

# A Pedestrian Detection and Tracking Framework for Autonomous Cars: Efficient Fusion of Camera and LiDAR Data

Muhammad Mobaidul Islam, Abdullah Al Redwan Newaz, and Ali Karimoddini\*

**Abstract**—This paper presents a novel method for pedestrian detection and tracking by fusing camera and LiDAR sensor data. To deal with the challenges associated with the autonomous driving scenarios, an integrated tracking and detection framework is proposed. The detection phase is performed by converting LiDAR streams to computationally tractable depth images, and then, a deep neural network is developed to identify pedestrian candidates both in RGB and depth images. To provide accurate information, the detection phase is further enhanced by fusing multi-modal sensor information using the Kalman filter. The tracking phase is a combination of the Kalman filter prediction and an optical flow algorithm to track multiple pedestrians in a scene. We evaluate our framework on a real public driving dataset. Experimental results demonstrate that the proposed method achieves significant performance improvement over a baseline method that solely uses image-based pedestrian detection.

## I. INTRODUCTION

Real-time accurate pedestrian detection and tracking are crucial to ensure safety and reliability in autonomous driving [1]. This often requires the estimation of multiple pedestrian trajectories from multi-modal sensor data. There are, however, several challenges that make pedestrian detection and tracking a notoriously hard task. For instance, pedestrians may appear in a scene with different articulations of body parts, partial occlusion, and appearance in various poses. Furthermore, the detection and tracking may fail due to the sensitivity of some sensors under low illumination or bright sunlight conditions. Despite all these challenges, it is vitally important for autonomous driving to achieve accurate and robust pedestrian detection and tracking under various conditions in real-time to be able to transition from the innovation space to real-world operations [2].

Pedestrian tracking consists of two main stages: pedestrian detection and association of the detected pedestrians with current and past estimations. Generally, there are two streams of research directions for pedestrian tracking —either the use of deep learning models in an end-to-end fashion or the development of a machine learning pipeline with a mix of deep learning and classical methods, to enable pedestrian tracking. Some of the notable end-to-end deep learning-based pedestrian tracking methods include TrackR-CNN [3], Tracktor++ [4] and JDE [5]. These methods are created

M. M. Islam, A. A. Redwan Newaz, and A. Karimoddini are with the Department of Electrical and Computer Engineering, is with the Department of Industrial and Systems Engineering, North Carolina Agricultural and Technical State University, Greensboro, NC 27411 USA.

\*Corresponding author: A. Karimoddini. Address: 1601 East Market Street, Department of Electrical and Computer Engineering North Carolina A&T State University Greensboro, NC, US 27411. Email: akarimod@ncat.edu (Tel: +13362853313).

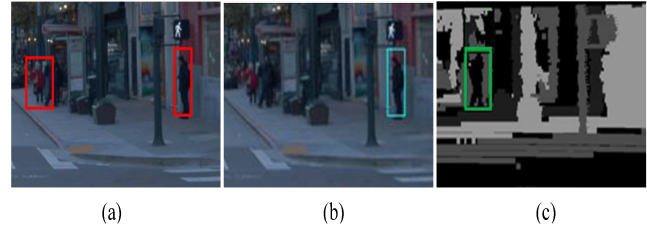


Fig. 1: Fusion of multi-modal sensor data can improve pedestrian detection and tracking accuracy: (a) ground-truth bounding boxes of two anticipated pedestrians, (b) detector can detect only one pedestrian on camera image, (c) and another pedestrian on corresponding depth image from LiDAR scan. Therefore, the best result can be achieved by combining these independent results from (b) and (c) into a joint prediction

by augmenting existing pedestrian/object detectors, e.g., by adding additional recurrent layers to the detection head in order to incorporate the temporal context of the scenes. There are few end-to-end multimodal sensor fusion mechanisms that use 3D LiDAR and camera data for pedestrian tracking [6], [7]. These methods commonly suffer from low runtime efficiency because of the complex network architecture. On the contrary, machine learning pipeline-based methods combine a pedestrian detector with the classical filtering technique to achieve better runtime efficiency [5], [8], [9]. These methods mostly rely on a specific sensor modality such as camera data. Nonetheless, these methods suffer from either rich feature descriptors or sensitivity to environmental conditions (e.g., illumination variations in case of using a camera). Hence, these methods are not robust enough for detecting and tracking pedestrians.

In this work, we hypothesize that a fusion between camera and LiDAR can enhance the robustness and accuracy of pedestrian detection and tracking. Fig. 1 shows a realistic event where it is anticipated from a pedestrian detector to detect at least two pedestrians. As it is obvious from Fig. 1.b, the detector can detect only one pedestrian on the camera image. On the other hand, as shown in Fig. 1.c, another pedestrian can be detected on the corresponding depth image from the LiDAR scan. Therefore, in this paper, we propose a fusion framework that combines LiDAR streams and camera data as well as the estimation of vehicle motion using the camera-based optical flow method. To achieve computationally tractable and real-time pedestrian tracking, first, LiDAR 3D streams are converted to 2D depth images and then are fed to a pedestrian detector by vertically concatenating these frames with camera images. Finally, the Kalman filter is used for fusing the prediction over the concatenated images as

well as the locations of previously detected pedestrians in the current frame.

In summary, the contributions of this paper are as follows: 1) we develop a method for real-time sensor fusion of camera and LiDAR data, 2) we develop a real-time accurate pedestrian detection and tracking framework, 3) we also integrate optical flow information into the developed tracking technique for achieving accurate predictions of multiple pedestrians over the RGB and depth images.

## II. RELATED WORK

In recent years, numerous approaches for detecting and tracking pedestrians in sequential images have grown steadily. With the recent advancement in deep learning, we are able to utilize machine learning models to accurately detect and classify pedestrians in complex scenarios. In this section, we will begin with a brief overview of learning-based pedestrian detection, then some existing fusion techniques for combining multimodal sensor data, and finally a brief overview of pedestrian tracking.

Recent works focus on improving the robustness and accuracy using deep neural networks [10]–[12]. Though these methods exhibit satisfactory performance in well-lit environments, they struggle to detect pedestrians in low light conditions such as nighttime, dawn, sunrises, and sunsets. This is because it is hard to generate shape information from images in ill-lit environments. On the other hand, the LiDAR can provide comparatively better shape features under these scenarios. As LiDAR can provide the only geometric feature of pedestrians, inferring context-aware relations of pedestrians' body parts is one way to distinguish among multiple pedestrians in a complex scenario [13]. To improve the classification performances of pedestrians, hand-crafted features such as slice feature and distribution of reflection intensities are explored [14]. The slice provides human body information based on body height and width ratio. Some works are focused on the density enhancement method for improving the sparse point cloud of LiDAR and they provide an improved shape feature for long-distance pedestrian detection [15]. Pioneering work on the conversion of a 3D point cloud of LiDAR into the 2D plane extracts both hand-crafted features and learned features, and then trains a support vector machine (SVM) classifier to detect pedestrians [16]. Later, 3D point clouds are converted into 2D panoramic depth maps and these depth maps are used in pedestrian detection [17]. Even though the LiDAR provides better results in the nighttime while it is difficult to get shape features using the camera or compare to a distorted image frame, camera-based methods perform better for long-distance pedestrians in the daytime where they appear in small sizes. The best result can be achieved by fusing both of these sensors to jointly predict pedestrians.

Since using the LiDAR or the camera independently unveils their limitations, it becomes interesting research direction to fuse different sensor modalities. In this setting, the improvement can be achieved from the use of multiple views of the pedestrian by learning a strong classifier that

accommodates both different 3D points of view and multiple flexible articulations. In order to integrate multiple sensor modalities several fusion mechanisms are investigated [18], [19]. These sensor fusion techniques mostly focus on either combining feature information from different sensors or generating candidate regions from one sensor and map these candidate regions to other sensor information. For instance, a deformable part detector is trained using optical images and depth images generated from 3D point clouds using upsampling technique [20]. Some fusion techniques cluster the LiDAR point cloud to generate candidate regions and map these regions on an image frame for detecting pedestrians [18], [21]. Most of these methods sacrifice runtime performance while improving detection accuracy. Therefore, a balanced fusion mechanism is needed to deal with the trade-off between accuracy and speed.

Recent pedestrian trackers are designed mostly based on end-to-end deep learning networks. A common approach is adding recurrent layers with the detector module. For example, the ROLO [22] is the combination of the convolutional layers of YOLO and the recurrent unit of LSTM. TrackR-CNN [3] is considered as a baseline method of multi-object tracking that adds instance segmentation along with multi-object tracking. Tractor++ is an efficient multiple object tracking that utilizes the bounding box regression on predicting the position of an object in the next frame where there is no train or optimization on tracking data [4]. On the other hand, the machine learning pipeline-based methods such as the Deep SORT which integrates appearance information along with Simple Online and Realtime Tracking (SORT) technique, adopts a single hypothesis tracking methodology with the recursive Kalman filter and the frame-by-frame data association. This technique focuses on an offline pre-training stage where the model learns a deep association metric on a large-scale person re-identification dataset [9]. A single-stage efficient multi-object tracking is introduced in [5], where target detection and appearance embedding are to be learned in a shared model and a Kalman filter is used for predicting the locations of previously detected objects in the current frame. While considering the LiDAR data for pedestrian tracking, a stochastic optimization method is introduced in [23] that merges the clustering and assignment task in a single stage.

Inspired by these works, we use both LiDAR and camera sensors to complement individual sensor limitations on detection and tracking performances. Thus, our solution can be applied to a wide variety of complex scenarios.

## III. PRELIMINARIES

Pedestrian tracking is a challenging problem as pedestrians need to be firstly detected in the current frame and then associated frames. The success in deriving a good tracker is mainly governed by a superior detector. In this work, we use YOLOv5 [24] as a base module of the pedestrian detector. Therefore, we will briefly explain the working principle of YOLOv5 as a pedestrian detector. Since the YOLOv5 architecture is the same as the YOLOv4 [25] except the

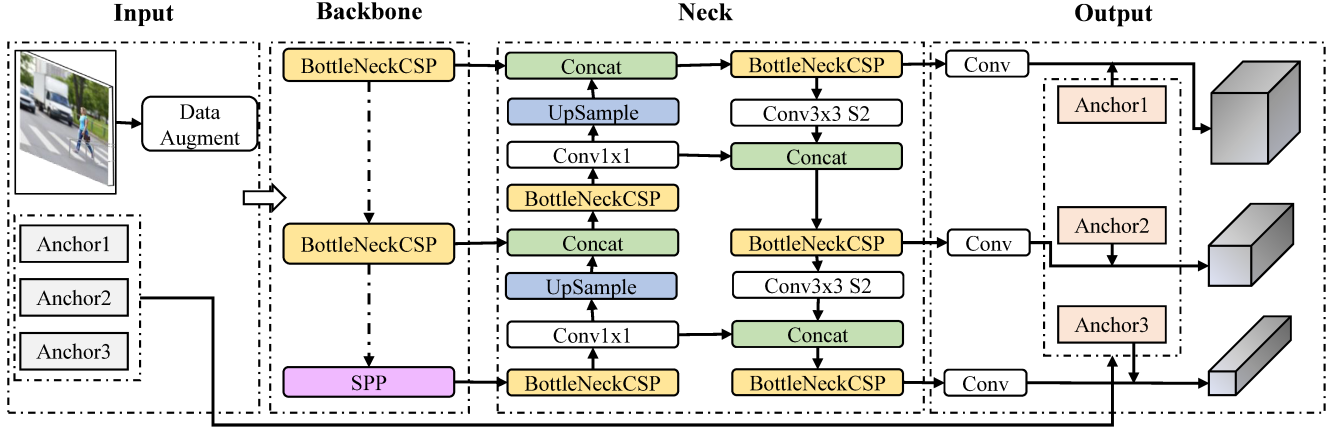


Fig. 2: **Architecture of YOLOv5:** YOLOv5 has four major sections. It starts with input then merges to the backbone for feature extraction. Feature information is enhanced in the neck section, and the output section produces a bounding box with confidence scores for a specific class of objects

training procedure, we will describe the architecture of the YOLOv4 and then will highlight the training differences.

Fig. 2 illustrates the architecture of YOLOv4 which can be segmented into four major parts: input, backbone, neck, and output. In the input section, the network takes an image and completes a data augmentation procedure that uses a data loader for scaling, color space adjustments, and mosaic augmentation. Among these augmentation techniques, mosaic augmentation firstly introduces in YOLOv4. The mosaic augmentation combines four training images into one in certain ratios to simulate four random crops which help to detect small-scale and partially occluded pedestrians.

After data augmentation, the augmented image is feed into the backbone of the network. In the backbone section, a BottleNeckCSP is used which is a modification of DenseNet [26]. Using BottleNeckCSP different shallow features like edges, colors, etc., are extracted. During training, the backbone module learns these features. Besides, an additional Spatial Pyramid Pooling (SPP) block is used to increase the receptive field and separate out the most important features from the feature maps of the BottleNeckCSP. The next part of the network is the neck part where the network enhances the understanding and extraction of the shallow features adopted in the backbone part. To do that a Path Aggregation Network (PANet) is used that includes a bottom-up augmentation path in conjunction with the top-down path used in Feature Pyramid Network (FPN). The PANet processes combine and analyze the extracted features and finally optimizes based on the target of the model. The last part of the network is output where the model yields the detection results using dense predictions. Dense predictions provide a vector by combining predicted bounding boxes and confidence scores for the classified pedestrians.

During the training process of YOLOv5, the floating-point precision is set to 16 bit instead of the 32-bit precision used in YOLOv4. Therefore, YOLOv5 exhibits higher performances than Yolov4 under certain circumstances.

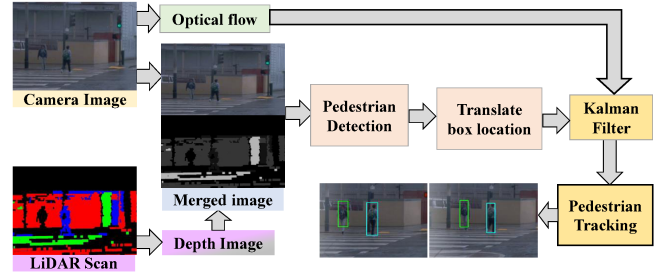


Fig. 3: **The proposed pedestrian detection and tracking framework:** A camera image and its synchronized LiDAR scan are the input of this framework. First, the LiDAR scan is converted into a depth image and then merge with the camera image. Next, merged fed into pedestrian detection module and outputs are mapped into the single-camera image frame by translating the detected boxes locations. Finally, for pedestrian tracking, a Kalman filter is implemented in which inputs are these translated bounding boxes and optical flow of input image.

#### IV. PROPOSED METHODOLOGY

We propose Fused-YOLO which is an integrated framework of multi-modal sensor information to track pedestrians at a real-time speed.

##### A. Conversion from LiDAR Scans to Depth Images

Camera-based pedestrian detection systems suffer from either low illumination or over-exposed images. It is then a better idea to complement the system with a LiDAR which acts as the primary depth sensor due to its high accuracy and long sensing range. LiDAR scan produces sparse point clouds, albeit this representation of data is rather challenging to incorporate as an input to neural networks. Instead, depth images are better correspondents of point clouds that are easy to manipulate constructively. Therefore, we convert a 3D LiDAR scan to the depth image in 2D image space.

Formally, the LiDAR stream consists of a sequence of registered 3D scans  $\{S_1, S_2, \dots, S_t\}$  arriving at time points  $t_1, t_2, \dots, t_t$ . Each scan  $S_t$  is a point cloud, i.e., a set of 3D points,  $S_t = \{\mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_i\}$  and  $\mathbf{p}_i := \{x, y, z\}$  represents

the Euclidean coordinate. Due to the huge amount of memories that are required over time, it is inefficient to work upon the raw point clouds. Instead, we can convert the 3D scan to a 2D depth map. A depth image can be thought of as a 2D grid map comprised of  $u_n$  cells. To generate the depth image we need to compute the distance of the scan objects from a viewpoint in such a way that maps  $\mathbf{p}_i$  to  $u$ . Then, we transform each point in point clouds from the Euclidean coordinate  $(x, y, z)$  to Spherical coordinate  $(\theta, \phi, r)$ . This way we can map each point to the corresponding grid cell such that  $u : \{\theta, \phi\} \rightarrow r$ . The pixel values of depth images lie in either gray or RGB color spaces. For the grayscale image, we normalize each cell value in the grid map to  $0 \rightarrow 255$  to the known maximum depth value and thus the intensity of the gray image represents the depth information. On the other hand, For the RGB scale image, we assign a distinct color from the RGB space to each cell value in the grid map based on the  $r$  parameter.

### B. Fusion with Depth Image

Our solution utilizes the Kalman filter along with parallel processing of RGB and depth images. To predict in a joint space, first, we project the LiDAR scan as a depth image to the RGB camera space. Let  $x_r$  and  $x_d$  be the RGB and depth images, respectively. Since (in common settings) the positions of camera and LiDAR are fixed but the resolution of camera image and the depth image from the LiDAR scan varies in size, we can project the depth image to the RGB camera space with either zero padding to the smaller image or cropping each of them into a same size. We denote this synchronized depth image by  $x_s$ . Second, we vertically concatenate the RGB image  $x_r$  and the depth image  $x_f$  by cropping each of them into a fixed size such that  $\mathbf{x} = \{x_s, x_r\}$ . Although it is possible to use an image classification network to predict pedestrians directly over  $x_s$ , it requires multiple calls for the joint prediction, i.e., the  $x_r$  and the  $x_s$  need to be fed to the pedestrian detection and pedestrian classification models, respectively. Thus, concatenating the  $x_r$  and the  $x_s$  into  $\mathbf{x}$  reduces the number of calls to different models and significantly improves the runtime efficiency. Finally, we feed this concatenated image  $\mathbf{x}$  to the pedestrian detector  $f$  to obtain bounding boxes and scores over fused images such that  $\hat{y} = f(\mathbf{x})$ . From the prediction  $\hat{y}$ , we can also separate individual predictions the  $\hat{y}_r$  and the  $\hat{y}_s$  for the  $x_r$  and the  $x_s$  correspondents, respectively. Since we vertically concatenate the  $x_s$  with the  $x_r$ , we can calculate an offset  $o$  based on the height of  $x_r$ . Then, we translate each bounding box  $b_s \in \hat{y}_s$  to the down by  $o$ .

Overlaying  $\hat{y}_s$  to  $\hat{y}_r$  may raise three distinct types of scenarios. Firstly,  $\hat{y}_s$  reduces the miss detection by accurately detecting pedestrians. Secondly,  $\hat{y}_s$  provides redundant inference with respect to  $\hat{y}_r$ . Finally,  $\hat{y}_s$  does not improve detection accuracy since it cannot detect any pedestrians. To overcome these scenarios, we utilize a Kalman filter to evaluate the joint predictions in a systematic manner. In our next subsection, we will provide a description of the proposed Kalman filter in detail.

### C. Integrated Framework for Pedestrian Tracking

The Kalman filter has been extensively applied in pedestrian tracking from the camera stream. Our framework uses such a technique to predict and update the pedestrian trajectories from the continuous camera and LiDAR streams. Our integrated framework augments the capability of the existing pedestrian tracking method by fusing depth information. To track multiple pedestrians in a frame, our framework uses three important information, i.e., bounding boxes from the RGB images, optical flow between consecutive RGB image frames, and bounding boxes from the depth images.

One of the important properties of the Kalman filter is that the state vector is a hidden parameter and the observation provides useful information to update the state vector. Therefore, in our setting while using the Kalman filter, the observations, i.e., bounding boxes, from the detector are not directly use for tracking of pedestrians. Basically, the proposed Kalman filter-based tracking has two stages: the *prediction* and the *update* stages. In prediction stage, the bounding boxes for pedestrians are predicted using the corresponding state of the bounding boxes in the previous frames. In the update stage, the observation of pedestrian in the current frame is used to update the predicted states of pedestrians.

Let  $\mathbf{s}_t^i$  be the state vector of  $i^{\text{th}}$  bounding pedestrian window in frame  $t$ . To track multiple pedestrians, it is convenience to have multiple Kalman filters, e.g., one for each pedestrian detected in the frame as follows

$$\begin{aligned}\mathbf{s}_t^i &= \mathbf{F}_{t-1}^i \mathbf{s}_{t-1}^i + w_{t-1}, \\ \mathbf{z}_t^i &= \mathbf{H}_t^i \mathbf{s}_t^i + v_t,\end{aligned}$$

where  $\mathbf{F}_{t-1}^i$  and  $\mathbf{H}_t^i$  denote the state transition and the measurement matrices for the  $i^{\text{th}}$  pedestrian, respectively. The vectors  $w_{t-1}$  and  $v_t$  are noise terms which are assumed to be Gaussians with zero mean and covariance matrices  $Q_t$  and  $R_t$ . The prediction stage involves reasoning about the state vectors and their associated error covariance matrices at time  $t$  given the measurements up to  $t-1$  as follows:

$$\begin{aligned}\mathbf{s}_{t|t-1}^i &= \mathbf{F}_{t-1}^i \mathbf{s}_{t-1}^i + w_{t-1}, \\ \mathbf{P}_{t|t-1}^i &= \mathbf{F} \mathbf{P}_{t-1|t-1}^i \mathbf{F}^T + Q_t.\end{aligned}$$

Next, in the update stage updates, the state vectors and their error covariance matrices with the current observations are as follows:

$$\begin{aligned}\mathbf{K}_t^i &= \mathbf{P}_{t|t-1}^i \mathbf{H}^T \left( \mathbf{H} \mathbf{P}_{t|t-1}^i \mathbf{H}^T + R_t \right)^{-1}, \\ \mathbf{s}_{t|t}^i &= \mathbf{s}_{t|t-1}^i + \mathbf{K}_t^i \left( \mathbf{z}_t^i - \mathbf{s}_{t|t-1}^i \right), \\ \mathbf{P}_{t|t}^i &= \left( \mathbf{I} - \mathbf{K}_t^i \mathbf{H} \right) \mathbf{P}_{t|t-1}^i,\end{aligned}$$

where  $\mathbf{K}_t^i$  is the Kalman gain which emphasizes how prediction and measurement are intimately related. Therefore, the process of fusion begins with identifying observation models and associated measurement noises for each observation modality. For instance, in our settings, the bounding boxes from the RGB and depth images are considered as



positional information whereas the optical flow provides the velocity information only. This way we can assign a separate observation model for updating the joint prediction state.

## V. EXPERIMENT RESULTS

We evaluate the pedestrian detection performances in terms of Miss Rate (MR) vs False Positive Per Image (FPPI) and also provide accuracy, precision, recall, and run-time efficiency of the model. We conduct our experiments on a 64-bit Ubuntu 18.04 server that has an Intel(R) Core(TM) i9-7900X CPU @ 3.30GHz with 64GB memory. In our setup, we also have an NVIDIA GeForce RTX 2080 GPU with 8GB memory.

### A. Dataset

We perform our experiments on the Waymo open dataset which contains a wide range of diverse examples since data are collected among Phoenix, Mountain View, and San Francisco cities in the USA, plus it contains daytime and nighttime driving data [27]. This dataset is recently released and comprising of large-scale multimodal sensor data, i.e., high-resolution camera and LiDAR data. In particular, the dataset is collected using five LiDAR sensors and five high-resolution pinhole cameras and contains four object classes: Vehicles, Pedestrians, Cyclists, Signs. It has 12.6M high-quality 3D bounding box labels in total for 1,200 segments for LiDAR data. On the other hand, it has 11.8M 2D tightly fitting bounding box labels in total for 1,000 segments of camera data. In our setup, we use front camera images and project LiDAR data onto their corresponding camera images.

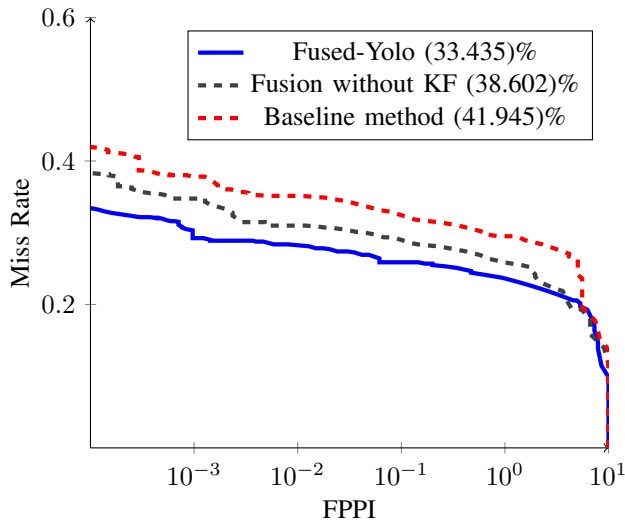


Fig. 4: The comparison among Fused-YOLO, Fusion without applying Kalman Filter, and Baseline YOLOv5. Fused-YOLO model get the lowest miss rate of 33.435% among others.

### B. Performance Analysis

We evaluate our proposed Fused-YOLO and the baseline YOLOv5 in terms of Miss Rate vs False Positive Per Image (FPPI) curve in Fig. 4. Testing on 456 number of images from the Waymo dataset, the proposed Fused-YOLO shows

the miss rate of 33.435% whereas the YOLOv5 has the miss rate of 41.945%. This is because fusion helps more for accurately detecting pedestrians in ill-lit conditions. In low illumination conditions, the shape of pedestrians given by depth images entails useful features for pedestrian detection, which are more challenging to detect from camera images. Especially, we notice that the Fused-YOLO achieves significantly less miss rate in low illumination conditions in contrast to the baseline YOLOv5 model. On the other hand, fusion without the Kalman filter exhibits 38.602% miss rate. Because when naively fusing the bounding boxes from the depth and RGB images, it increases the false detection. On contrary, the Kalman filter provides a systematic approach to reduce false detection and achieves the lowest miss rate.

We find that our tracking method performs very well even in the cases where there are some miss detections in sequential frames. Fig. 5 illustrates the performance of our tracker on a sequence of images. Our tracking method uses a Kalman filter to predict multiple pedestrian bounding boxes. Then, fusing the detection results from RGB images and depth images, the Kalman filter update pedestrians' state estimation. Correlating the previous bounding boxes to the current estimation, the Kalman filter can track pedestrians even if the detector might fail to detect pedestrians on the current frame. Furthermore, additional detected bounding boxes from depth images help the Fused-YOLO to track the pedestrians robustly. We observe that the Fused-YOLO achieves negligible runtime performance overhead (28 FPS) in contrast to the baseline YOLOv5 model (30 FPS).

## VI. CONCLUSION

We developed a real-time accurate pedestrian detection and tracking framework by fusing camera and LiDAR sensor data. The developed framework is integrated with the Kalman filter to accurately and robustly detect and track multiple pedestrians. The novelty of our framework lies in the adoption of the Kalman filter for both sensor fusion and tracking applications while minimizing the overall runtime overhead. Experimental results demonstrated the improvement over the baseline YOLOv5 model. Our fusion method outperforms YOLOv5 in terms of detection and tracking accuracy with a negligible amount of runtime overhead. The difference becomes even more pronounced in ill-lit conditions when pedestrians are hard to find on camera images. Finally, our implementation of the Kalman filter along with the optical flow algorithm reduced the detection miss rate and improved the overall performance. Future work includes extending our tracking method for pedestrians' behavior/intention analysis.

## ACKNOWLEDGMENT

The authors also appreciate the support from the North Carolina Department of Transportation (NCDOT) under the award numbers RP2019-28 and TCE2020-03, the National Science Foundation under award numbers 2018879 and 2000320, as well as the support from SAE International and Ford Motor Company.



Fig. 5: **Robust pedestrians tracking:** In the top row, there are two pedestrians detected in the first frame and then in the second frame model failed to detect one of the pedestrians. However, in the next two consecutive frames that pedestrian is detected and tracked again which represents the robustness of our method. The green and cyan bounding boxes in the top row represent detected and tracked pedestrians, respectively. On the other hand, the bottom row shows the corresponding LiDAR scans where the relative distance of objects from the ego vehicle is shown by a jet color spectrum.

## REFERENCES

- [1] M. M. Islam, A. A. R. Newaz, B. Gokaraju, and A. Karimodini, "Pedestrian detection for autonomous cars: Occlusion handling by classifying body parts," in *2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, 2020, pp. 1433–1438.
- [2] A. Homaifar, A. Karimodini, B. A. Erol, M. A. Khan, E. Tunstel, R. L. Roberts, R. F. Young, K. Snyder, R. S. Swanson, M. Jamshidi *et al.*, "Operationalizing autonomy: A transition from the innovation space to real-world operations," *IEEE Systems, Man, and Cybernetics Magazine*, vol. 5, no. 4, pp. 23–32, 2019.
- [3] P. Voigtlaender, M. Krause, A. Osep, J. Luiten, B. B. G. Sekar, A. Geiger, and B. Leibe, "Mots: Multi-object tracking and segmentation," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 7942–7951.
- [4] P. Bergmann, T. Meinhardt, and L. Leal-Taixe, "Tracking without bells and whistles," in *IEEE/CVF International Conference on Computer Vision*, 2019, pp. 941–951.
- [5] Z. Wang, L. Zheng, Y. Liu, and S. Wang, "Towards real-time multi-object tracking," *arXiv preprint arXiv:1909.12605*, p. 4, 2019.
- [6] A. Milan, S. H. Rezatofighi, A. Dick, I. Reid, and K. Schindler, "Online multi-target tracking using recurrent neural networks," in *AAAI Conference on Artificial Intelligence*, vol. 31, no. 1, 2017.
- [7] D. Frossard and R. Urtasun, "End-to-end learning of multi-sensor 3d tracking by detection," in *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2018, pp. 635–642.
- [8] A. Asvadi, P. Girao, P. Peixoto, and U. Nunes, "3d object tracking using rgb and LiDAR data," in *International Conference on Intelligent Transportation Systems*. IEEE, 2016, pp. 1255–1260.
- [9] N. Wojke, A. Bewley, and D. Paulus, "Simple online and realtime tracking with a deep association metric," in *IEEE international conference on image processing*. IEEE, 2017, pp. 3645–3649.
- [10] W. Zhang, L. Tian, C. Li, and H. Li, "A SSD-based crowded pedestrian detection method," in *International Conference on Control, Automation and Information Sciences*. IEEE, 2018, pp. 222–226.
- [11] E. Zdobrischi and M. Negru, "Pedestrian detection based on tensor-flow yolov3 embedded in a portable system adaptable to vehicles," in *2020 International Conference on Development and Application Systems (DAS)*. IEEE, 2020, pp. 21–26.
- [12] S. Zhang, J. Yang, and B. Schiele, "Occluded pedestrian detection through guided attention in cnns," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 6995–7003.
- [13] L. Oliveira and U. Nunes, "Context-aware pedestrian detection using LiDAR," in *Intelligent Vehicles Symposium*. IEEE, 2010, pp. 773–778.
- [14] K. Kidono, T. Miyasaka, A. Watanabe, T. Naito, and J. Miura, "Pedestrian recognition using high-definition LiDAR," in *Intelligent Vehicles Symposium (IV)*. IEEE, 2011, pp. 405–410.
- [15] K. Liu, W. Wang, and J. Wang, "Pedestrian detection with LiDAR point clouds based on single template matching," *Electronics*, vol. 8, no. 7, p. 780, 2019.
- [16] T.-C. Lin, D. S. Tan, H.-L. Tang, S.-C. Chien, F.-C. Chang, Y.-Y. Chen, W.-H. Cheng, and K.-L. Hua, "Pedestrian detection from LiDAR data via cooperative deep and hand-crafted features," in *IEEE International Conference on Image Processing (ICIP)*. IEEE, 2018, pp. 1922–1926.
- [17] G. Chen, Z. Mao, H. Yi, X. Li, B. Bai, M. Liu, and H. Zhou, "Pedestrian detection based on panoramic depth map transformed from 3d-LiDAR data," *Periodica Polytechnica Electrical Engineering and Computer Science*, vol. 64, no. 3, pp. 274–285, 2020.
- [18] D. Matti, H. K. Ekenel, and J.-P. Thiran, "Combining LiDAR space clustering and convolutional neural networks for pedestrian detection," in *IEEE International Conference on Advanced Video and Signal Based Surveillance*. IEEE, 2017, pp. 1–6.
- [19] T. Kim, M. Motro, P. Lavieri, S. S. Oza, J. Ghosh, and C. Bhat, "Pedestrian detection with simplified depth prediction," in *International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2018, pp. 2712–2717.
- [20] C. Premebida, J. Carreira, J. Batista, and U. Nunes, "Pedestrian detection combining RGB and dense LiDAR data," in *IEEE/RSS/International Conference on Intelligent Robots and Systems*. IEEE, 2014, pp. 4112–4117.
- [21] R. Lahmyed and M. E. Ansari, "Multisensors-based pedestrian detection system," in *IEEE/ACS International Conference of Computer Systems and Applications (AICCSA)*, 2016, pp. 1–4.
- [22] G. Ning, Z. Zhang, C. Huang, X. Ren, H. Wang, C. Cai, and Z. He, "Spatially supervised recurrent convolutional neural networks for visual object tracking," in *IEEE International Symposium on Circuits and Systems*, 2017, pp. 1–4.
- [23] K. Granström, S. Renter, M. Fatemi, and L. Svensson, "Pedestrian tracking using velodyne data—stochastic optimization for extended object tracking," in *2017 IEEE intelligent vehicles symposium (iv)*. IEEE, 2017, pp. 39–46.
- [24] G. Jocher, A. Stoken, J. Borovec, A. Chaurasia, T. Xie, C. Liu, V. Abhiram, T. Laughing *et al.*, "Ultralytics/yolov5: v5.0—yolov5-p6 1280 models," AWS, *Supervise. ly and YouTube Integrations*. [accessed on 1 May 2021], 2021.
- [25] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "Yolov4: Optimal speed and accuracy of object detection," *arXiv preprint arXiv:2004.10934*, 2020.
- [26] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.
- [27] P. Sun, H. Kretschmar, X. Dotiwalla, A. Chouard, V. Patnaik, P. Tsui, J. Guo, Y. Zhou, Y. Chai, B. Caine *et al.*, "Scalability in perception for autonomous driving: Waymo open dataset," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 2446–2454.