ISSN 2089-8673 (Print) | ISSN 2548-4265 (Online)
Volume 12, Issue 3, December 2023

YOLOV4 DEEPSORT ANN FOR TRAFFIC COLLISION DETECTION

Arliyanti Nurdin[1], Bernadus Seno Aji[2], Yupit Sudianto[3], Mardhiyyah Rafrin[4]

[1,2]*Teknologi Informasi, Institut Teknologi Telkom Surabaya*
[3]*Sistem Informasi, Institut Teknologi Telkom Surabaya*
[4]*Ilmu Komputer, Institut Teknologi Bacharuddin Jusuf Habibie*

email: arliyanti.n@ittelkom-sby.ac.id[1], bernadus.seno@ittelkom-sby.ac.id[2],
yupit@ittelkom-sby.ac.id[3], rafrinmardhiyyah@ith.ac.id[4]

Abstract

Every collision must be handled right away to prevent further harm, damage, and traffic bottlenecks. Hence, the implementation of a systematic approach for accident detection becomes imperative to expedite response mechanisms. Our proposed accident detection system operates in three stages, encompassing vehicle object detection, multiple object tracking, and vehicle interaction analysis. YOLOv4 is employed for object detection, while DeepSort is utilized to the tracking of multiple vehicle objects. Subsequently, the positional and interactional data of each object within the video frame undergo thorough analysis to identify collisions, utilizing an Artificial Neural Network (ANN). Notably, collisions involving a single vehicle and not affecting other road users are excluded from the scope of this study. The evaluation of our approach reveals that the ANN model achieves a commendable F-Measure of 0.97 for detecting objects without collisions and 0.88 for objects involved in collisions, based on the conducted tests.

Keywords: traffic collision detection, YOLOv4, DeepSort, object detection, object tracking

Received: 07-06-2023 | **Revised:** 27-08-2023 | **Accepted:** 10-10-2023
DOI: https://doi.org/10.23887/janapati.v12i3.62923

INTRODUCTION

Transportation system is one of vital aspects in city developments. Therefore, urban transportation problems should be taken into account seriously. In Indonesia, the high number of traffic accidents is one of the most challenging and crucial problems. Delay in detecting traffic accidents may lead to more serious issues from traffic jams, injuries, damages and even material loss. Based on Statistics Indonesia (BPS), the number of traffic incidents in 2018 was 199,245 cases with 29,472 deaths and material loss of IDR 231,866,000,000 [1].

Monitoring traffic can be performed by installing surveillance cameras in several points on the road. The surveillance cameras record the traffic in real-time and the footage can be sent to a management system equipped with a smart transportation system such as automatic accident detection. Computer vision techniques are largely used for that [2]–[4].

When an accident occurs, there is a dramatic change of vehicles' speed and the distances among vehicles involved may be very close or even zero. Therefore, an accident can be detected by analyzing the interaction among vehicles. This analysis is attainable by detecting all object vehicles then tracking them until the change of vehicle positions in a time unit are collected. The change of the positions, therefore can be used to identify traffic incidents.

Currently, numerous studies have implemented computer vision technologies for detecting and tracking multiple objects in videos using deep learning algorithms. Some of them are Faster R-CNN [5], [6], InceptionResnetV2 [7], Deep Convolutional Neural Netwok [8], Mask R-CNN [9], YOLO [[10]–[12], Siamese Neural Network [13], LSTM [14], [15], Siamese Track-RCNN [[16], and developed ENet [17]. In this study, we implement YOLOv4 [18] to detect objects. The performance of YOLO in image processing is undoubtedly faster in real time, noted at 45 frames per second. We use DeepSort [19]–[21] to track multiple vehicles from a dataset of collected videos. This approach applied CNN based detection to detect multiple objects, adopted conventional tracking method of recursive Kalman Filtering and method of association data frame-by-frame to track object movements.

The examination of object movement within each frame yields valuable data that can serve as parameters for the analysis of accident detection. Such parameters encompass the inter-object distance, relative speed, positional coordinates, directional vectors, and tilt angles associated with the movement of objects. Diverging from an approach reliant on a rule-based algorithm for accident detection on freeways or highways, as illustrated by the calculation of the accident index based on object parameters [22], our methodology employs a machine learning algorithm—specifically, the Artificial Neural Network (ANN)—for the analysis of vehicle interactions. This study introduces a collision detection system structured in three sequential stages: initial vehicle object detection through YOLOv4, subsequent tracking of multiple objects utilizing DeepSort, and ultimate analysis of vehicle interactions employing the Artificial Neural Network (ANN).

METHOD

The architecture of algorithms for detecting and tracking vehicle objects is illustrated in the Figure 1. This system runs on GPU NVDIA GeForce GTX 165 and is developed with python version 3.8, TensorFlow 2.0, and OS Windows.

Dataset

The processing of vehicle detection using YOLOv4 requires many images of vehicles as a training dataset. Those images are collected from *Microsoft Common Objects in Context* or MSCOCO [23] and *Open Image Google.* MSCOCO consists of a large-scale dataset or 2,5 million instances of objects labelled from 328.000 images. In addition, some data get customized label using *LabelImg* tool with YOLO format.

Beside those image data, videos of traffic are also equally important to the training process of the tracking model. Total 300 videos of traffic with 1560x720 resolution and 30 frame rate from YouTube and the Department of Transportation of Surabaya are collected as the dataset for this system. The videos duration is cut to approximately 10 seconds.

Object Annotation

Annotating is a process for labelling objects by drawing a bounding box over the objects then classifying them into object classes. A single image may consist of one or more objects and those objects belong to either the same or different classes.

Annotating objects is simply doable by dragging a cursor to the targeted objects. The examples of annotating a motorcycle or a car are displayed in the Figure 2 and 3. The result which contains information of box location together with their annotating classes are saved into a .txt file as displayed in the Figure 4. The information structure starts with object classes in the first column followed by the bounding box position in the next four columns.

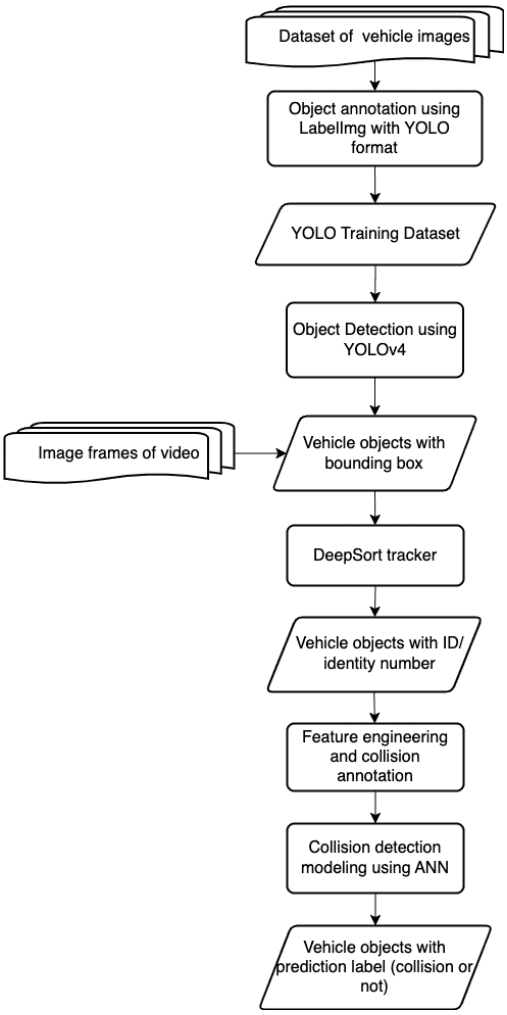


Figure 1. Architecture of YOLOv4 + DeepSort + ANN for Collision Detection System

ISSN 2089-8673 (Print) | ISSN 2548-4265 (Online)
Volume 12, Issue 3, December 2023

Figure 2. Annotating Motorcycle Object using *LabelImg*

Figure 3. Annotating Car Object using *LabelImg*

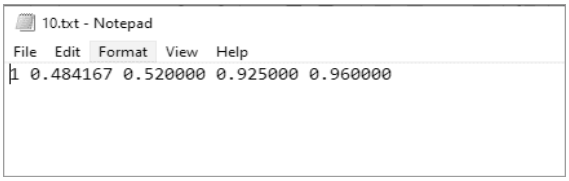


Figure 4. The Result of Annotating Objects using *LabelImg*

You Only Look Once (YOLO)

YOLO [24] uses convolutional neural network to predict the bounding box of detected objects in the images simultaneously and to directly classify them. Every image input is divided into *S x S* grids so that objects within every grid cell are detected. The model, therefore, predicts the objects within the grid cells and then it sets a confidence value in every box. The prediction process is performed by the convolutional network extracts the feature of the images and then fully connected layers detect the object completely. This process returns the output as a matrix with size *S x S x* (*5 x B* + *C*), where *B* is the number of bounding boxes, and *C* is a probability distribution.

YOLO uses the Sum-Squared Error (SSE) as a loss function [24].

$$SSE = E_1 + E_2 + E_3 + E_4 + E_5 \quad (1)$$

$$E_1 = \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} LF_{ij}^{object} [(x_i - \tilde{x}_i)^2 + (y_i - \tilde{y}_i)^2] \quad (2)$$

$$E_2 = \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} LF_{ij}^{object} [\left(\sqrt{\omega}_i - \sqrt{\tilde{x}_i}\right)^2 + \left(\sqrt{h_i} - \sqrt{\tilde{h}_i}\right)^2] \quad (3)$$

$$E_3 = \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} LF_{ij}^{object} (C_i - \tilde{C}_i)^2 \quad (4)$$

$$E_4 = \lambda_{no\_object} \sum_{i=0}^{S^2} \sum_{j=0}^{B} LF_{ij}^{no\_object} (C_i - \tilde{C}_i)^2 \quad (5)$$

$$E_5 = \sum_{j=0}^{B} LF_i^{object} \sum_{c \, \epsilon \, classes} (p_i^{(c)} - \tilde{p}_i^{(c)})^2 \quad (6)$$

where,
E_1 is xy_loss when the object exists at box_j in $cell_i$;
E_2 is wh_loss when the object exists at box_j in $cell_i$;
E_3 is $confidence\_loss$ when the object exists at box_j in $cell_i$;
E_4 is $confidence\_loss$ when the objects do not exist in the boxes;
E_5 is $class\_probability\_loss$ in the cell where the object exists.
$LF_{ij}^{object} = 1$ if in the i^{th} cell, there is a j^{th} box containing an object;
$LF_{ij}^{no\_object}$ is the opposite of LF_{ij}^{object};
$LF_{ij}^{object} = 1$ if in the i^{th} cell contains an object (otherwise, it is 0);
λ_{coord}, λ_{no_object} is the component weight.

The confidence value in every bounding box shows how confident the model is that the box contains objects and how accurate its prediction is. If there is no object detected, the confidence value is equal to zero. Every cell contains 5 prediction results as *x, y, w,* and *h* with confidence value in every cell. *x* and *y* are box coordinates relative to the border of the grid, *w* is the length and *h* is the height of the detected object relative to the whole image input. YOLO filters out redundant bounding boxes (duplicate and same class) using IOU formula that calculates the overlap–interference between two bounding boxes.

YOLOv4 was developed to improve the accuracy of YOLOv3. YOLOv4 architecture is divided into three main parts: Backbone, Neck and Head. The backbone utilizes CSPDarknet53 as a feature extractor [25]. CSPDarknet53 is built on a combination of CSP (Cross-Stage-Partial connections) and Darknet53. The backbone is comprised of 53

convolutional layers. The neck is SPP (Spatial Pyramid Pooling) and PAN (Path Aggregation Network) to collect feature maps from different stages of the backbone, and the head is YOLOv3 [26] as object detector. Figure 5 shows the architecture of the YOLOv4 [15].

As for video, all frames are extracted and input to the YOLOv4 object detector with image size configuration 416 x 416, IOU threshold 0.45, and confidence threshold 0.5. Each detected object is given a bounding box and is labelled with the object class. After that, this detection result is forwarded to the DeepSort tracker system. The architecture of YOLOv4 object classifier and DeepSort tracker system is depicted in Figure 1.

YOLOv4 has been the state-of-the-art object detector for real-time applications, an official version YOLO, before YOLOv7 [16] was released. In this study we use YOLOv4 because it is suitable for scenarios that demand precise object detection with high accuracy [17,18], while YOLOv7 offers a balance between speed and accuracy [19]. The accuracy comparison of YOLOv4 and YOLOv7 regarding vehicle detection in this dataset is left for future work.

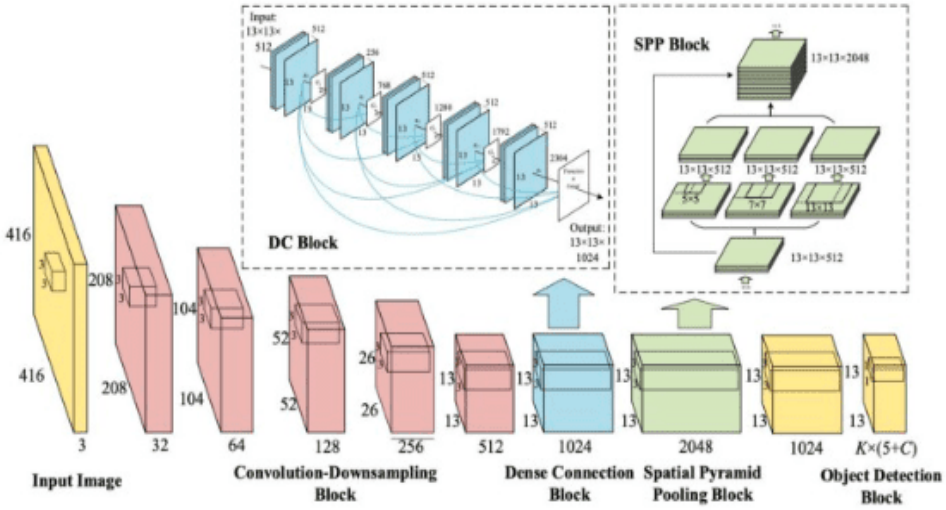


Figure 5. The Architecture of The YOLOv4 [27]

DeepSort

DeepSort was develop as an extension of the SORT (Simple Online Tracking) algorithm. DeepSort algorithm adds a deep appearance descriptor to reduce identity switches in SORT. This algorithm consists of four components: detection, estimation, data association, and creation and deletion of track identities. Figure 6. shows the components of DeepSort.

The first step is detecting the object, and then propagate the detections from current frame to the next to estimate position of the target in the next frame using Kalman prediction. After estimating position of the target, the distance between each detection and all predicted bounding boxes from existing targets are calculated using intersection-over-unit (IOU). The assignment is solved optimally using the Hungarian algorithm. The assignment is rejected if the IOU of detection and target is less than a certain threshold value.

DeepSort uses an association metric that combines both motion and appearance descriptors. DeepSort tracks objects not only based on the velocity and motion of the object but also the appearance of the object. A pre-trained CNN consisting of two convolutional layers followed by six residual blocks is used to compute bounding box appearance descriptors.

Mahalanobis distance is used to incorporate motion information between predicted Kalman states and newly arrived measurements:

$$d^{(1)}(i,j) = \left(d_j - y_i\right)^T S_i^{-1} \left(d_j - y_i\right) \qquad (7)$$

Where the projection of the *i*-th track distribution is denoted into measurement space by (y_i, S_i) and the *j*-th bounding box detection by d_j.

For each bounding box detection d_j, an appearance descriptor r_j with $\|r_j\| = 1$ is computed. A gallery is kept $\mathcal{R}_k = \{r_k^{(i)}\}_{k=1}^{L_k}$ of the last $L_k = 100$ associated appearance descriptors for each track *k*. The second metric for the smallest cosine distance measurement

ISSN 2089-8673 (Print) | ISSN 2548-4265 (Online)
Volume 12, Issue 3, December 2023

between the *i*-th track and *j*-th detection in appearance space:

$$d^{(2)}(i,j) = \min\{1 - r_j{}^T r_k{}^{(i)} \mid r_k{}^{(i)} \in \mathcal{R}_i\} \quad (8)$$

A binary variable is used to indicate if an association is admissible following to this metric.

$$b_{i,j}^{(2)} = \mathbb{I}[d^{(2)}(i,j) \leq t^{(2)}] \quad (9)$$

The Mahalanobis distance provides information about object location based on motion and the cosine distance considers appearance information. These metrics are combined using weighted sum.

$$C_{i,j} = \lambda d^{(1)}(i,j) + (1-\lambda) d^{(2)}(i,j) \quad (10)$$

Where an association is acceptable if it is within the gating region of both metrics:

$$b_{i,j} = \prod_{m=1}^{2} b_{i,j}^{(m)} \quad (11)$$

The impact of each metric on the combined association cost can be controlled by adjusting hyperparameter λ.

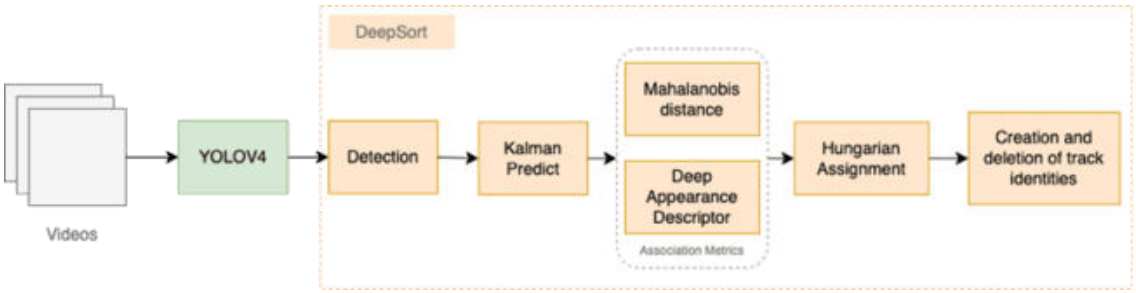


Figure 6. The Components of DeepSort

Similar to the study by [28], the process of object detection and tracking over time is shown in Figure 7. The system receives image frames of the selected videos in a certain time interval (*T)*, then the objects will be detected by YOLOv4 in order to simultaneously collect *BBox* coordinates and *class* $_T$ of every object.

After that, DeepSort object tracker will set ID_T number at every detected object and the predict the object position in the next frame, *BBox'*$_T$ based on the information from the YOLOv4. DeepSort Tracking object utilize *Intersection Over Unit* (IOU) concept in order to implement Kalman filter and association of *frame-by-frame* data to predict the object's next position [19]. The IOU between predicted position of BBox'T object at the time T and the position detected at T+1 or $BBox_{T+1}$ is calculated in every frame alteration. The pair of objects with highest IOU score is the assumed as same object as the ID. *BBox'*$_T$ without pair of object with the IOU score higher that the threshold at T+1 is assumed to be gone or out of *Region of Interest* (ROI). Meanwhile the $BBox_{T+1}$ without pair of object with the IOU score higher that the *threshold* is assumed as a new object appeared in the ROI and therefore, it is given a new ID.

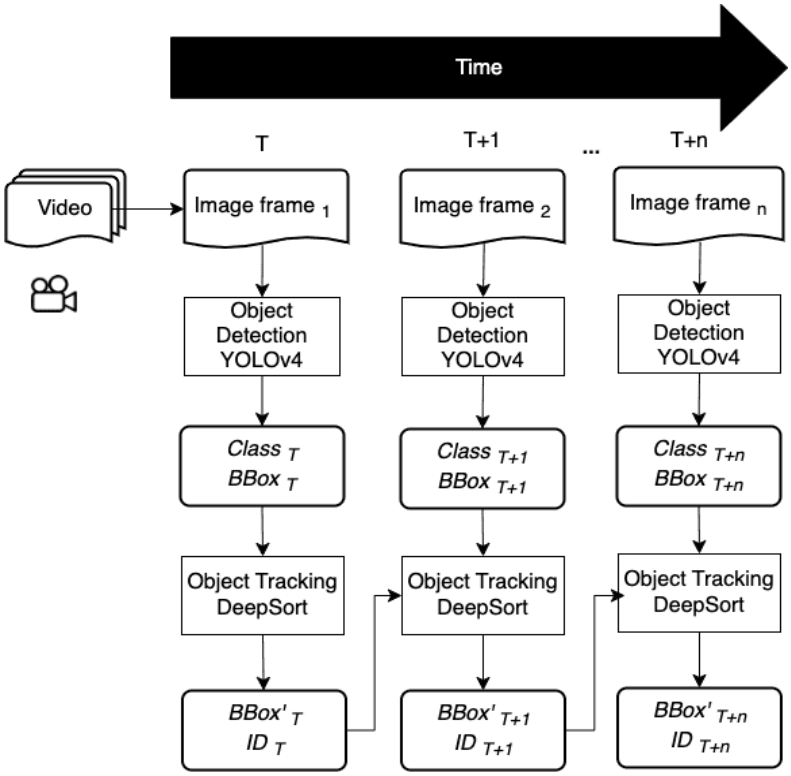


Figure 7. The Process of Object Detection by YOLOv4 and Tracking by DeepSort

Feature Engineering

Several features are extracted from bounding box information on each frame obtained from the detection and tracking process. These features include the position of objects in each frame, the distance between objects, and changes in the position of objects relative to other objects.

The features used are extracted from three sequential frames to obtain a pattern of changes in the position of each object that identifies an accident or collision. Table 1 describes the features used for collision detection using ANN algorithm.

Table 1. The Description of The Features for Collision Analysis

No	Feature	Description
1	dX_{n-2}	The change in the position of object X from $frame_{n-3}$ to $frame_{n-2}$
2	dX_{n-1}	The change in the position of object X from $frame_{n-2}$ to $frame_{n-1}$
3	dX_n	The change in the position of object X from $frame_{n-1}$ to $frame_n$
4	dY_{n-2}	The change in the position of object Y from $frame_{n-3}$ to $frame_{n-2}$
5	dY_{n-1}	The change in the position of object Y from $frame_{n-2}$ to $frame_{n-1}$
6	dY_n	The change in the position of object Y from $frame_{n-1}$ to $frame_n$
7	$(X\text{-}Y)_{n-2}$	The distance between X and Y objects in $frame_{n-2}$
8	$(X\text{-}Y)_{n-1}$	The distance between X and Y objects in $frame_{n-1}$
9	$(X\text{-}Y)_n$	The distance between X and Y objects in $frame_n$
10	$d(X\text{-}Y)_{n-2}$	The change in the position of object X relative to Y in $frame_{n-2}$
11	$d(X\text{-}Y)_{n-1}$	The change in the position of object X relative to Y in $frame_{n-1}$
12	$d(X\text{-}Y)_n$	The change in the position of object X relative to Y in $frame_n$

Changes in the position of each object in each frame are calculated using Euclidean Distance formula at the coordinates of the center point of bounding box object (x_{mid}, y_{mid}).

$$x_{mid} = (x_{min} + x_{max})/2 \quad (12)$$

$$y_{mid} = (y_{min} + y_{max})/2 \quad (13)$$

$$dN_i = \sqrt{(x_{mid(i)} - x_{mid(i-1)})^2 + (y_{mid(i)} - y_{mid(i-1)})^2} \quad (14)$$

where x_{min},x_{max},y_{min},y_{max} = the coordinate of bounding box object; dN_i = the change in the position of object N in the *i*-th frame.

Meanwhile, the distance between X and Y objects in *i*-th frame is calculated by the following equation.

$$(X-Y)_i = \sqrt{(X_{x_{mid(i)}} - Y_{x_{mid(i)}})^2 + (X_{y_{mid(i)}} - Y_{y_{mid(i)}})^2} \quad (15)$$

where $Xx_{mid(i)}$ = the x_{mid} coordinate of the object X in *i*-th frame, $Yx_{mid(i)}$ = the x_{mid} coordinate of the object Y in *i*-th frame, $Xy_{mid(i)}$ = the y_{mid} coordinate of the object X in *i*-th frame, and $Yy_{mid(i)}$ = the y_{mid} coordinate of the object Y in *i*-th frame.

The change in the position of object X relatives to object Y in *i*-th frame, $d(X\text{-}Y)_i$, is calculated by the following equation.

$$d(X-Y)_i = (X-Y)_i - (X-Y)_{i-1} \quad (16)$$

Collision Labeling

Each detected object with an identity number in each frame is labeled with condition status, collision or not.
The interaction of each pair of objects in one frame is analysed to detect collisions. Each pair of objects in the same frame will be labeled with a number 1 or 0 indicating whether a collision occurred or not. In Figure 8., the object with IDE 15 and 16 in *frame* 156 will be labeled 1 because these objects collide with each other, and others labeled 0.

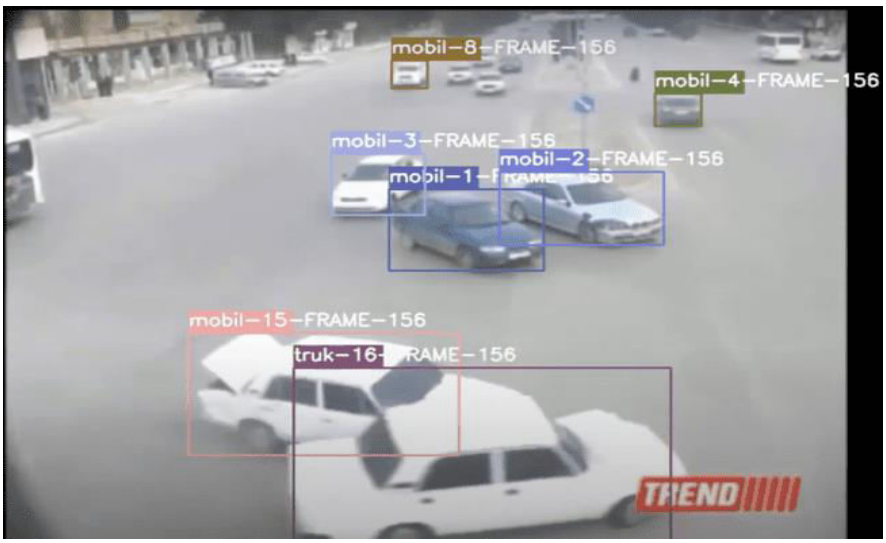


Figure 8. The Collision and Non-Collision Objects

Collision Detection using ANN

The dataset consisting of 12 numeric features labeled 1 or 0 is divided into training data and testing data. The training data is used to classification modelling using ANN algorithm with an architecture consisting of two hidden layer with 64 neurons in each layer and adam optimization algorithm.

RESULT AND DISCUSSION

After the training process of YOLOv4 and DeepSort have been done, the model is used to track vehicles on some traffic videos. Each video file is split into a collection of image frames and each frame is input to the YOLOv4 detector. The result of every object detected in each frame is tagged with bounding box and an IDE. After that, This bounding box and the IDE are used as input data to the DeepSort tracker for tracking process.

Performance measurement of the detection algorithm and object tracking system done by testing 4 different videos with various conditions and quality. Video #1 is low in lighting but has good quality, video #2 has occlusion condition or significant objects changes, video #3 shows sudden object movement and has blur quality, and the last video is taken at night with dark lighting.

Performance measurement is acquired by doing both quality and quantity analysis. Quantity performance measurement is done by using *Average Precision* (AP) and *Mean Average Precision* (mAP) with threshold IOU ≥ 50% as is depicted in Table 1. Objects predicted by the model must overlap at least 50% the corresponding ground truth object in order to be considered a correct detection. Average Precision is performance measurement for objects detected in each object class under the curve Precision x Recall. The Curve of Precision

x Recall can be seen in Figure 9. Meanwhile Mean Average Precision is average performance from all object classes.

Mean Average Precision (mAP) is calculated following the equation below.

$$mAP = \frac{1}{N}\sum_{i=1}^{N} AP_i \quad (19)$$

where N = The amount of AP data, AP = Average Precision.

And Average Precision is calculated following the equations below.

$$Precision = \frac{TP}{(TP+FP)} \quad (17)$$

$$Recall = \frac{TP}{TP + FN} \quad (18)$$

Where,
TP (True Positive) = The predicted data is TRUE and the actual data is also TRUE;
FP (False Positive) = The predicted data is TRUE and the actual data is FALSE;
FN (False Negative) = The predicted data is FALSE and the actual data is TRUE.

$$(IoU) = \frac{Area\ of\ Overlap}{Area\ of\ Union} \quad (19)$$

If the IoU value >= 0,5 then it has a TP (True Positive) value, while if IoU < 0,5 then it has a FP (False Positive) value.

$$AP = \sum_{k=0}^{k=n-1}[Recalls(k) - Recalls(k+1)] * Precisions(k) \quad (20)$$

where Recalls(n) = 0, Precisions(n) = 1, and n = number off threshold.

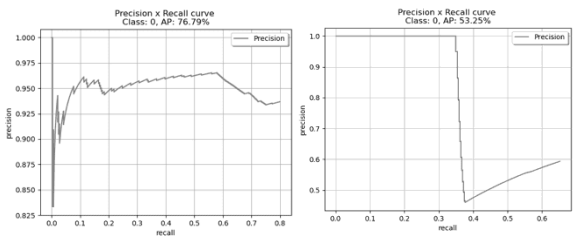
Quality analysis is done by observing the accuracy of the ID and the predicted classes from the object bounding box in every video frame. The bounding box visualisation collected from tracking objects can be seen at Figure 10., Figure 11., Figure 12. and Figure 13.

Table 2. Quantity Analysis

Video #	Number of Frames	Class Object	AP (%)	mAP (%)
Video #1	274	0 - Car	76.79	76.79
Video #2	194	0 - Car	53.25	53.25
Video #3	482	0 - Car	67	33.5
		1 - Motorcycle	0	
Video #4	217	0 - Car	54.81	13.7
		1– Motorcycle	0	
		2 - Truck	0	
		4 - Bicycle	0	

The accuracy of vehicles tracked by YOLOv4 and DeepSort is very good with AP and mAP 76,79% as depicted in Figure 10. Most cars can be detected and identified with the correct ID in every frame change.

In this research, YOLOv4 and DeepSort have successfully detected vehicles from different traffic videos. However, this model has some limitations to certain conditions as depicted in Figure 11., Figure 12. and 13. Some challenges in developing reliable visual tracking are various illumination on videos, dramatic changes of moving objects, occlusion, sudden and fast movement, complex background images, camera resolutions and even shadows [29], [30].



video #1 video #2

ISSN 2089-8673 (Print) | ISSN 2548-4265 (Online)
Volume 12, Issue 3, December 2023

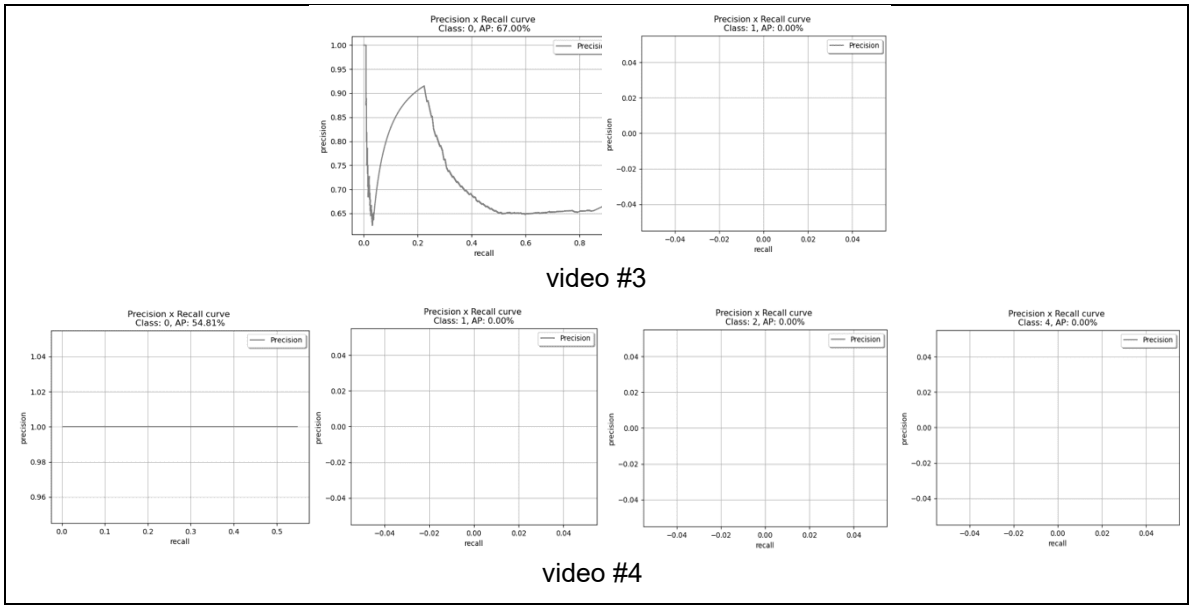
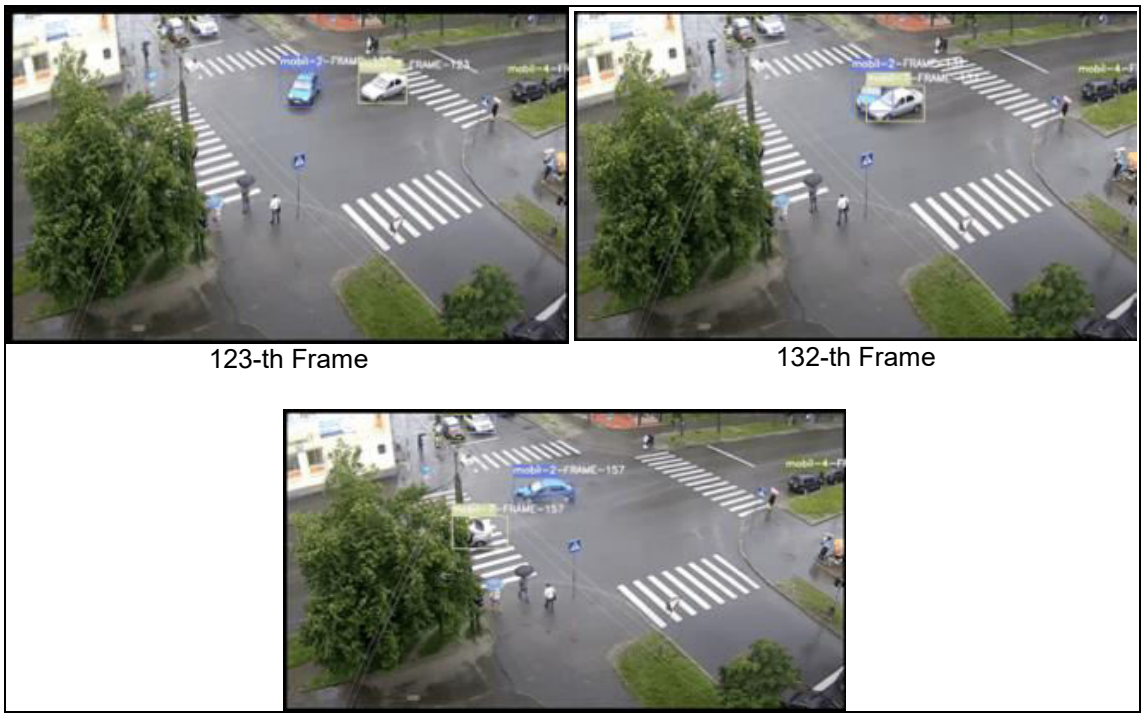


Figure 9. Curve of Precision x Recall

As shown in Figure 11., a collision between two different objects at frame 137th were detected as other objects at frame 169th due to fast movement of the cars. The Car with the ID 8 became the car with ID 15 and the truck with the ID 10 turned to become the ID 4. After the collision, the car with the ID 10 blocked the car with the ID 8 from the camera. This alteration is called occlusion. Therefore, when the car with ID 8 appeared in the next frames, it would be detected as a new object. Not only that, the truck with the ID 10 experienced a leaning and therefore the appearance was detected differ.

Based on the measurement result in the Table 2, AP of detecting the vehicles in the video #2 is 53.25%. Furthermore, it can be seen that there is a class change in some frames. Generally, real objects can move within 3 dimensions of space. In this video, however, the movement can be only projected into two dimensions as an image order. Every object which rotates to the third axis may be detected as another object and be identified as an incorrect object class.



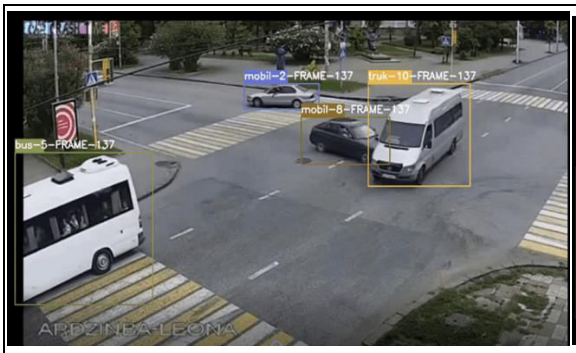
123-th Frame

132-th Frame

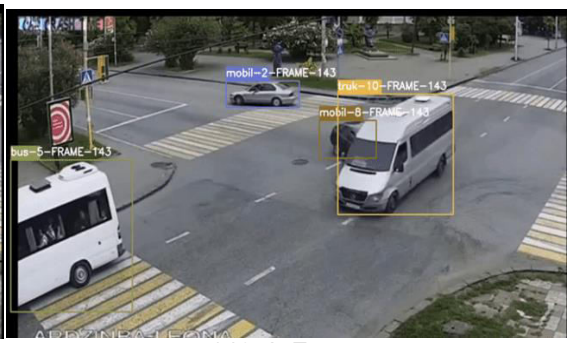
157-th Frame

Figure 10. Qualitative Analysis – Video #1

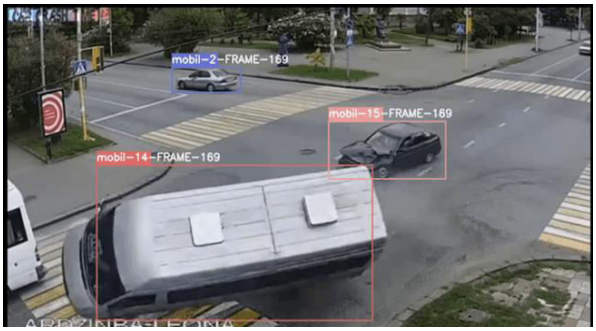
Another challenge of this detection model is that it hardly detects motorcycles in some video frames as depicted in Figure 12. The Figure 12 shows blur image of a motorcycle due to a very fast and sudden movement. In addition, the frames captured a bad visibility in the street at night. The bad lightening and the object fast movement decreased the tracker performance. The lighting in camera surveillance might change due to some circumstances such as the change of the position of the light sources, different times during the daylight, reflections from bright surfaces, weather etc. It can be seen in the video #4 in Figure 13. that only cars can be detected meanwhile trucks, bicycles, motorcycles are failed to be detected.



137-th Frame



143-th Frame



157-th Frame

Figure 11. Qualitative Analysis – Video #2

34-th Frame

40-th Frame

48-th Frame

Figure 12. Qualitative Analysis – Video #3.

30-th Frame

45-th Frame

50-th Frame

Figure 13. Qualitative Analysis – Video #4

Based on the measurement results of object detection and tracking operation on several videos, videos with mAP >70% were selected for further analysis. The bounding box information generated from YOLOv4 detection includes object ID, object class (car/bus/truck/motorcycle/bicycle), and the bounding box coordinates (*xmin, ymin, xmax, ymax)*, as shown in Figure 14. If there is an error label in object detection or tracking, it is corrected manually.

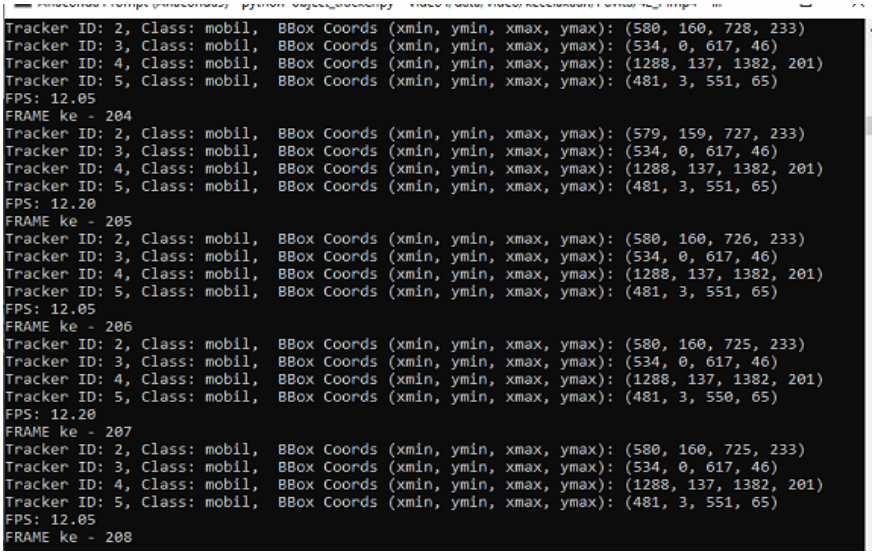


Figure 14. The Information Result of Object Detection and Tracking using YOLOv4 and DeepSort

Each object in each frame will generate a single data row consisting of 12 numeric features labelled 1 if in that frame the object encounters an accident or collision; labelled 0 if there is no collision occurs. This dataset is then classified using the ANN algorithm.

The following Table 3 shows the result of ANN performance in detecting collision using the Precision, Recall, and F-Measure evaluation metrics.

Table 3. ANN Performance in Collision Detection

Class	Precision	Recall	F-Measure
0	0,97	0,97	0,97
1	0,88	0,88	0,88

The ANN model can detect objects with class 0 or do not have a collision with F-Measure 0,97, and objects that have a collision with F-Measure 0,88.

CONCLUSION

YOLOv4 and DeepSort can detect and track several vehicle objects in real-time with good quality traffic video data, weather conditions, and lighting, with an Average Precision (AP) of 76,79%. There are several challenges in developing visual tracking methods such as variations in light illumination in the video, changes in the appearance of moving objects, occlusion, fast movements, complex background images, quality of camera resolution, and shadows. Therefore, additional processes such as image enhancement are needed before implementing the object detection algorithm. Information on the results of tracking vehicle objects in the form of the position of each object on the video frame is then analysed to build a collision detection model using a machine learning algorithm approach. By using the object position features in each frame, the distance between objects and changes in the object's position relatives to other objects from three sequential frames can identify the occurrence of an accident or collision. Based on test results, the ANN model can detect objects that have collision with F-Measure 0,97 and objects that have collision with F-Measure 0,88. In this study, collision detection was carried out by analysing the interaction between vehicles in each video frame, so that single accidents, namely accidents that only involve one vehicle and do not involve other road users, are not included in this study. There is also still a manual process for correcting the label errors on objects detected by YOLOv4 and DeepSort. The limitations of this study will be the development of further study.

REFERENCES

[1] B. P. STATISTIK, "Indonesia | English Jumlah Kecelakaan, Koban Mati, Luka Berat, Luka Ringan , dan Kerugian Materi yang Diderita Tahun 1992-2018," 2020.

[2] N. K. Jain, R. K. Saini, and P. Mittal, "A review on traffic monitoring system techniques," in *Advances in Intelligent*

Systems and Computing, Springer Verlag, 2019, pp. 569–577. doi: 10.1007/978-981-13-0589-4_53.

[3] K. K. Santhosh, D. P. Dogra, and P. P. Roy, "Anomaly Detection in Road Traffic Using Visual Surveillance: A Survey," *ACM Computing Surveys*, vol. 53, no. 6. Association for Computing Machinery, Feb. 01, 2021. doi: 10.1145/3417989.

[4] A. Fedorov, K. Nikolskaia, S. Ivanov, V. Shepelev, and A. Minbaleev, "Traffic flow estimation with data from a video surveillance camera," *J Big Data*, vol. 6, no. 1, Dec. 2019, doi: 10.1186/s40537-019-0234-z.

[5] N. Ran, L. Kong, Y. Wang, and Q. Liu, "A robust multi-athlete tracking algorithm by exploiting discriminant features and long-term dependencies," in *International Conference on Multimedia Modeling*, Springer, 2019, pp. 441–423.

[6] D. Yi, J. Su, and W. H. Chen, "Probabilistic faster R-CNN with stochastic region proposing: Towards object detection and recognition in remote sensing imagery," *Neurocomputing*, vol. 459, pp. 290–301, Oct. 2021, doi: 10.1016/j.neucom.2021.06.072.

[7] N. Pathik, R. K. Gupta, Y. Sahu, A. Sharma, M. Masud, and M. Baz, "AI Enabled Accident Detection and Alert System Using IoT and Deep Learning for Smart Cities," *Sustainability*, vol. 14, no. 13, p. 7701, Jun. 2022, doi: 10.3390/su14137701.

[8] D. Yang, Y. Wu, F. Sun, J. Chen, D. Zhai, and C. Fu, "Freeway accident detection and classification based on the multi-vehicle trajectory data and deep learning model," *Transp Res Part C Emerg Technol*, vol. 130, p. 103303, Sep. 2021, doi: 10.1016/j.trc.2021.103303.

[9] E. P. Ijjina, D. Chand, S. Gupta, K. Goutham, and B. Tech, "Computer Vision-based Accident Detection in Traffic Surveillance," *10th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, 2019, doi: 10.1109/ICCCNT45670.2019.8944469.

[10] M. A. A. Al-qaness, A. A. Abbasi, H. Fan, R. A. Ibrahim, S. H. Alsamhi, and A. Hawbani, "An improved YOLO-based road traffic monitoring system," *Computing*, vol. 103, no. 2, pp. 211–230, Feb. 2021, doi: 10.1007/s00607-020-00869-8.

[11] C. Dewi, R. C. Chen, X. Jiang, and H. Yu, "Deep convolutional neural network for enhancing traffic sign recognition developed on Yolo V4," *Multimed Tools Appl*, vol. 81, no. 26, pp. 37821–37845, Nov. 2022, doi: 10.1007/s11042-022-12962-5.

[12] C. Dewi, R. C. Chen, Y. T. Liu, X. Jiang, and K. D. Hartomo, "Yolo V4 for Advanced Traffic Sign Recognition with Synthetic Training Data Generated by Various GAN," *IEEE Access*, vol. 9, pp. 97228–97242, 2021, doi: 10.1109/ACCESS.2021.3094201.

[13] N. An and W. Q. Yan, "Multitarget Tracking Using Siamese Neural Networks," *ACM Transactions on Multimedia Computing, Communications and Applications*, vol. 17, no. 2s, Jun. 2021, doi: 10.1145/3441656.

[14] A. A. Micheal, K. Vani, S. Sanjeevi, and C. H. Lin, "Object Detection and Tracking with UAV Data Using Deep Learning," *Journal of the Indian Society of Remote Sensing*, vol. 49, no. 3, pp. 463–469, Mar. 2021, doi: 10.1007/s12524-020-01229-x.

[15] N. Marinello *et al.*, "TripletTrack: 3D Object Tracking using Triplet Embeddings and LSTM," *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 4499–4509, 2022, doi: 10.1109/CVPRW56347.2022.00496.

[16] B. Shuai, A. G. Berneshawi, D. Modolo, and J. Tighe, "Multi-Object Tracking with Siamese Track-RCNN," Apr. 2020, [Online]. Available: http://arxiv.org/abs/2004.07786

[17] N. S. Inthizami *et al.*, "Flood video segmentation on remotely sensed UAV using improved Efficient Neural Network," *ICT Express*, vol. 8, no. 3, pp. 347–351, Sep. 2022, doi: 10.1016/j.icte.2022.01.016.

[18] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal Speed and Accuracy of Object Detection," Apr. 2020, [Online]. Available: http://arxiv.org/abs/2004.10934

[19] N. Wojke, A. Bewley, and D. Paulus, "Simple online and realtime tracking with a deep association metric," in *2017 IEEE International Conference on Image Processing (ICIP)*, Beijing, China, 2017, pp. 3645–3649.

[20] A. Pujara and M. Bhamare, "DeepSORT: Real Time & Multi-Object Detection and

Tracking with YOLO and TensorFlow," in *Proceedings - International Conference on Augmented Intelligence and Sustainable Systems, ICAISS 2022*, Institute of Electrical and Electronics Engineers Inc., 2022, pp. 456–460. doi: 10.1109/ICAISS55157.2022.10011018.

[21] X. Hou, Y. Wang, and L.-P. Chau, "Vehicle Tracking Using Deep SORT with Low Confidence Track Filtering," *16th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, 2019.

[22] G. Desai, V. Ambre, S. Jakharia, and S. Sherkhane, "Smart Road Surveillance Using Image Processing," in *2018 International Conference on Smart City and Emerging Technology (ICSCET)*, Mumbai, India, 2018, pp. 1–5.

[23] T.-Y. Lin *et al.*, "Microsoft COCO: Common Objects in Context," in *In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds) Computer Vision – ECCV 2014. ECCV 2014. Lecture Notes in Computer Science*, vol. 8693, Springer, Cham., 2014.

[24] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA: IEEE, 2016, pp. 779–788. doi: doi: 10.1109/CVPR.2016.91.

[25] C. -Y. Wang, H. -Y. Mark Liao, Y. -H. Wu, P. -Y. Chen, J. -W. Hsieh, and I. -H. Yeh, "CSPNet: A New Backbone that can Enhance Learning Capability of CNN," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, Seattle, WA, USA, 2020, pp. 1571–1580.

[26] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," Apr. 2018, [Online]. Available: http://arxiv.org/abs/1804.02767

[27] C. Y. Wang, A. Bochkovskiy, and H. Y. M. Liao, "Scaled-yolov4: Scaling cross stage partial network.," in *In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Nashville, TN, USA: IEEE, Jun. 2021, pp. 13024–13033.

[28] K. B. Lee and H. S. Shin, "An Application of a Deep Learning Algorithm for Automatic Detection of Unexpected Accidents Under Bad CCTV Monitoring Conditions in Tunnels," in *2019 International Conference on Deep Learning and Machine Learning in Emerging Applications (Deep-ML)*, Istanbul, Turkey, 2019, pp. 7–11.

[29] M. Yazdi and T. Bouwmans, "New trends on moving object detection in video images captured by a moving camera: A survey," *Comput Sci Rev*, vol. 28, pp. 157–177, 2018.

[30] J. Zhang, J. Sun, J. Wang, and X. G. Yue, "Visual object tracking based on residual network and cascaded correlation filters," *J Ambient Intell Humaniz Comput*, vol. 12, no. 8, pp. 8427–8440, Aug. 2021, doi: 10.1007/s12652-020-02572-0.