Grenze International Journal of Engineering and Technology, June Issue

A Video Surveillance-based Enhanced Collision Prevention and Safety System

Bhumika Loungani[1], Janvi Agrawal[2] and Lija Jacob[3]
[1-2]Department of Data Science, Christ University, Pune, India
Email: bhumikaloungani06@gmail.com, janviagarwal40@gmail.com
[3]Head of Department of Data Science, Christ University, Pune, India
Email: lija.jacob@christuniversity.in

***Abstract*—Road traffic crashes that result in fatalities have become a global phenomenon. Therefore, it is imperative to use caution and vigilance while being on the road. Human mistake, going over the speed limit, being preoccupied while driving or walking, disobeying safety precautions, and other factors can also contribute to such unforeseen accidents or injuries, which can result in both bodily and material loss. So, safety is what we seek to achieve. Furthermore, as the number of automobiles has increased, so too have collisions between vehicles and pedestrians. Using computer vision and deep learning approaches, this research seeks to anticipate such encounters. The data often comes from traffic surveillance cameras in video formats. We have therefore concentrated on video sequences of vehicle-pedestrian collisions. We begin with a detection phase that includes the identification of vehicles and pedestrians; for this phase, we employed YOLO v3 (You Only Look Once). YOLO v3 has 80 classes, but we only took six of them: person, car, bike, motorcycle, bus, and truck. Following detection, the Euclidean distance approach is used to determine the interspace between the vehicle and the pedestrian. The closer the distance between a vehicle and a pedestrian, the more likely it is that they will collide. As a result, pedestrians in risk are located, and once we are aware of the pedestrians in danger, we search for nearby safer regions to alert them to head to the nearest location that is secure.**

***Index Terms*— vehicle detection, pedestrian detection, collision warning system, YOLOv3, non max suppression, enhanced vision-based system, video surveillance.**

I. INTRODUCTION

Increasing mobility and safety in transportation is the goal of an intelligent transportation system (ITS). Recently, ITS activities have been primarily focused on developing systems to detect and safeguard pedestrians. Pedestrian accidents frequently arise from complicated circumstances and are caused by a variety of factors including motorist and pedestrian behaviour as well as road infrastructure that is geometry of the road. Visibility in traffic and how well the traffic situation is perceived are key factors in safety. Road traffic accidents result in losses that include instant death as well as permanent disability or serious injuries, as well as loss of property. Millions of deaths or injuries occur every year around the world, which is magnified. According to statistics, almost 1.3 million people pass away in such crashes, and more than half of those fatalities include motorcyclists, pedestrians, or cyclists [1]. Additionally, they range in age from 5 to 29 years old, which is problematic for the economy. This explains why safety must be prioritized in order to notice the deteriorating situation in road

Grenze ID: 01.GIJET.9.2.548_1

fatalities and injuries and to take the necessary action.
The majority of traffic surveillance cameras are now operated manually, which requires time, money, and effort. Today, it's also simple to find security cameras on roads, such as near sidewalks, highways, or close stores, or at intersections. As a result, computer vision-based automated traffic accident detection will be more reliable [2]. Automatic detection is used in many areas, including security, laparoscopy, missile monitoring, and not just the prediction and prevention of accidents [3]. We need automatic systems since there is a huge need for video cameras on the roadways to evaluate and monitor the traffic moving through the city. There are many ways to automate this process and stop traffic accidents, including statistical techniques, machine learning, and deep learning. Therefore, many issues with video processing for the recognition and categorization of traffic accidents have been resolved by these techniques [4].
Background and foreground subtraction, frame differencing and the optical flow method, are frequently employed for object tracking. Additionally, algorithms like HOG (Histogram of Oriented Gradient), YOLO (You Only Look Once), Blob Boundary Box, R-CNN (Region-based CNN), etc. are used for detecting purposes. YOLO is distinct from traditional categorization methods and, in contrast to conventional classification algorithms, can classify numerous objects as it is a single shot detector. Because YOLO is built on convolutional neural networks, it takes a unique approach to object detection. As a result, YOLOv3 is used in this research to find the necessary moving objects to make collision avoidance systems. The goal of this research is to identify vehicles and pedestrians from videos, measure the distance between them, determine whether they meet a distance threshold, and forecast when danger will arise between them.
Section II, literature and related works, is the next section in this paper's format. Section III has Theoretical Background. The collected dataset, the proposed approach all described in depth in Section IV. Section V has the experimental findings. The future scope or discussion along with conclusion will be the subject of part section VI.

II. Related Works

The pre-processing and detection of vehicles, pedestrians, and articles combining both vehicle and pedestrian pre-processing and detection make up the different categories that makes up the literature review.

A. Vehicle Pre-Processing and Detection

Video samples in *.avi format was gathered [5]. Multiple frames make up a video. A variety of pre-processing techniques were used to apply to the video in order to extract the relevant features. According to Gunjan Goyal et al. [5] Grayscale formats have been transformed into their own image views using a pre-processing technique. To separate the video into frame formats, the frame differencing technique was created. The interest area was separated from the background using K-means clustering, which also helps to separate each frame of the movie. After segmentation, the Speed Up Robust Feature method was used to extract useful features. After that, the trustworthy feature set was chosen using the ABC optimization algorithm. A brand-new stochastic algorithm for global optimization was called An Artificial Bee Colony. In the PTZ (Pan, Tilt, and Zoom) dataset, the moving object is classified and detected using the DNN algorithm. When developing a system to prevent accidents between a vehicle and a pedestrian, speed estimation is crucial. By measuring speed, the separation between vehicles and pedestrians is determined, on which the likelihood of a collision is predicated.
In order to identify and track moving vehicles, Jency Rubia et al. [6] used speed detection from city surveillance videotapes. The process for extracting items from videos was unsupervised. On the basis of optical flow estimation, the system was developed. The background subtraction method is a well-known tool for identifying moving vehicles in a series of movies. Noise was eliminated using a median filter. Undesired objects were removed by thresholding techniques like morphological procedures. Using the location of the car in each frame, the speed of the vehicle was calculated. Accuracy was the parameter used to calculate performance. The model achieved 99% of Accuracy. Illumination is a problem that has been mentioned.
A technique to estimate the vehicle's speed utilising computer vision, digital video, and image processing was put out by Arash Gholami Rad et al. [7]. First, Background and Foreground Subtraction was performed using the mean filter approach. After that, noise and object dilatation were removed by converting the processed photos into binary format. The frame differencing method was used to calculate the speed of the vehicle, and Blobs Bounding Box was used to measure the rate and time between the two entry and exit regions. The model's performance score was more than 93%.

B. Pedestrian Pre-Processing and Detection

Person detection entails locating humans in a video sequence and separating people from other items. The technology performs tracking and detecting of humans. A video sequence camera that was utilised to follow every pedestrian was suggested by Tudor Barbu [8]. The set of videos was produced in grayscale. The frames were converted into binary format using a threshold value of T=30. The noise in the video frames was removed using partial differential equations. An approach based on the histogram of oriented graphics was used to perform video tracking. Precision, Recall, and F1 Score were used to evaluate the accuracy which are 90%, 85%, and 87% respectively.

The detecting speed of pedestrians has been improved, according to Chunli Wang et al. [9]. To improve the detection accuracy of small targets, a deconvolution layer was added to the framework. The Cityscapes dataset was used for testing, whereas the VOC and COCO datasets were used for training.

For feature extraction, the SSD method served as the foundation. Furthermore, relevant frames were extracted and classified by regression using six different layers.

To recognise pedestrians, shallow convolution layer was added. The learning rate for the model was 0.01. Generalization accuracy can be considerably impacted by the learning rate parameter [10]. They achieved accuracy of 73%.

There was proposed a rapid way for identifying pedestrians in tracking systems with limited memory and processing units by Sangjun Kim and colleagues [11]. A Random Forest classifier built on the teacher-student architecture was used to do model compression. First, a Student Random Forest was trained using a compressed model. Fast and precise pedestrian recognition was achieved by combining Image Scaling and Student Random Forest. Information Gain was calculated via entropy estimate. The minimization threshold was set at 0.39 and the maximum number of trees was 50. The dataset was split into two groups for teacher and student, A and B, respectively, for training. The Performance Evaluation of Tracking and Surveillance 2006, Town Centre, and Caltech benchmark dataset underwent two measurements. Results of the evaluation were Precision and Recall of 90% and 84% respectively.

C. Vehicle-Pedestrian Pre-Processing and Detection

Haarish Gopalan et al. [12] suggested a Deep Convolutional Neural Networks method to track and count the number of automobiles and pedestrians using the surveillance camera in the traffic lights. There were two sections to this algorithm. The categorization algorithm in the first section was implemented using the convolutional neural network-based on You Only Look Once technique. The proposed algorithm, which was used to follow the vehicles and gauge their performance, was the subject of the second portion.

Vehicle categorization and object detection have both been accomplished using convolutional neural network technology. Using object identification, monitoring, tracking, and classification techniques, CNN extracts feature from each frame of video. CNN displays object detection as boxes enclosing the item's frame [13]. Software engineers and academics utilise computational time as a crucial evaluation metric to ascertain whether an algorithm can operate efficiently. Each step of the detection pipeline is looked at and optimised using an architecture that offers working accuracy comparable to state-of-the-art methods and uses less processing time. The testing dataset used the NVIDIA Jetson TK1 platform. The author suggested a decision forest detector as a method to produce the current, well-known Caltech USA dataset. The regularisation technique known as "dropout" was particularly successful in reducing overfitting. Using organised taxonomies for counting cars and pedestrians, a new algorithm for tracking vehicles and pedestrians from video sequences has been introduced. This taxonomy can analyse a binary threshold by employing contours or blob detection, or it can divide a scene into its foreground and background. Its advantages include precise item detection estimation and a well-evolved and modified taxonomy that functions for both the background and the foreground.

Machine learning, Open CV, and Python are used to illustrate several car and pedestrian recognition approaches by Aayush et al. [14]. Before using a detection technique, the frames were converted to black and white. For feature extraction, histograms of the Directed method were employed. The MIT pedestrian, INRIA, and Caltech datasets were used to train three different detector models, namely the Deformable Part Model, Deep Network, and Decision Forest. Their main problem was insufficient lighting, which produced unreliable results.

III. BACKGROUND

Yolov3, which was initially put forth by [15], [16], has developed to include a number of cutting-edge features, opening the way for more precise and accurate state-of-the-art Object detection methods. The YOLO framework now has multiple versions (v4, v5, v6, v7, and v8) with added layers and are known for their lightweight

architecture, increased detection speed, and increased mean average precision (mAP). Darknet-53 is used as the foundation in the YOLO v3. Darknet-53, a deep neural network, which is written in C and implemented using CUDA which has convolutional layers for extracting features from images, are utilized for feature extraction. In order to extract feature maps from the input images, it builds a Feature Pyramid Network (FPN) [17].
As shown in Figure 1, the Darknet-53 has 53 layers of convolutional neural networks, proceeded by 3*3 and 1*1 fully connected convolutional layers for object detection. An ensemble of 106 convolutional layers is created in YOLOv3 by adding an extra 53 layers, each of these layers is followed by the Leaky ReLu activation function and batch normalization. As illustrated in Figure 2, the three scales used for YOLO v3's prediction are exactly generated by reducing the input image's strides or dimensions by 32, 16, and 8 accordingly. Large object detection is performed at the 82nd layer with stride 32, medium object detection at the 94th layer with stride 16, and small object detection at the 106th layer with stride 8. The output size is 13x13 at the 82nd layer. The output size is 26x26 at the 94th layer. The output size is 52x52 at the 106th layer.
The architecture offers up-sampling connections and residual overhead. The final output of the YOLO network, which is entirely convolutional, is created by adding a 1 x 1 kernel to a feature map. Applying 1 x 1 detection kernels or filters to three-size feature maps at three separate points in the network is how YOLO v3 detects objects. The detection kernel is 1 x 1 x (B x (5 + C)), where, 5 denotes the number of attributes of bounding box, including an object's confidence score, B denotes the number of bounding boxes which cell predicts on feature map, and C denotes the number of classes. For YOLOV3 trained on the COCO dataset, B = 3, and C = 80, the kernel size is 1 x 1 x 255. A layer, also known as a network stride, is the ratio by which the input is down-sampled. The input file is 416 by 416 pixels in size. Before we get into how detection works at three levels, let's go through four modifications in YOLOv3. Residual Box and Skip Connections, Grid Cells, Bounding Box Regression, and IOU are major of them.

	Type	Filters	Size	Output
	Convolutional	32	3 × 3	256 × 256
	Convolutional	64	3 × 3 / 2	128 × 128
1×	Convolutional	32	1 × 1	
	Convolutional	64	3 × 3	
	Residual			128 × 128
	Convolutional	128	3 × 3 / 2	64 × 64
2×	Convolutional	64	1 × 1	
	Convolutional	128	3 × 3	
	Residual			64 × 64
	Convolutional	256	3 × 3 / 2	32 × 32
8×	Convolutional	128	1 × 1	
	Convolutional	256	3 × 3	
	Residual			32 × 32
	Convolutional	512	3 × 3 / 2	16 × 16
8×	Convolutional	256	1 × 1	
	Convolutional	512	3 × 3	
	Residual			16 × 16
	Convolutional	1024	3 × 3 / 2	8 × 8
4×	Convolutional	512	1 × 1	
	Convolutional	1024	3 × 3	
	Residual			8 × 8
	Avgpool		Global	
	Connected		1000	
	Softmax			

Figure 1: Darknet-53 Architecture

A. Residual Box and Skip Connections

Darknet-53 employs a new type of block known as Residual Block. Deep neural networks are challenging when it comes to training. As the depth increases, network accuracy can get drenched, resulting in larger training error. To address this issue, the residual block was devised. The insertion of a skip connection distinguishes the conventional convolution block from the residual block in terms of architecture. The skip connection transports the input to deeper layers.

B. Grid Cells

Grid cells are just detection cells. A frame of a video is divided into S x S grids, and if an object's CenterPoint lands on a certain grid cell, that grid cell is responsible for recognising it using its three or more anchor boxes. The size of the grid cell is determined solely by the neural network's down sampling procedures e.g., max pooling. The five down sampling stages in Darknet-53 divides the resolution, making the grid cells 32 pixels at the minimum or smallest resolution feature map, and requiring an input image with a resolution that is a multiple of 32. The network can be kept small and quick by pre-processing images that don't have resolutions that are multiples of 32 pixels by cropping, stretching, or padding them to fulfil this requirement.

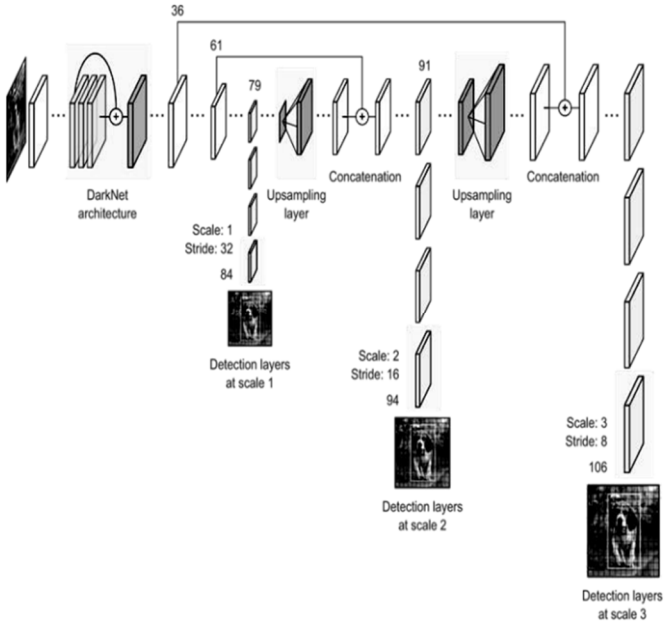

Figure 2: YOLOv3 Architecture

C. Bounding Box Regression

Rectangle-shaped bounding boxes that represent all the things in an image. As many bounding boxes as, there are objects in a given image is possible. Anchor Boxes are utilized by YOLOv3 to predict bounding boxes. These anchor boxes are also known as priors used to determine the actual width and height of anticipated bounding boxes. There are a total of nine anchor boxes utilized, three for each scale, with the first scale using the three largest anchors, the second scale using the next three, and the third scale using the final three. This indicates that every grid scale of the feature map at each output layer may forecast three bounding boxes using three anchor boxes. In YOLOv3, K-Means Clustering is used to calculate these anchors. The properties of these bounding boxes are determined by YOLO using a single regression module, where Y is the final vector representation of each bounding box.

$$Y = [pc, bx, by, bh, bw, c] \quad (1)$$

Where, pc=probability score of the grid that contains an object
bx, by=bounding box centre coordinates
bh, bw=height and width of the bounding box
c=corresponding class of the object
Figure 3 depicts the modifications of anchors' network output attributes. tx, ty, tw, th are the network outputs. cx and cy are the top-left coordinates of the grid. pw and ph are anchor dimensions for the grid box. Therefore, YOLOv3 runs outputs (tx, ty) through a sigmoid function to estimate the centre coordinates of bounding boxes (bx, by). In light of the aforementioned equations also represented in Figure 3, we can obtain the centre coordinates as well as the width and height of the bounding boxes. Additionally, NMS is used to suppress all of the redundant bounding boxes from 10,847 boxes.

D. Intersection Over Union (IOU)

The IOU's (a number between 0 and 1) purpose is to maintain just the appropriate bounding boxes from grid boxes that have several bounding boxes for a single object that are overlapping in an image. The function known as Intersection over Union is used in Non-Maximum Suppression. The IOU selection threshold is set to be 0.5. The IOU is then calculated for each grid cell using YOLO. Finally, it considers grid cells with an IOU > threshold rather than those predicted to have an IOU less than or equal to threshold. The IOU formula is given by:

$$IOU=\text{Area of Intersection}/\text{Area of Union} \quad (2)$$

Three levels of detection are performed, starting with the first detection at layer 82. The network samples the first 81 layers of the picture down. After the first detection at the 82nd layer with the 1 x 1 kernel on the 416 x 416 input picture, the generated feature map is of size or dimensions 13 x 13 x 255. Following a few

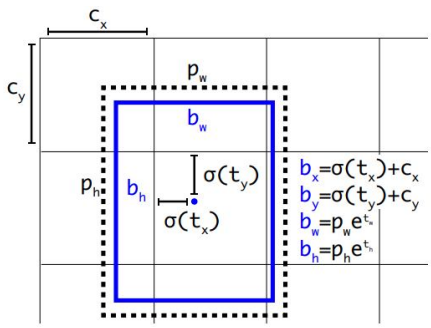

Figure 3: Anchor Box Attributes

convolutional layers, the feature map of layer 79 is then sampled up in the dimensions of 26 x 26 x 26. Then, a feature map from layer 61 is merged with this feature map. The features of the previous layer are then recombined by subjecting the combined feature map to a few 1 x 1 convolutional layers (61). Then, at layer 94, a secondary detection occurs, producing a detection feature map that measures 26 by 26 by 255. A layer 36 feature map is similarly depth concatenated with a layer 91 feature map after being subjected to a few convolutional layers of processing. Similar to before, a few 1 x 1 convolutional layers are used to merge the data from the previous layer (36). The final prediction takes place on the 106th layer, giving the feature map a size of 52 by 52 by 255.

IV. Proposed Framework

Collision videos that have been compiled from YouTube are included in the dataset. The first stage in the suggested process is to look for pedestrian-vehicle collisions. Out of the 80 classes offered by YOLOv3, we only took six of the classes that belong to vehicles. Following detection, we measured the proximity between the pedestrian and the vehicle. The centroid coordinates of the recognised objects are employed in the Euclidean distance technique, which is used to calculate distance. Based on videos, a distance threshold is established. If the distance threshold is lower, both objects are said to be approaching one another, and vice versa. Similar to how we can measure distance, we can also compute speed with the same formula. But in order to speed up processing, we merely considered distance here. A boundary box with danger is visible if objects breach the distance threshold, alerting pedestrians to their risk. Once pedestrians in danger have been located, the closest, safest location to the at-risk pedestrian is determined. If there are safe items nearby, such as potted plants on roads, fire hydrants, stop signs, benches on footpaths, or the footpath itself, the safest place around the pedestrian who is in danger can be computed with knowing the nearby safe objects. In order to warn pedestrians to move to that specific location if any vehicle is approaching at speed and the pedestrian is considered to be in danger, the distance from the pedestrian who is in danger and the nearest safest item are identified. The system architecture of our proposed framework for collision detection is shown in Figure 4.

A. Detection and Tracking

The input image used in our implementation is 416x416x3. Then, this image is split into 13x13 cells with a stride of 32x32 in each cell. From each box or grid cell, 9 bounding boxes are extracted. As a result, the scoring uses a value between 0 and 1. It shows the likelihood that an interesting object is present within the bounding box. The six classes are represented by the C=6x1 vector,
Calculate the Euclidean distance for each object to determine the gap between the current frame and the referenced frame. The formula for Euclidean distance is given below in equation 3.

$$d(p,q) = d(q,p) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \ldots + (q_n - p_n)^2} = \sqrt{\sum (q_i - p_i)^2} \quad (3)$$

which stands for person, car, bicycle, motorbike, bus, and truck. As a result, the vector's size is equal to 9*(5+6) = 99 according to the formula we discussed previously for Bx(5+C). The result will be 13x13x99=16731 for each image. Table 1 summarizes the parameters we used for detection.

B. Euclidean Distance

The Euclidean distance measures the separation between two vectors with real values. The square root of the sum of the squared differences between the two vectors is used to determine Euclidean distance. The standard

straight-line distance between two points in Euclidean space is known as the Euclidean distance or Euclidean metric.

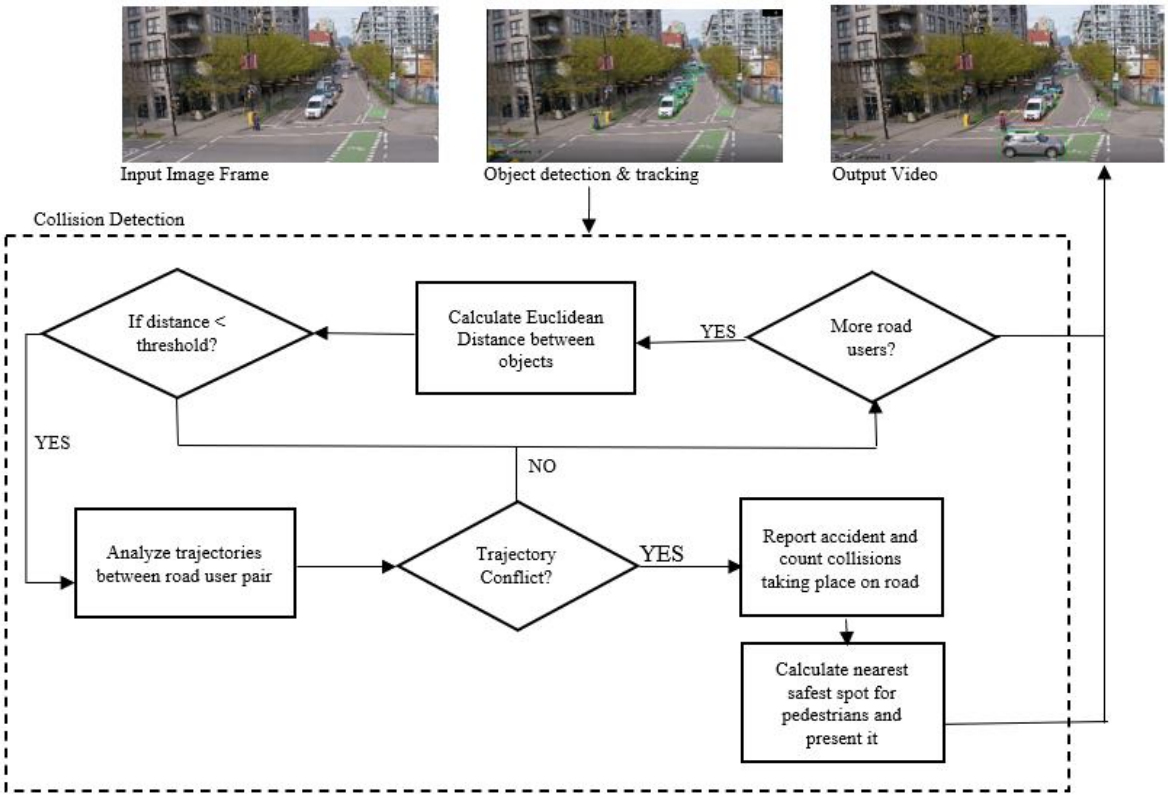

Figure 4: System Architecture of Proposed Framework

TABLE I: YOLOV3 EXECUTION PARAMETERS

Parameters	Values
Input Image Size	416x416x3
Number of cells per image	13x13
Number of bounding boxes per cell	9
Classes	6
Classification threshold	0.6
Non max suppression overlapping threshold	0.4

Figure 5 illustrates the intended or enhanced model's flow, which includes the actions taken during the system's development. A video is initially taken as input and split up into frames. YOLOv3 is used for further object recognition and tracking, and distances are calculated using Euclidean distance, alerts, and instructions such as Move toward this place or in this direction are given.

V. RESULTS

The dataset is video surveillance and a video was created for accurate testing of the algorithm. On a variety of video footage featuring crashes between a car and a pedestrian, the algorithm is evaluated. The classification scoring threshold is set at 0.6, and the non-maximum suppression for overlapping detection boxes is set at 0.4. The result displays the detection with boundary boxes in red and green. Given the distance between the pedestrian and the vehicle, the pedestrian is shown to be safe by the green bounding box over the object, while

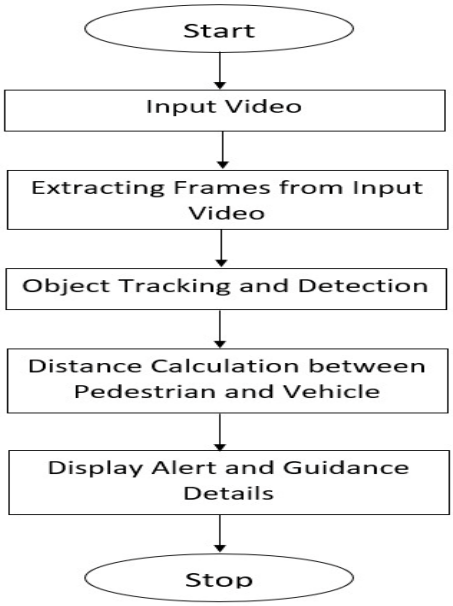

Figure 5: Steps involved in the proposed architecture

the objects that are in danger are shown by the red bounding box. Safer areas are found nearby and are indicated by a white bounding box once the danger pedestrians have been detected. A message is printed for the pedestrian who is instructed to go to the designated direction, zone, or spot. As was previously discussed, we must consider the immediate surroundings of the pedestrians who are at risk if we have to find safer areas. Either there are no objects to be found or the only objects we can find are on the side of the road, where there aren't likely to be any automobiles. Potted plants along the edges of footpaths, the footpath itself, and other similar objects are examples of safer objects that are included in our final video and whose frames are represented in the Figure 6 (A) and (B). All the safer objects will be detected but only with nearest safest object or zone is highlighted with boundary boxes. The number of collisions is also visible in the video frame. Moreover, the number of objects is also counted as per their classes for better understanding.

Figure 6 (A): Represents the nearest safest zone in white boundary box when the collision is detected which is shown with red bounding box

Figure 6 (B): Represents the nearest safest zone in white boundary box when the collision is detected which is shown with red bounding box

VI. Conclusion

In this work, we demonstrated the bounding box concept-based detection of cars and pedestrians from a video. We picked the YOLO approach because it is widely used and is quick at detecting things, which are crucial components of surveillance systems. Object detection efficiency has been a big difficulty in surveillance systems even today. The rising death rates necessitate the use of alert systems. These safety enhancing devices might be employed with sensors and computer vision techniques to alert pedestrians when a car and related vehicles is identified in its hitting zone, preventing such unanticipated collisions. Stereo vision videos, which are 3d simulations, can also be utilized for the same purpose so that we can measure the distance between objects and more accurately assess the scenario. We have also seen the YOLOv3 architecture. In order to construct detection

bounding boxes in real time, localization coordinates of objects on retrieved video frames were used. The experimental findings in this study demonstrate the concepts underlying the object detecting method.

REFERENCES

[1] G. Khorasani, A. Tatari, A. Yadollahi and M. Rahimi, "Evaluation of intelligent transport system in road safety," Int. J. Chem. Environ. Biol. Sci.(IJCEBS), vol. 1, p. 110–118, 2013.

[2] H. Ghahremannezhad, H. Shi and C. Liu, "Real-Time Accident Detection in Traffic Surveillance Using Deep Learning," in 2022 IEEE International Conference on Imaging Systems and Techniques (IST), 2022.

[3] K. Ravikanth, M. Swetha, K. Swathi and T. A. Roseleena, "Video Surveillance System for Pedestrian and Vehicle Detection Using Convolutional Neural Networks," vol, vol. 4, p. 25–27.

[4] S. Robles-Serrano, G. Sanchez-Torres and J. Branch-Bedoya, "Automatic detection of traffic accidents from video using deep learning techniques," Computers, vol. 10, p. 148, 2021.

[5] G. Goyal and T. Gupta, "Moving Object Detection in Video Streaming Using Improved DNN Algorithm," in 2020 International Conference on Smart Electronics and Communication (ICOSEC), 2020.

[6] A. T. Al-Heety and others, "Moving vehicle detection from video sequences for traffic surveillance system," ITEGAM-JETIA, vol. 7, p. 41–48, 2021.

[7] M. R. Karim and A. Dehghani, "Vehicle speed detection in video image sequences using CVS method," International journal of the Physical Sciences, vol. 5, p. 2555–2563, 2010.

[8] T. Barbu, "Pedestrian detection and tracking using temporal differencing and HOG features," Computers & Electrical Engineering, vol. 40, p. 1072–1079, 2014.

[9] C. Wang and J. Bai, "Pedestrian detection method in vehicle video based on AMSSD," in 2019 IEEE International Conference on Signal Processing, Communications and Computing (ICSPCC), 2019.

[10] D. R. Wilson and T. R. Martinez, "The need for small learning rates on large problems," in IJCNN'01. International Joint Conference on Neural Networks. Proceedings (Cat. No. 01CH37222), 2001.

[11] S. Kim, S. Kwak and B. C. Ko, "Fast pedestrian detection in surveillance video based on soft target training of shallow random forest," IEEE Access, vol. 7, p. 12415–12426, 2019.

[12] P. H. Gopalan, K. P. Kumar, K. V. Kumar and A. Sanampudi, "Vehicle and Pedestrian Video-Tracking with Classification Based on Deep Convolutional Neural Network," strategies, vol. 8, 2020.

[13] P. Varsha, D. K. Priyadharshini, S. Swetha and others, "Video analysis of vehicle and pedestrian using neural network," Annals of the Romanian Society for Cell Biology, p. 4727–4733, 2021.

[14] A. P. Singh and M. Gupta, "Video based vehicle and pedestrian detection," Annals of the Romanian Society for Cell Biology, vol. 25, p. 14653–14658, 2021.

[15] J. Redmon, S. Divvala, R. Girshick and A. Farhadi, "You only look once: Unified, real-time object detection," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016.

[16] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," arXiv preprint arXiv:1804.02767, 2018.

[17] K. J. Oguine, O. C. Oguine and H. I. Bisallah, "YOLO v3: Visual and Real-Time Object Detection Model for Smart Surveillance Systems (3s)," arXiv preprint arXiv:2209.12447, 2022.