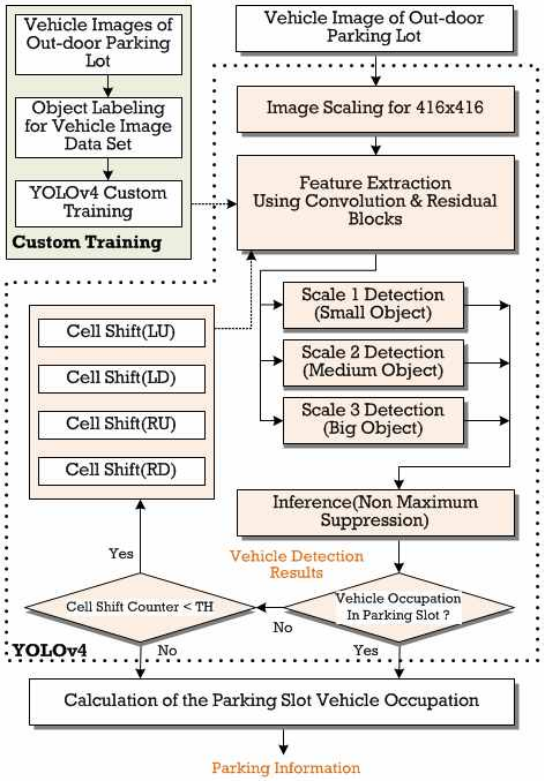
차량 객체 탐지를 위해 먼저 노상 주차 차량을 CCTV 카메라로 촬영하여 영상 데이터베이스를 구축하고 차량 객체의 위치와 크기 정보를 추출하기 위한 라벨링을 수행한다. 라벨링한 영상 데이터베이스를

대상으로 YOLOv4를 이용하여 객체 탐지를 위한 학습을 수행하고 가중치 파일을 생성한 다음 이를 이용하여 차량 객체 탐지 애플리케이션을 구현하고 주차면의 차량 탐지를 시도한다.

YOLOv4에서는 객체 탐지를 위해 영상을 가로 및 세로 13x13개의 그리드 셀로 구분하고 각 셀마다 Anchor 박스를 배정하여 각 Anchor 박스별로 객체의 Bounding 박스를 추정할 수 있도록 하였다. Anchor 박스들은 각 그리드 셀을 기준으로 할당되기 때문에 인접한 셀과 셀 사이가 최적 위치인 객체를 탐지해야하는 경우 배정된 Anchor 박스의 추론 과정에 오차가 발생할 수 있다. 특히 고정된 객체를 탐지하지 못했을 경우 YOLOv4의 그리드 셀의 중심 위치를 이동 시키면 추가로 객체 탐지를 다시 시도할 수 있기 때문에 탐지 정확도를 개선시킬 수 있다. YOLOv4를 이용한 주차 차량 탐지 과정에서 셀의 중심 위치를 이동시킴으로서 객체 탐지 정확도를 개선시키는 알고리즘을 <그림 2>에 도시하였다.

YOLOv4 라이브러리에 객체 탐지를 위해 차량 영상을 입력하면 416x416 크기로 정규화하고 컨볼루션을 통한 피쳐 추출 과정을 거친 다음 3개의 스케일별로 각각 객체 탐지를 수행한다. 각 스케일별로 추정한 Bounding 박스들의 객체 존재 유무 및 클래스 확률을 바탕으로 최종 객체 탐지 결과를 출력한다. 정상적인 YOLOv4 객체 탐지 과정을 수행한 다음 차량이 탐지되지 않은 주차 면에 대해 그리드 셀의 위치를 대각선 방향 또는 상하좌우 방향으로 이동시켜 추가로 객체 탐지를 수행한다. 그리드 셀을 이동시키면 셀 간격에 따른 Anchor 박스 추론 오차를 줄이면서 추가로 객체 탐지를 시도할 수 있기 때문에 객체 탐지 정확도를 개선시킬 수 있다.

주차 차량 탐지 애플리케이션의 실시간 객체 탐지 허용 시간 범위 내에서 적절하게 셀 이동 방향의 종류 및 횟수를 설정할 수 있다. 다음 절부터 본 논문에서는 제안한 알고리즘에 대한 단계별로 설명한다.



<그림 2> 그리드 셀 조정 방식의 YOLOv4를 이용한 주차 차량 탐지 알고리즘 개요

Ⅲ. 셀 조정에 의한 YOLOv4 차량 객체 탐지 성능 개선

야외 노상 주차장에 주차된 차량을 YOLOv4를 이용하여 탐지하기 위해 CCTV로 주차 차량들의 영상을 촬영하여 데이터 셋을 만들어 학습을 시킨 다음 학습 결과 가중치 파일을 이용하여 차량 객체 애플리케이션을 구현한다. 차량을 탐지한 결과는 주차면 정보와 정합하여 주차면의 차량 주차 유무를 판단할 수 있다. 특히 차량이 주차되지 않은 주차 면에 대해 YOLOv4 셀 조정 알고리즘을 적용하여 추가로 차량 객체 탐지를 시도하면 주차 차량의 탐지 정확도를 높일 수 있다.

3.1 YOLOv4를 이용한 주차 차량 탐지 학습

야외 노상 주차 면은 평행 주차 또는 직각 주차를 할 수 있도록 구획 되어 있으며 현장 상황에 따라 여러 주차 면에 대해 한 대의 카메라로 차량 탐지가 가능한 곳도 있지만 한 주차 면에 하나의 카메라를 사용해야하는 경우도 있다. 따라서 다양한 형태로 주차된 차량 영상을 촬영하여 차량 객체를 라벨링할 필요가 있다. <그림 3>은 야외 노상 주차 차량 탐지 학습을 위해 여러 대의 차량을 주차한 주차 면들과 한 대의 차량이 주차된 주차 면에 대한 영상들을 라벨링한 예를 보인 것이다.

<그림 3> 야외 노상 주차 차량 영상들을 라벨링한 예

야외 노상 주차 차량 탐지 용도에 따라 각 주차 면의 차량 주차 유무만 판단할 때는 여러 주차 면에 대해 하나의 카메라로 차량 객체들을 탐지할 수 있고 주차 면에 차량 주차 유무의 판단과 동시에 번호판까지 인식할 경우 한 주차 면에 대해 하나의 카메라로 차량 객체를 탐지하게 된다.

YOLOv4의 주차 차량 탐지 학습을 위해 여러 주차 면에 주차된 차량들의 영상과 한 주차 면에 한 대의 차량을 주차한 영상들을 동시에 수집하여 차량 객체에 대한 라벨링을 수행하고 아래와 같은 방식으로 YOLOv4를 학습시켰다.

① YOLOv4 개발환경 설정

YOLOv4 차량 객체 탐지 학습을 위해 GPU 환경에서 darknet YOLOv4를 실행시킬 수 있도록 CUDA 및 cuDNN을 설치한다[8, 9]. CUDA는 GPU에서 병렬처리를 가능하게 해주는 기술이고 cuDNN은 딥러닝 신경망의 동작 루틴을 고속으로 병렬 처리할 수 있도록 지원해주는 GPU 가속화 라이브러리이다. CUDA 및 cuDNN 라이브러리는 GPU 하드웨어 규격에 종속적이기 때문에 컴퓨터에 탑재된 GPU 종류에 적합한 버전을 선택한다.

② YOLOv4 학습용 파라미터 설정

라벨링한 차량 영상들을 대상으로 YOLOv4를 학습하기 위해 그리드 셀에 배치할 최적의 Anchor 박스 크기를 추정해야한다. darknet YOLOv4에서는 학습 대상 영상들을 라벨링한 결과를 분석하여 최적의 Anchor 박스 크기를 추정하는 기능을 제공하고 있다. 이를 이용하여 셀에 적용할 9개의 Anchor 박스들의 최적 크기를 추정한 다음 YOLOv4 학습을 위한 하이퍼 파라미터들이 정의된 구성 파일의 Anchor 박스 크기 항목을 수정한다.

③ YOLOv4 커스텀 학습

차량 객체를 라벨링한 영상들과 YOLOv4 학습을 위한 하이퍼 파라미터들이 정의된 구성 파일 그리고 미리 학습된 pre-trained weights 파일을 이용하여

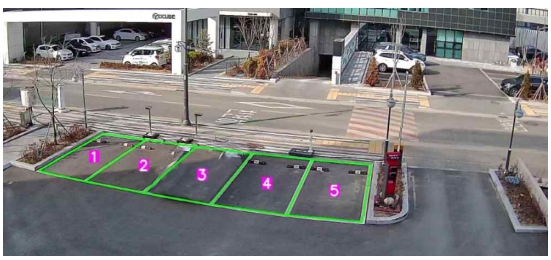
darknet YOLOv4 객체 탐지 학습을 수행한다.

학습이 종료되면 YOLOv4 학습에 사용된 구성 파일과 학습 결과로 생성된 가중치 파일을 이용하여 차량 객체 탐지를 위한 애플리케이션 프로그램을 작성할 수 있다.

3.2 YOLOv4 기반의 주차 차량 탐지

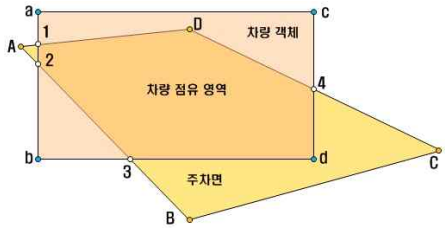
YOLOv4를 이용하여 차량 객체 탐지 학습을 완료한 다음 야외 노상 주차 면의 차량 객체 탐지를 위한 애플리케이션을 구현한다. 야외 노상 주차장에 그려진 주차 면과 탐지된 차량 객체의 최소 외접 사각형(Minimum Boundary Rectangle)의 정합 정도를 계산하여 차량의 주차 유무를 판단할 수 있다.

차량 객체의 탐지 영역인 야외 노상 주차장의 주차 면은 촬영하는 카메라의 위치에 따라 다양하게 기울어진 형태로 표현될 수 있다. <그림 4>에 5개의 주차 영역으로 구성된 주차 면을 구획한 예를 도시하였다.



<그림 4> 차량 객체 탐지 대상 주차 면의 구획을 설정한 예

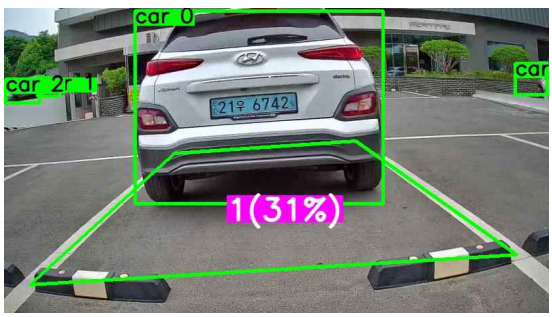
YOLOv4를 이용해서 차량 객체들을 탐지한 다음 주차 면의 사각형 영역과 정합(matching)을 시켜 해당 주차 면에 대한 차량의 주차 유무를 판단할 수 있다. 차량 객체의 탐지 결과는 주차 면의 기울어진 사각형역과 다르게 직사각형의 최소 외접 사각형으로 표현된다. 따라서 <그림 5>에 보인 것과 같이 기울어진 주차 면의 사각형 영역과 직사각형 차량 객체가 겹쳐진 영역을 크기를 구한 다음 주차 면에 대한 차량 점유율을 계산하고 그 결과를 이용하여 해당 주차 면에 대한 주차 유무를 판단할 수 있다.



<그림 5> 주차 면에 대한 차량 점유율 계산하기 위한 개념도

노상 주차장에 구획된 주차면 ABCD 영역과 차량 객체의 최소 외접 사각형 abcd 영역의 크기를 구한 다음 두 영역이 겹쳐진 123d4D 꼭지점들로 이루어진 다각형을 찾아서 그 영역의 크기를 구하면 주차 면에 대한 차량의 점유율을 계산할 수 있다.

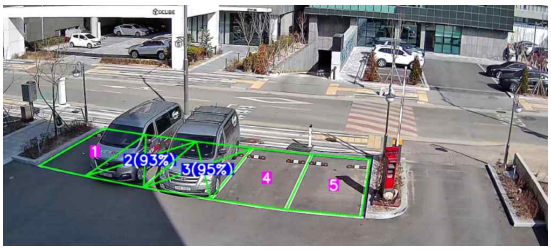
YOLOv4를 이용하여 차량 객체들을 탐지한 다음에 주차면 영역과 정합시켜 해당 주차면의 차량 점유율을 계산한 예를 <그림 6>에 도시하였다. 주차 면의 점유율에 대한 임계치를 설정하고 일정 임계치 이상일 때 차량이 주차된 것으로 간주하여 차량 번호를 인식할 수 있다.



<그림 6> 탐지된 차량 객체의 주차 면의 점유율 추정 예

다중 주차 면에 대해서도 동일한 방법으로 탐지된 차량 객체들의 최소 외접 사각형을 각 주차 면에 정

합시키고 해당 주차면의 주차 유무를 판단한 예를 <그림 7>에 도시하였다. 원거리에서 촬영한 다중 주차 면들에 대한 차량 점유 유무를 판단하면 주차 차량 대수 및 주차 가능 주차 면의 정보 등을 추정할 수 있다.



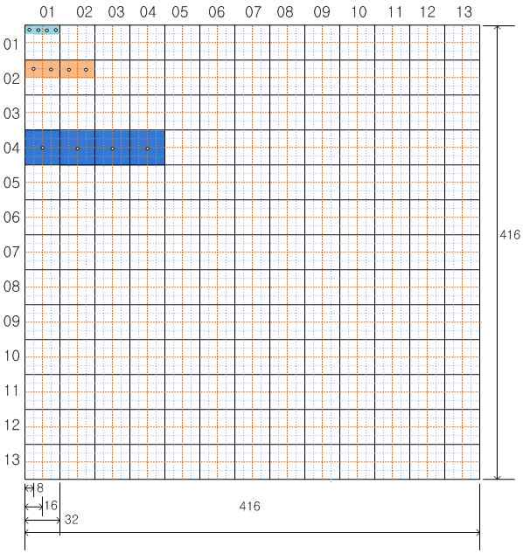
<그림 7> 다중 주차 면에 대한 차량 객체들의 점유율 계산 예

3.3 YOLOv4 셀 조정 알고리즘

YOLOv4를 이용하여 차량 객체들을 탐지하고 주차 면들과 정합시켜 해당 주차 면에 대한 차량 주차 유무를 판단할 때 초기에 주차된 차량의 탐지에 실패할 경우 연속적으로 입력되는 영상 프레임에서도 동일한 위치와 형태의 차량 영상이 입력되기 때문에 차량 탐지에 실패할 확률이 높아진다. 따라서 주차 면에 차량이 모두 점유 되지 않았을 경우 동일한 영상 프레임에 대해 YOLOv4 알고리즘의 객체 추론 방식을 다르게 하여 추가로 차량 객체 탐지를 시도하면 차량 객체 탐지 정확도를 높일 수 있다.

YOLOv4 알고리즘에서는 입력 영상을 416x416 크기로 정규화하고 이를 13 x 13개의 그리드 셀로 구분한 다음 각 그리드 셀마다 9개의 Anchor 박스를 배정하여 객체의 존재 유무 및 객체의 종류를 추정한다. 이 때 Anchor 박스는 그리드 셀 별로 탐지 위치를 설정하기 때문에 인접한 그리드 셀들 사이의 거리 차이로 인해 Anchor 박스를 기준으로 실제 차량 객체들의 Bounding 박스를 추정할 때 오차가 발생할 수 있다. 그리드 셀의 위치를 이동시키면 해당 셀이 할당된 Anchor 박스도 이동하게 되어 실제 탐지 대상 객체와 정합을 시도하는 위치를 달리하면서 탐지 시도 횟수가 늘어나기 때문에 좀 더 정확하게 객체를 추론할 확률이 높아지게 된다.

<그림 8>에 3가지 스케일별로 분할되는 YOLOv4의 그리드 셀의 개념도를 도시하였다.



<그림 8> 3가지 스케일별로 분할한 그리드 셀의 개념도

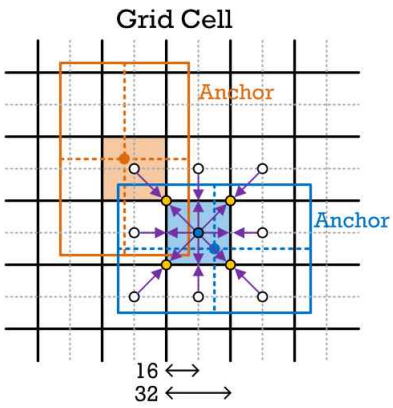
YOLOv4[3]에서는 <그림 8>에 보인 것과 같이 416x416 크기로 정규화한 영상에 대해 가로x세로 방향으로 13x13, 26x26 및 52x52개의 셀로 각각 분할하여 3개 스케일로 나누어 피쳐 맵을 구성한다. 각 스케일별 피쳐 맵에서 한 개 셀의 크기는 32x32, 16x16 및 8x8 이 되어 각각은 큰 객체, 중간 객체 그리고 작은 객체들을 탐지 대상으로 한다.

YOLOv4에서 각 스케일별로 1개 그리드 셀에 3개의 Anchor 박스를 배정하면 13x13x3= 507, 26x26x3= 2,028 그리고 52x52x3= 8,112개가 되고 이 Anchor 박스들을 이용하여 객체의 Bounding 박스들을 추정할 수 있다. YOLOv4에서는 할당된 Anchor 박스를 기준으로 오프셋(offsets)을 이용하여 객체의 Boundng 박

스 위치 및 크기를 추정한다. 3개의 스케일에서 추정한 전체 10,847개의 Bounding 박스들에 대해 NMS(Non-Max Suppression)을 적용하여 최종 탐지된 객체들을 출력하게 된다.

학습용 영상 객체들을 대상으로 각 그리드 셀에 할당된 Anchor 박스들을 이용하여 객체의 Bounding 박스를 추정하는 훈련을 수행한 다음 애플리케이션 영상을 대상으로 객체 탐지를 시도할 경우 그리드 셀 사이의 간격에 따라 Anchor 박스 위치에 의한 Bounding 박스 추정 오차가 발생할 수 있다. 따라서 기본적인 객체 탐지를 수행한 다음에 그리드 셀의 위치를 이동시키면 Anchor 박스도 이동되기 때문에 다른 위치에서 객체 탐지를 시도할 수 있어서 탐지 정확도를 높일 수 있는 기회를 갖게 된다. YOLOv4에서는 각 그리드 셀에 할당되는 Anchor 박스의 중심이 해당 그리드 셀에 포함되어야 한다. 따라서 그리드 셀을 이동시키면 Anchor 박스의 위치도 같이 이동하게 된다.

<그림 9>에 그리드 셀들에 할당된 Anchor 박스와 셀 조정 개념을 도시하였다.



<그림 9> 그리드 셀의 분할 오차에 따른 셀 조정 개념도

<그림 9>에 셀들의 간격에 따른 Anchor 박스의 추론 오차를 줄이기 위해 그리드 셀을 이동시킬 수 있는 8개 방향 및 이동 거리를 화살표로 표현하였다. 그리드 셀을 여러 방향으로 이동시켜 추가적인 객체 탐지를 시도할 경우 객체 탐지에 소요되는 시간이 길어지기 때문에 어느 방향으로 몇 회 이동시킬지는 주차 차량 탐지 애플리케이션의 용도와 GPU의 성능을 고려해서 설정할 수 있다.

제안한 방식을 이용하면 야외 노상 주차 차량 객체 탐지를 위해 주차 면에 차량이 탐지되지 않았을 때 그리드 셀 조정을 통해 추가로 차량 객체 탐지를 시도할 수 있어서 객체 탐지 정확도를 높일 수 있다.

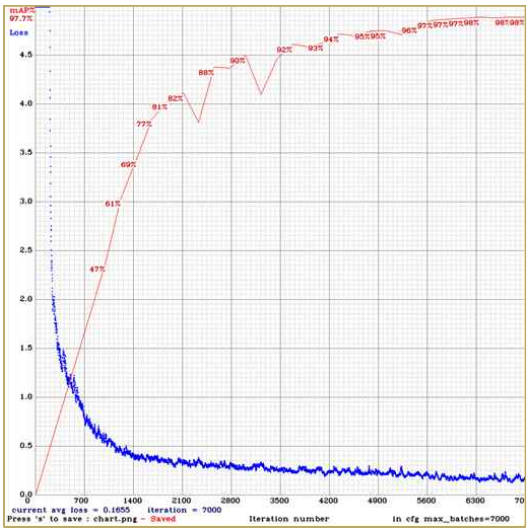
IV. 실험 및 결과 고찰

본 논문에서 제안한 야외 노상 주차장 주차 차량 탐지를 위한 YOLO 셀 조정 알고리즘을 구현하고 실제 야외 주차장의 주차 차량들을 대상으로 탐지 성능을 실험하였다. <그림 10>에 주차 차량 객체 탐지 실험을 위해 Visual Studio 2019를 기반으로 YOLOv4 MFC 애플리케이션을 작성한 예를 도시하였다.

<그림 10> 노상 주차 면 차량 탐지를 위한 인터페이스

실험을 위해 NVIDIA GeForce RTX 3080 GPU가 탑

재된 컴퓨터에 CUDA 11.2 및 cuDNN 11.2 라이브러리를 설치하고 Visual Studio 2019 기반의 Darknet YOLOv4 MFC 애플리케이션 프로그램을 작성하였다. Darknet에서 제공하는 YOLOv4 라이브러리의 차량 클래스는 MS-COCO 데이터 셋의 12,251개 차량 영상에 라벨링된 12,247대의 차량 객체를 대상으로 학습된 것이다. 본 연구에서는 MS-COCO 데이터 셋으로부터 주차 차량 영상 6,520장을 추출하여 라벨링한 6,520개 차량 객체에 야외 노상 주차장에 주차된 차량들을 촬영한 8,150장의 영상을 대상으로 라벨링한 8,150대의 차량 영상 객체를 추가하여 전체 14,670장의 영상에 포함된 차량 객체들을 학습시켰다. <그림 11>에 YOLOv4를 이용하여 차량 영상 객체 탐지를 위한 학습과정에서의 mAP 및 Loss 변화 그래프를 도시하였다. MS-COCO 데이터 셋의 차량 영상과 노상 주차 차량을 촬영한 영상에 대해 7,000회 반복 학습을 수행한 결과 mAP=97.7%, Avg. Loss= 0.1655를 나타내었다.



<그림 11> YOLOv4 학습과정에서 mAP와 Loss 변화 그래프

YOLOv4를 이용한 학습을 수행하고 Darknet YOLOv4 dll 라이브러리, 학습 결과 가중치 파일 및 구성파일 등을 이용하여 MFC 애플리케이션 프로그램을 작성하고 주차 차량 객체 탐지 성능을 분석하였다. 학습에 참여하지 않은 1,520장의 주차 차량 영상에 대해 원본 YOLOv4와 제안한 알고리즘이 구현된 YOLOv4을 이용한 탐지 성능을 비교 분석하였다.

주차 차량의 탐지 성능을 분석하기 위해 800x448 24비트로 촬영한 1,520장의 차량 영상에 대해 YOLOv4 원본 버전과 주차 차량 영상을 학습시킨 YOLOv4 학습 버전의 차량 객체 탐지 성능을 비교한 결과를 <표 1>에 도시 하였다.

<표 1> YOLOv4 원본과 주차 차량 학습 결과 차량 탐지 성능

Algorithm	Detection	Speed/frame
YOLOv4 원본	1,361(89.5%)	0.027sec
YOLOv4 학습	1,438(94.6%)	0.028sec

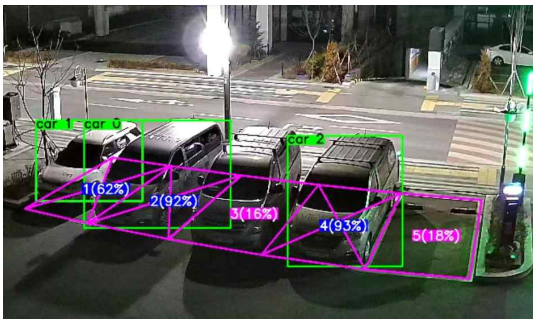
YOLOv4 원본 학습 yolov4.weights 파일을 이용한 탐지에서는 89.5%의 탐지 성능을 보였고 영상 프레임당 평균 0.027초의 처리 속도를 보였다. 차량 영상을 추가로 학습한 YOLOv4 학습 버전의 경우 처리속도는 비슷하지만 94.6%의 탐지 성능을 보여 주차 차량 전용 탐지 애플리케이션의 성능이 높게 나타나는 것을 확인할 수 있었다. 다른 데이터베이스를 대상으로 한 기존 연구[1]의 탐지율 93.3%보다 높게 나타났다.

<표 2> YOLOv4 기반의 셀 조정 알고리즘 차량 탐지 성능

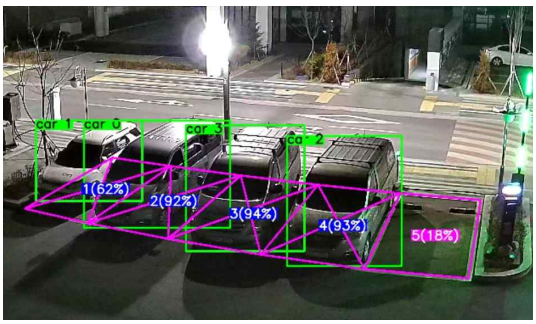
Algorithm	Detection	Speed/frame
YOLOv4 학습	1,438(94.6%)	0.028sec
+ 셀 이동(↖)	1,465(96.4%)	0.046sec
+ 셀 이동(↘)	1,467(96.5%)	0.045sec
+ 셀 이동(↗)	**1,468(96.6%)**	0.046sec
+ 셀 이동(↙)	1,465(96.4%)	0.046sec
+ 셀 이동(←)	1,456(95.8%)	0.044sec
+ 셀 이동(→)	1,456(95.8%)	0.045sec
+ 셀 이동(↑)	1,458(95.9%)	0.046sec
+ 셀 이동(↓)	1,455(95.7%)	0.045sec

<표 2>에 YOLOv4 주차 차량 탐지 학습 버전에 각 방향으로 셀 조정 알고리즘을 추가했을 때의 차량 탐지 성능을 도시하였다. 대각선 방향으로 셀을 이동시켰을 때 상하좌우 방향으로의 셀 이동 보다 차량 객체 탐지 성능이 조금 높게 나타난 것을 알 수 있다. 영상 프레임당 차량 탐지 속도는 두 배로 증가되었으며 셀 이동 방향에 따른 처리 속도 차이는 거의 없었다. 주차 차량 탐지 애플리케이션의 용도에 따라 허용되는 실시간 탐지 범위 내에 실행될 수 있도록 GPU를 선택하고 셀 이동 방향들을 추가한다면 전체 차량 탐지 성능이 증가될 수 있음을 보였다.

YOLOv4 주차 차량 탐지 학습을 수행한 버전의 차량 탐지 결과와 셀 조정을 추가한 버전 각각의 차량 탐지 결과와 주차 면에 대한 점유율을 계산한 예를 <그림 12>에 도시하였다.



(a) YOLOv4 학습 버전 차량 탐지 결과



(b) YOLOv4 학습 + 셀 조정 버전 차량 탐지 결과

<그림 12> YOLOv4 학습 버전 및 셀 조정을 추가한 버전의 차량 객체 탐지 결과 및 주차 면 점유율을 계산한 예

YOLOv4 차량 탐지 학습을 수행한 버전에서도 차량들을 잘 탐지할 수 있었지만 셀 조정을 추가함으로서 이전에 탐지되지 않은 차량들까지 탐지하는 결과를 볼 수 있었다. 차량 객체를 탐지한 결과 Bounding 박스와 주차면의 영역 정보를 정합시켜 차량 점유율을 계산하고 해당 주차면의 차량 주차 유무를 판단할 수 있다.

본 논문에서 제안한 주차 차량 탐지를 위한 YOLOv4 셀 조정 알고리즘을 적용한 차량 탐지 솔루션은 야외 노상 주차 면에 주차된 차량의 주차 유무를 판단하는데 효율적으로 활용할 수 있음을 보였다.

V. 결론

본 논문에서 제안한 노상 주차 차량 탐지 성능 개선을 위한 YOLOv4 그리드 셀 조정 알고리즘을 RTX3080 GPU, CUDA 11.2 및 cuDNN 11.2 라이브러리 환경에서 MFC 기반으로 구현하여 차량 탐지 실험을 수행하고 그 결과를 분석하였다.

주차 차량 전용 영상 데이터 셋을 추가로 학습시킨 결과 YOLOv4 원본 버전보다 차량 탐지 성능이 높게 나타났으며 처리 속도에는 차이가 없었다. YOLOv4 차량 탐지 학습을 시킨 버전에 셀 조정을 추가로 적용하여 차량 탐지 실험을 수행한 결과 대각선 방향의 셀 이동이 상하좌우 방향의 셀 이동에 비해 차량 탐지 성능이 조금 높게 나타났으며 이동 방향에 따른 탐지 속도 차이는 거의 없는 것을 확인하였다.

제안한 주차 차량 탐지 방법을 이용하면 노상 주차면의 차량 주차 유무를 판단하는데 효과적으로 활용할 수 있고 GPU 성능 및 실시간 처리 요구 조건에 따라 셀 이동 방향을 추가할 수 있기 때문에 노상 주차 차량 객체 탐지 성능을 더 높일 수 있음을 확인하였다.

참고문헌

[1] X. Ding and R. Yang, "Vehicle and Parking Space Detection Based on Improved YOLO Network Model," pp.1-7, ICAITA 2019.

[2] P. Kalebere, G. Deshpande, P. Gavade, H. Pudale and A. Jagtap, "Automated Parking Space Detection using Darknet, YOLOv3 and Android," pp.524-536, Vol. 10, Issue 5, J. of Information and Computational Science, 2020.

[3] A. Bochkovskiy, C. Wang and H. Liao, "YOLOv4: Optimal Speed and Accuracy of Object Detection," arXiv preprint arXiv: 2004.10934, 2020.

[4] A. Bathija and G. Shama, "Visual Object Detection and Tracking using YOLO and SORT," International Journal of Engineering Research, Vol. 8, Issue 11, 2019, pp.705-708.

[5] Deep Learning Tutorial, Release 0.1, LISA Lab, U. of Montreal, 2015.

[6] T. Liu, S. Fang, Y. Zhao, P. Wang and J. Zhang, "Implementation of Training Convolutional Neural Networks," arXiv:1506.01195v2, 2015.

[7] G. Sarker and S. Ghosh, "A Convolution Neural Network for Optical Character Recognition and Subsequent Machine Translation," Int. Journal of Computer Application, Vol. 182, No. 30, 2018,

[8] A. Sharma, "Achieving Optimal Speed and Accuracy in Object Detection (YOLOv4)," https://pyimagesearch.com/2022/05/16/achieving-optimal-speed-and-accuracy-in-object-detection-yolov4

[9] Yolo v4, v3 and v2 for Windows and Linux, https://github.com/AlexeyAB/darknet

■ 저자소개 ■

김 진 호
(Kim, Jin Ho)

1992년 3월~현재 경일대학교 전자공학과 교수
2002년 2월 경북대학교 전자공학과(공학박사)
1994년 2월 경북대학교 전자공학과(공학석사)
1992년 2월 경북대학교 전자공학과(공학사)

관심분야 : 인공지능, 딥러닝, 패턴인식
E-mail : anyface2009@gmail.com

논문접수일 : 2022년 12월 5일
수 정 일 : 2022년 12월 15일
게재확정일 : 2022년 12월 18일