2022

Automated Vehicle to Vehicle Conflict Analysis at Signalized Intersections by Camera and LiDAR Sensor Fusion

Alabi Mehzabin Anisha
University of Central Florida

Part of the Civil Engineering Commons, and the Transportation Engineering Commons

Find similar works at: https://stars.library.ucf.edu/etd2020

University of Central Florida Libraries http://library.ucf.edu

This Masters Thesis (Open Access) is brought to you for free and open access by STARS. It has been accepted for inclusion in Electronic Theses and Dissertations, 2020- by an authorized administrator of STARS. For more information, please contact STARS@ucf.edu.

STARS Citation

https://stars.library.ucf.edu/etd2020/972
AUTOMATED VEHICLE TO VEHICLE CONFLICT ANALYSIS AT SIGNALIZED INTERSECTIONS BY CAMERA AND LIDAR SENSOR FUSION

by

ALABI MEHZABIN ANISHA
B.S. Bangladesh University of Engineering and Technology, 2018

A thesis submitted in partial fulfillment of the requirements for the degree of Master of Science in the Department of Civil, Environmental, and Construction Engineering in the College of Engineering and Computer Science at the University of Central Florida

Orlando, Florida

Spring Term
2022

Major Professor: Mohamed Abdel-Aty
© 2022 Alabi Mehzabin Anisha
ABSTRACT

This research presents an approach for safety diagnosis using sensor fusion techniques. This work fuses the outputs of a roadside low-resolution camera and a solid-state LiDAR. For vehicle classification and detection in videos, the YOLO v5 object detection model was utilized. The raw 3D point clouds generated by the LiDAR are processed by two manual steps - ground plane transformation and background segmentation, and two real-time steps - foreground clustering, and bounding box fitting. Taking the generated 2D bounding boxes of both camera and LiDAR, we associate the common bounding box pairs by thresholding on the Euclidean distance threshold of 6 ft between the centroid pairs. We perform weighted measurement update based on the RMSE of each of the sensor’s detection compared to manually labeled ground truths. The fused measurements are tracked by using linear constant velocity Kalman Filter. With the generated trajectories, we compute post encroachment time (PET) at pixel level conflicts based on the generated vehicle trajectories. We have proposed a complete bipartite graph matching strategy of vehicle parts along with the conflict angle to obtain conflict types - rear-end, sideswipe, head-on, and angle conflict. A case study on a signalized intersection is presented. The output of the proposed framework performs significantly better than the single sensor-based systems in terms of the number of detections and localization. It is expected that the proposed method can be employed to diagnose road safety problems and inform the required countermeasures.
This thesis is dedicated to my husband, Shakib, who have supported me in every way throughout this academic journey, and my family back at home who have fought the struggles of COVID-19 pandemic while cheering me all the way.
ACKNOWLEDGMENTS

I want to thank my Advisor Dr. Mohamed Abdel-Aty for his continuous support and guidance. My colleagues - Amr Abdelraouf and Dr. Yina Wu deserve a special thanks for supporting me to make this work into fruition.
TABLE OF CONTENTS

LIST OF FIGURES viii
LIST OF TABLES ix
CHAPTER 1: INTRODUCTION 1
CHAPTER 2: LITERATURE REVIEW 3
  2.1 Vehicle Detection Using Camera 3
  2.2 Vehicle Detection Using LiDAR 4
  2.3 Sensor Fusion 6
CHAPTER 3: METHODOLOGY 9
  3.1 Video-based Vehicle Detection 9
  3.2 Point Cloud-based Vehicle Detection 9
    3.2.1 Ground Plane Transformation 10
    3.2.2 Background Segmentation 11
    3.2.3 Foreground Clustering 12
    3.2.4 Bounding Box Fitting 12
  3.3 Sensor Fusion 14
    3.3.1 Synchronization 14
    3.3.2 Coordinate Transform 14
    3.3.3 Data Association 15
    3.3.4 Measurement Fusion 16
    3.3.5 Tracking 18
  3.4 Conflict Estimation 19
CHAPTER 4: EXPERIMENTATION AND FINDINGS 20
  4.1 Data Collection 20
  4.2 Vehicle Detection Performance 21
    4.2.1 Fusion Multiplier Estimation 21
    4.2.2 Detection Performance Using Video 22
    4.2.3 Detection Performance Using Point Cloud 23
    4.2.4 Detection Performance Using Fusion 24
<table>
<thead>
<tr>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.3 Safety Performance</td>
<td>26</td>
</tr>
<tr>
<td>CHAPTER 5: CONCLUSION</td>
<td>30</td>
</tr>
<tr>
<td>LIST OF REFERENCES</td>
<td>32</td>
</tr>
</tbody>
</table>
LIST OF FIGURES

Figure 1. (a) Example frame captured by the camera and the corresponding output for the different LiDAR detection steps (b) background segmentation (c) foreground clustering (d) bounding box fitting........................................................................................................................................ 13

Figure 2: An example frame demonstrating association of camera (blue) & LiDAR (red) pair (a) on a camera-view and (b) top-view. ............................................................................................................................................. 16

Figure 3: A sample depiction of camera (blue) and LiDAR (red) measurements and their fusion (green) at camera view........................................................................................................................................... 18

Figure 4: Workflow of computing PET.................................................................................................................................................. 19

Figure 5: Experimental setup of camera and LiDAR facing the intersection............................................. 21

Figure 6: Trajectories generated by (a) camera (b) LiDAR and (c) Fusion are plotted in a top view......................................................................................................................................................... 25

Figure 7: A Bar chart showing the number of conflicts of different types – sideswipe, rear-end, head-on, and angle calculated by camera, LiDAR, and fusion at (a) PET < 2 seconds and (b) PET < 1-second thresholds for 15 minutes ............................................................................................................................ 27

Figure 8: A Heatmap showing the spatial distribution of conflicts - (a) camera rear-end, (b) camera sideswipe, (c) LiDAR rear-end, (d) LiDAR sideswipe, (e) fusion rear-end & (f) fusion sideswipe conflicts............................................................................................................................................. 29
LIST OF TABLES

Table 1: Performance of camera localization under varying IOU thresholds.......................... 23
Table 2: Performance of LIDAR Object Detection based on detection count.......................... 24
Table 3: Comparison of detection counts and performance across different methods ............... 26
CHAPTER 1: INTRODUCTION

Traffic safety analysis depends on the nature of collected data from the field. Traditional research on traffic data collection involves sensors located at fixed locations to analyze the traffic conditions for specific areas or road segments. Researchers relied on loop detectors or radar sensors deployed at road-segments that aggregate data 5 to 30 seconds. However, these works suffer from data unavailability on road segments where these sensors are not deployed and sensor malfunctioning. Therefore, the current research community has shifted its focus towards microscopic analysis using non-intrusive sensors like cameras and LiDAR. However, there are certain limitations associated with each of the sensors. Therefore, we apply sensor fusion framework to extract trajectory from camera and LiDAR. This research utilizes sensor fusion techniques to overcome the limitations of each sensor.

The contribution of this work is proposing a robust sensor fusion framework for traffic safety assessment. This study uses camera and LiDAR sensors on the lower end of the market price keeping wide-scale deployment in mind. We experimented with camera and LiDAR vehicle detection methods that do not require manual labeling. The key contribution is to propose a roadside camera and LiDAR output fusion framework. The idea is to leverage the visual information from the camera for classification and increase the precision of localization from LiDAR. Fused detections are tracked using SORT tracker (Bewley et al., 2016). After extracting the trajectories, surrogate safety parameters are computed for traffic safety assessment. Time-based surrogate safety parameter Post Encroachment Time (PET) is calculated for safety assessment. The fusion performance is compared with the standalone LiDAR and camera systems and ground truth.
The following sections of this thesis discuss the state-of-the-art of object detection using camera and LiDAR, and sensor fusion strategies. This section is followed by the discussion of our proposed methodology and the results analysis. Finally, we discuss the key takeaway of this research and possibilities of further exploration that this study opens.
CHAPTER 2: LITERATURE REVIEW

This section discusses the current state-of-the-art of road-user detection using camera and LiDAR, and safety analysis from detections and sensor fusion techniques that combine the outputs of multiple sensors for better performance.

2.1 Vehicle Detection Using Camera

Video-based traffic data analysis has been a long-researched topic. Researchers have been using video cameras to model safety parameters. Automatic conflict detection using traffic videos from roadside cameras has been a focus of researchers since cameras provide rich visual features to identify vehicle categories and localize them. Before the deep learning era, feature-based detectors were widely used. The classical methods of object detection like - Hough transforms and Scale Invariant Feature Transform (SIFT) would use feature descriptors (Illingworth and Kittler, 1988, Lowe, 1999). Recent convolutional neural network (CNN) based approaches are divided into two categories for object detection, namely – one stage and two stage detectors. One stage detector like the algorithms from YOLO family (YOLO, YOLO9000, YOLOv3) and SSD can perform real-time detection because they make a fixed number of predictions on grid. On the other hand, two stage detectors of R-CNN family (Fast R-CNN, Faster R-CNN, Mask R-CNN) use a proposal generation and making final prediction performs better in terms of precision (Girshick, 2015, Ren et al., 2017, He et al., 2017).

Feature-based tracking of vehicles at a signalized intersection using roadside cameras has been used widely (Saunier and Sayed, 2006). Later studies have annotated the extrinsic and intrinsic
camera parameters manually to transform the camera coordinates to the real world and extract feature-based trajectories (Autey et al., 2012). More recent researches used video analysis to compute TTC (Essa and Sayed, 2018). They have used TTC as a rear-end conflict indicator and developed Safety Performance Function (SPF) at signalized intersections. Researchers have also developed a traffic danger recognition model using traffic surveillance cameras (Yu et al., 2018). The purpose of the model was to identify and predict car crashes. This model predicts and identifies car crashes from surveillance cameras exploiting a 3D reconstruction of the road plane and predicted trajectories. Researches have used TTC extracted from roadside cameras to evaluate the safety impact of right-turn treatments on right-turn-on-red conflicts at signalized intersections (Guo et al., 2020). Another research have proposed a vision-based model to detect and count vehicles on the highway (Song et al., 2019). Roadside cameras have also been used in vehicle-pedestrian and vehicle-bicycle interactions. Researchers have developed an automated video analysis system to detect road users and identify conflicts in pedestrian-vehicle interactions (Ismail et al., 2009). Numerous research works predict pedestrian safety at intersections using roadside video cameras (Zhang et al., 2020b, Zhang et al., 2020c).

2.2 Vehicle Detection Using LiDAR

Roadside LiDAR provides high-resolution trajectories of vehicles and other road users. Moreover, reduced visibility conditions such as rain, reduced lighting, and climatological adversities do not impact the accuracy of 3D point clouds of LiDAR. Researchers proposed a method to identify near-crashes by exploiting trajectories from roadside LiDAR data (Wu et al., 2018b). This research aimed at vehicle-pedestrian near-crash identification and vehicle-pedestrian
conflicts. They considered three parameters to identify vehicle-pedestrian near-crash in the study: Time Difference to the Point of Intersection (TDPI); Distance between Stop Position and Pedestrian (DSPP); Vehicle-pedestrian speed-distance profile. Further research has developed a unique approach to classify vehicles based on roadside LiDAR (Wu et al., 2019). This approach considers six different features, including object height, to classify vehicles. These features are obtained from the vehicle trajectories collected by roadside LiDAR sensors. In this study, they applied Naive Bayes, K-nearest neighbor classification, random forest (RF), and support vector machine and found that RF exhibits the highest overall accuracy. Researchers have also proposed a method of data processing for the detection and tracking of multi-lane multi-vehicle trajectories using roadside LiDAR sensors in the context of connected vehicles (Chen et al., 2019). Their approach contains five main steps: region of interest (ROI) selection, ground surface filtering, point clustering, vehicle/non-vehicle classification, and geometrical vehicle tracking. Vehicle speed is an essential parameter for estimating surrogated safety measures. Although a roadside LiDAR sensor accurately provides the positions of an object, LiDAR-based speed estimation and tracking is still an open question. Other researchers developed a framework of roadside LiDAR to estimate vehicle speed accurately for vehicle detection and tracking (Zhang et al., 2020a). They first detected on-road vehicles from the observed point clouds and performed centroid-based tracking for extracting initial vehicle transformations. They applied unscented Kalman Filter and joint probabilistic data association filter in the tracking. Finally, they refined the tracking using an image matching process to increase the estimated vehicle speeds accuracy. The current state-of-the-art of roadside LiDAR-based vehicle tracking procedure follows a tracking-by-detection principle (Zhang et al., 2020a). In the first step, the raw LiDAR data require preprocessing, where the moving points are filtered.
out from the background. Subsequently, small groups of point cloud clusters are used for vehicle extraction. However, this approach does not utilize the visual features of the camera for vehicle classification; hence, only a broad classification of vehicle and non-vehicle is possible in this approach. One way of vehicles extraction is to detect the lanes that vehicles occupy (Wu et al., 2019). Researchers also suggested classifying vehicle and non-vehicle objects from the determined clusters for vehicle extraction (Chen et al., 2019). For vehicle tracking, Kalman filtering (KF) (Zhao et al., 2019) and Global Nearest Neighbor (GNN) (Wu et al., 2019) are widely used.

2.3 Sensor Fusion

It is evident from the literature that the camera performs well when it comes to classification and recognition based on visual features, and LiDAR performs well when it comes to localization. The performance of the video analysis system falls with the video quality. Most of the roadside traffic surveillance cameras have low resolution. Even though LiDAR is good at classifying motorized vs. non-motorized road users, it is hard to classify the different vehicle types. Researchers are trying to fuse the camera and LiDAR outputs to gain a better perception of the environment.

Sensor fusion is a widely used solution to use sensors for each other’s advantage. Researchers have tried to classify sensor fusion approaches in their respective books (Durrant-Whyte and Henderson, 2008, Bar-Shalom et al., 2011). Sensor fusion can be classified into three broad categories from an abstraction point of view, computational centralization point of view and the competitiveness of measurements point of view.
From an abstraction point of view, sensor fusion is classified into three categories – early fusion, mid-level fusion and high-level fusion. Early fusion deals with the raw data coming from each of the sensors and thus results in the least amount of information loss. The problem of this approach is not only the computational complexity but also the model loses its interpretability. Mid-level fusion utilizes detection output from each of the sensors where the sensors process their own data and generates detection. The high confidence estimates of multiple sensors are fused together to have better and more accurate results. Algorithms from the Bayesian family consisting of linear and nonlinear Kalman filters perform mid-level fusion from the detections. Mid-level fusion framework is more interpretable, but the performance of the overall process solely depends on the detector performance. Late fusion occurs in the trajectory level where each sensor generates their own trajectory by tracking multiple objects. After that, the multi-object tracks of different sensors are further fused together. Even though late fusion offers ease of implementation, this process causes the highest amount of information loss.

From a computational nature, sensor fusion can be classified into three categories – centralized, decentralized, and distributed fusion. Centralized fusion is typically a computing unit doing both computation and fusion. Decentralized fusion instead of fusing all the sensors together, sequentially fuses two sensors and forwards the output to the central unit. Distributed sensor fusion technique is an extension of decentralized fusion. Instead of sending the intermediate fusion results to a centralized unit, the intermediate fusion results are further processed together. This result is then sent to the central processing unit.

From a competition of measurements, sensor fusion can be of three types – competitive, complementary, and coordinated fusion. Competitive fusion aims to get better localization
accuracy as most accurate measurements are selected from competitive measurements of each sensor since each sensor is measuring the same output. Complementary fusion combines the insight observed by different sensors from different perspectives. Each sensor complements each other to obtain a bigger picture of the environment. Coordinated sensor fusion technique uses multiple sensors to observe the same environment or scene to generate a high dimensional reconstruction of the environment.

Recent research has used it in autonomous driving scenarios by training the model with a labeled dataset. The research effort of several researchers is focusing on fusing the outputs or the region of interest of camera and LiDAR by using a deep learning framework (Pang et al., 2020, Chen et al., 2021). The training dataset used is the self-driving cars dataset like KITTI or NuScenes (Geiger et al., 2013, Caesar et al., 2020). There is a lack of such datasets in roadside scenarios, and data labeling is exhaustive and costly. Also, training the models is a computationally expensive process that requires specialized hardware. Recent research on roadside camera and RADAR fusion emphasizes complementing vehicle localization from camera and speed estimation from RADAR (Wang et al., 2021, Liu et al., 2022, Du et al., 2021). To the best of our knowledge, we can claim that there is no existing research using roadside camera and LiDAR aiming to enhance perception for road safety. Our research aims to fill the gap by applying sensor fusion in roadside camera and LiDAR that is mid-level, centralized and competitive to gain a better localization performance for traffic safety assessment in real-time.
CHAPTER 3: METHODOLOGY

This section describes the methodology of this study. The workflow consists of camera detection, LiDAR detection, trajectory extraction through sensor fusion, and safety evaluation. In a real-time scenario, the camera and LiDAR detection can happen in parallel while the sensor fusion process follows generating detection from both sensors. Finally, with the generated trajectory we discuss the procedure of conflict estimation.

3.1 Video-based Vehicle Detection

The camera provides rich visual features about road users. Each image is directly passed through a deep learning-based video detector that outputs the classification and localization of the detected vehicles. In this study we have used the most recent implementation of YOLOv5 for the object detection (Jocher et al., 2020). The weights are trained using publicly available Common Object in Context (COCO) dataset that can classify objects of categories – passenger car, bus, and truck (Lin et al., 2014).

3.2 Point Cloud-based Vehicle Detection

The proposed methodology for vehicle detection in the LiDAR point cloud consists of two preprocessing steps: ground plane transformation and background segmentation and two real-time steps: foreground clustering and bounding box fitting. The preprocessing steps utilize the first n point cloud frames to calibrate and compute the ground plane and background map. Those assets can then be used for online processing of the remainder of the point cloud stream.
3.2.1 Ground Plane Transformation

Due to the high position and the downward-tilted scanning angle of the LiDAR sensor, it is crucial to identify the ground plane in the point cloud and to transform it into the $XY$ plane. This step leads to the computation of more semantically correct detection—i.e., upright 3D bounding boxes with their bottom faces placed on the ground, which consequently simplifies the bounding box fitting step and the sensor fusion process. Firstly, rather than estimating the ground plane based on a single point cloud, all the point clouds in the preprocessing segment were aggregated into a single space. An array of point clouds was selected to identify the ground plane to account for space displacement variations caused by the sensor and mounting pole vibrations. Next, a point mesh was manually annotated in the ground plane of the aggregated LiDAR cloud point, and the points in the mesh were used to estimate a plane of best fit. Next, the normal vector $\vec{n}$ to the estimated plane was calculated. The rotation matrix $R$ which aligns $\vec{n}$ with the z-axis $\langle 0,0,1 \rangle^T$ was then computed. To correct the point cloud position, a translation vector $\vec{t} = \langle 0,0,t_z \rangle^T$, where $t_z = -z_{0,0}$, and $z_{0,0}$ is the $z$ value of the plane at $x = 0, y = 0$, was calculated. A final transformation matrix $T$ was constructed as shown in equation 1.

$$T = \begin{bmatrix} R & \vec{t} \\ \vec{0} & 1 \end{bmatrix}$$

$$\vec{0} = \langle 0,0,0 \rangle$$

(1)
3.2.2 Background Segmentation

To detect vehicles in the background, it is required first to classify which points belong to moving objects in the environment and which points are reflected off background objects. Inspired by the methodology proposed by Wu et al. (Wu et al., 2018a, Wu et al., 2017), a background map $BM$ was constructed. The background map determines whether a specific area belongs to the background by measuring the point cloud density across time. To achieve this goal, the background map divided the 3-dimensional point cloud space into $BM_s$-sided cubes. The stream of point clouds in the preprocessing segment was consecutively processed, and each 3D point was recorded in the background map according to its position. Static objects like stop signs or traffic lights amassed a large number of points in the same cubes, while points reflected off moving objects were scattered across the background map since the objects did not maintain their position in each point cloud frame. Finally, cubes that contained a number of points more prominent than the threshold $BM_{THRES}$ were considered part of the background. The foreground was simply defined as the set of points that do not belong to the background. The choice of the background map cube side length $BM_s$ is essential to the quality of the output map. If it is too small, the map would be too sensitive to small displacements caused by vibrations. The map won't be precise enough to separate background points from foreground points if it's too large.
3.2.3 Foreground Clustering

After identifying the foreground points in the point cloud, the next step is to cluster the points into individual objects. To identify those objects, the Density-Based Spatial Clustering of Applications with Noise (DBSCAN) clustering algorithm (Ester et al., 1996, Schubert et al., 2017) was utilized. DBSCAN was chosen because it is a clustering algorithm that makes no assumptions about the number of output clusters, but rather groups together point based on a distance function, a minimum number of points per cluster, and a maximum inter-cluster distance. Since the LiDAR deals with the real-world environment, the Euclidean function was chosen as the distance function.

3.2.4 Bounding Box Fitting

The final step is to find the best-fitting 3D bounding box for each identified object cluster. Since each object lies on the ground plane, the bottom four vertices of the bounding box have the same $x$ and $y$ coordinates as their corresponding top vertices. Additionally, the bottom points all have $z = 0$, and the top vertices all have an equal $z$ value. Hence, the problem was simplified to finding the top-view 2D bounding box for each detected object. Firstly, the 3D points for each object were projected to the 2D $XY$ plane. Next, the Convex Hull algorithm was applied to find the minimum enclosing polygon which contains all points in the cluster. The Quick hull implementation was utilized (Barber et al., 1996). Afterward, the Rotating Calipers algorithm (Toussaint, 1983) was used to find the minimum area bounding box that fits over the computed polygon. Finally, to extrude the 2D bounding box into the 3rd dimension, the $x$ and $y$ values from
the 2D box were used to create a 3D box with a bottom surface at \( z = 0 \) and a top surface at \( z \) equal to the maximum \( z \) value in the 3D cluster.

Figure 1 (a) demonstrates an example frame as captured by the camera. Figure 1 parts (b), (c), and (d) illustrate the corresponding output from the background segmentation, foreground clustering, and bounding box fitting steps, respectively.

Figure 1. (a) Example frame captured by the camera and the corresponding output for the different LiDAR detection steps (b) background segmentation (c) foreground clustering (d) bounding box fitting
3.3 Sensor Fusion

After obtaining the bounding box outputs from the camera and LiDAR, this section discusses the data fusion process. The camera provides the pixel localization and classification of vehicles in each frame. On the other hand, LiDAR provides the localization of the road users in its own coordinate system. The advantage of LiDAR is that distance between points can be measured in feet. However, LiDAR cannot be relied on for vehicle classification as it cannot differentiate distinctive visual features of each road user. Hence, for the vehicle classification process, we are entirely relying on the camera. The generated bounding box from camera and LiDAR is fused and tracked in the steps – synchronization, coordinate transform, data association, measurement fusion, and tracking.

3.3.1 Synchronization

Sensor synchronization is the first and important step before sensor fusion. Real-world sensors tend to have different frequencies of operation. Therefore, it is important to synchronize them at a common frequency before processing the data. The sensors we are using in this research – camera and LiDAR, have different frequencies of operation. The LiDAR performs on frequency $f_L$, and the camera is on frequency $f_c$. The highest common factors of $f_L$ and $f_c$ is $f$. Both camera and LiDAR is sampled at $f$ to ensure the synchronization of the data.

3.3.2 Coordinate Transform

To perform data association on camera and LiDAR, the coordinates of camera and LiDAR must be transformed to common coordinate system. However, we have applied a homographic transformation to both the camera and the LiDAR coordinates to enable fusion computation. A
homographic matrix $H_c$ transforms the camera coordinates $[x_i^c, y_i^c, 1]$ to its global coordinates $[X_i^c, Y_i^c, 1]$. Another homographic matrix $H_l$ is estimated that transforms the LiDAR coordinates $[x_j^l, y_j^l, 1]$ to its global coordinate $[X_j^l, Y_j^l, 1]$. The transformed coordinate system is the pixel coordinates of the snapshot of the top view of the study area. A linear least square method is used to find the homographic matrix based on manually selected points. In order to obtain solutions for equation (2) and (3) with $c_9 = 1$ and $l_9 = 1$ respectively for $H_c$ and $H_l$ singular value decomposition method is used (Zhang et al., 2020b, Naphade et al., 2019, Španhel et al., 2019, Tang et al., 2019).

$$H_c = \begin{bmatrix} c_1 & c_2 & c_3 \\ c_4 & c_5 & c_6 \\ c_7 & c_8 & c_9 \end{bmatrix}$$ \hfill (2)

$$H_l = \begin{bmatrix} l_1 & l_2 & l_3 \\ l_4 & l_5 & l_6 \\ l_7 & l_8 & l_9 \end{bmatrix}$$ \hfill (3)

3.3.3 Data Association

After coordinate transformation, this section discusses the data association. For each frame $t$, there is $N$ distinct observations from camera $\{C_1, C_2, ..., C_N\}$ and $M$ distinct observations from LiDAR $\{L_1, L_2, ..., L_M\}$. Before passing these observations to the measurement fusion step, it is important to associate the camera and LiDAR observations pairs corresponding to the same object. For computational efficiency, we are using the Euclidean distance metric to evaluate the distance between a camera and a LiDAR pair. After constructing a cost matrix $Q_t$ for the frame $t$, we have applied the Hungarian algorithm for linear assignment on $Q_t$ (Kuhn, 1955). This gives us $K$ pairs.
of common observations, \((N - K)\) unassociated camera observations and \((M - K)\) unassociated LiDAR observation. Figure 2 demonstrates a scene where the camera-LiDAR detection association is projected onto a camera plane (on the left) and the top-view plane (on the right).

![Figure 2](image)

**Figure 2**: An example frame demonstrating association of camera (blue) & LiDAR (red) pair (a) on a camera-view and (b) top-view.

### 3.3.4 Measurement Fusion

This section discusses the Kalman Filter based fusion on the associated and unassociated observations of camera and LiDAR. Kalman filter continuously predicts the next state and updates it based on the measurements. The fusion happens in the update step of the Kalman Filter. As Gan et al. mentioned in their work, the fusion can happen in the measurement step in two ways (Gan and Harris, 2001, Durrant-Whyte and Henderson, 2008). One by augmenting the measurements, and another is by using weighted measurement. In our case, we have experimented with equally weighted measurements. For associated objects in the previous step, the filter is updated using the weighted measurements of camera and LiDAR as suggested in the work of Goh et al. (Goh et al.,
Kalman gain is computed after each measurement. For unassociated objects, the update happens once with one measurement only. We are using the constant velocity model in Kalman Filter. As shown in the equations (4)-(6), our observed state, $S$ is an 8-dimensional vector that consists of x and y position of centroid, area, aspect ratio, velocity in x and y direction, and change of area.

$$S = [X_{c_i}, Y_{c_i}, A_i, R_i, V_{xi}, V_{yi}, \Delta A_i]^T$$  \hspace{1cm} (4)

$$M_c = [X_{c1}, Y_{c1}, X_{c2}, Y_{c2}, X_{c3}, Y_{c3}, X_{c4}, Y_{c4}]^T$$ \hspace{1cm} (5)

$$M_l = [X_{l1}, Y_{l1}, X_{l2}, Y_{l2}, X_{l3}, Y_{l3}, X_{l4}, Y_{l4}]^T$$ \hspace{1cm} (6)

Hence, the algorithm is –

- Case 1: For each associated pair of $\{C_i, L_j\}$, update first with $M_{ij} = \frac{\alpha}{\alpha + \beta} M_{lj} + \frac{\beta}{\alpha + \beta} M_{ci}$

  where $\alpha$ and $\beta$ are the average measurement errors of camera and LiDAR respectively.

- Case 2: For each unassociated camera observation $C_p$, update with $M_{cp}$

- Case 3: For each unassociated LiDAR observation $L_q$, update with $M_{tq}$

A sample of measurement fusion is showed in the Figure 3 on camera view. We can see the bounding box generated by fusion are more alike to the LiDAR measurements and corrects distortion caused by the camera.
Figure 3: A sample depiction of camera (blue) and LiDAR (red) measurements and their fusion (green) at camera view

3.3.5 Tracking

After fusion, we use the SORT (Simple Online Real-time Tracking) tracker to track the detections in every frame (Bewley et al., 2016). In our case, we have used constant velocity Kalman filters to predict the locations of our tracks, and the SORT tracker algorithm associates previous and current measurements by computing the $IOU$ (Intersection over Union) and thresholding on $IOU_{TH}$. One of the challenges of the tracking process is to choose between camera or LiDAR-based measurements to associate with it. Therefore, we measure the IOU between all the camera and LiDAR detections and the tracks. The measurement that has maximum IOU with the track is chosen. Here, we apply the Hungarian algorithm (Kuhn, 1955) once again to perform a linear assignment. The detections that have $IOU$ less than the threshold are treated as new detections, and therefore, new trackers are created and associated with it.
3.4 Conflict Estimation

After trajectory extraction, we use the extracted trajectories to compute the conflicts in that region. Researchers have proposed a methodology to estimate conflicts after trajectory extraction (Wu et al., 2020). We compute Post Encroachment Time (PET) to assess the conflicts at the intersection by predicting intersecting point of each pair of vehicles at a frame. Figure 4 demonstrates the workflow of computing PET. Vehicle trajectories provide us with the information of position and speed. We also get the coordinates of all four corners of the vehicle bounding box. We compute the predicted conflict point coordinates of a vehicle pair at each frame by considering the current velocity and position. The predicted conflict point provides us with the angle between the conflicts. We perform a complete bi-partite graph matching between each point and select the pair of points with minimum distance. Each point corresponds to a vehicle part. The conflict angle with the vehicle parts given by matched pair of points helps us to compute the conflict type – rear-end, sideswipe, angle and head-on conflict. Conflicts are detected based on thresholding PET values.

![Workflow of computing PET](image)

Figure 4 : Workflow of computing PET
CHAPTER 4: EXPERIMENTATION AND FINDINGS

This section discusses our data collection procedure and then the performance of camera, LiDAR, and fusion based detection. Finally, the conflict analysis is discussed from the derived trajectories of all three methods.

4.1 Data Collection

We have collected data to validate our study at a typical four-leg intersection at the University of Central Florida (UCF) campus on several occasions from March 23, 2021, to November 24, 2021, during morning and afternoon peak periods. An ELP 1-megapixel indoor-outdoor CCTV camera was used for the video data collection. A Livox Horizon LiDAR, which is a solid-state LiDAR, was used for the point cloud data collection. The camera provides us with an image resolution of 640X480 at a 30 Hz frequency. The effective detection range of the LiDAR is 100-120 meters with a field of view 81.7° (Horizontal) × 25.1° (Vertical). Even though LiDAR operates by default at 20 frame per second, we have sampled both sensors at 10 frame per second for the easiness of synchronization. The equipment was set up at about 20-25 ft higher than the ground at the second floor to get a better view of the intersection as shown in Figure 5. It was made sure that camera and LiDAR have nearly the same view of the intersection. We have obtained only the vehicle trajectories from our study area, and no other sensitive or confidential information was retrieved. Faster R-CNN detection was conducted at a 0.1-second interval of the data. For background map construction, the number of frames used $n$ is set to 900, which consists of a 90 sduration, and the threshold parameter for clustering $BM_5$ is set to 0.05m. For vehicle tracking after
fusion, the $IOU_{TH}$ is set to 0.5, which is a standard in the literature. All the frameworks used for the experimentation are implemented in Python.

![Experimental setup of camera and LiDAR facing the intersection](image)

Figure 5 : Experimental setup of camera and LiDAR facing the intersection

### 4.2 Vehicle Detection Performance

#### 4.2.1 Fusion Multiplier Estimation

Measurement fusion between camera and LiDAR requires us to estimate the multipliers $\alpha$ and $\beta$. We have determined the root mean squared error (RMSE) of the detections of camera and LiDAR with the ground truth generated by us. In this case, we have compared the distance between centroids of sensor detections with the ground truth in feet. For camera the RMSE is 4.96 ft and for LiDAR this is 3.32 ft. Therefore, according to our proposed framework, $\alpha = 4.96$ for camera and $\beta = 3.32$ for LiDAR to perform measurement fusion.
4.2.2 Detection Performance Using Video

A semi-automated strategy was employed to determine the ground truth bounding boxes in the input videos. Firstly, the road user bounding boxes were generated using the mask RCNN (He et al., 2017) object detection algorithm. After that the generated outputs were manually fixed which generated the ground truth for camera detection evaluation.

The COCO evaluation metrics has been used to compute the AP (Average Precision) and AR (Average Recall) for the vehicles based on the bounding box IOU (Intersection over Union) between ground truth and generated results. For each frame, the True Positive (TP), False Positive (FP) and False Negative (FN) detection for every road user class were computed. Intersection Over Union (IOU) metric was used to determine the true positive and false positives between ground truth and generated outputs. For the same frame the set of generated outputs denoted as $GO$ and ground truth denoted as $GT$. For $i^{th}$ box in $GO$ and $j^{th}$ box in $GT$, the bounding box IOU was calculated as shown in equation (7).

$$IOU(i,j) = \frac{Intersection(GO_i, GT_j)}{Union(GO_i, GT_j)} \quad (7)$$

The TP, FP, and FN were generated using different $IOU$ thresholds within the range 0.5 – 0.95 as showed in Table 1. False positives were computed as the number of bounding boxes in GO that did not match with any of the bounding boxes in GT. False negatives were calculated as the number of bounding boxes in GT that did not match with any of the bounding boxes in GO. True Positives were quantified as the number of bounding boxes in GO and GT that have $IOU$ greater than or equal to the defined threshold. Matched bounding boxes having $IOU$ less than the defined threshold were added to the false positives count.
\[ AP = \frac{TP}{TP + FP} \]  (8)
\[ AR = \frac{TP}{TP + FN} \]  (9)

Table 1: Performance of camera localization under varying IOU thresholds

<table>
<thead>
<tr>
<th>IOU</th>
<th>AP</th>
<th>AR</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.5</td>
<td>87%</td>
<td>99%</td>
</tr>
<tr>
<td>0.7</td>
<td>87%</td>
<td>99%</td>
</tr>
<tr>
<td>0.75</td>
<td>87%</td>
<td>99%</td>
</tr>
<tr>
<td>0.8</td>
<td>87%</td>
<td>99%</td>
</tr>
<tr>
<td>0.85</td>
<td>86%</td>
<td>99%</td>
</tr>
<tr>
<td>0.9</td>
<td>73%</td>
<td>99%</td>
</tr>
<tr>
<td>0.95</td>
<td>29%</td>
<td>98%</td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td><strong>80%</strong></td>
<td><strong>99%</strong></td>
</tr>
</tbody>
</table>

4.2.3 Detection Performance Using Point Cloud

For validation, a 15-minutes-long segment of point cloud and the corresponding video was considered. Table 2 lists the detection results. In this case, we have taken a count-based evaluation approach from collected ground truths and LiDAR detections rather than IOU evaluation. The rationale behind doing count-based evaluation is LiDAR detection precision do not vary much within IOU threshold ranges. However, LiDAR happen to miss vehicles present in the scene. The number of ground truth vehicles in this period was 21,193 instances in 5240 frames. LiDAR detection for all these frames was 15550. The proposed algorithm had a precision of 90%. However,
the recall was 66%. The reason behind this is that the proposed algorithm outputs very few false positives. However, the number of false negatives is relatively high due to the presence of black cars, which poorly reflect the light beam generated from LiDAR. The vehicles adjacent to the black traffic poles were also prone to misdetection due to occlusion.

Table 2: Performance of LIDAR Object Detection based on detection count

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>Detected Object Count</td>
<td>15550</td>
</tr>
<tr>
<td>Ground Truth Count</td>
<td>21193</td>
</tr>
<tr>
<td>True Positive</td>
<td>13939</td>
</tr>
<tr>
<td>False Positive</td>
<td>1611</td>
</tr>
<tr>
<td>False Negative</td>
<td>7254</td>
</tr>
<tr>
<td>Precision</td>
<td>90%</td>
</tr>
<tr>
<td>Recall</td>
<td>66%</td>
</tr>
</tbody>
</table>

4.2.4 Detection Performance Using Fusion

We have compared the number of vehicle detections by three types of methodology – the camera only, LiDAR only, and our proposed methods. The generated trajectories from three types of methods are plotted side by side for comparison in Figure 6. The plot shows that the camera trajectories are relatively dense on the approaches and shallow at the intersection. However, LiDAR bounding boxes are dense at the intersection and able to capture turning movements at the intersection. Fusion combines these two types of trajectories and generate a bigger picture at an intersection and its approaches.
We have listed a numerical comparison of vehicle detection performance using camera, LiDAR, and proposed fusion method in Table 3. We have manually counted car instances from videos of 15 minutes. The total number of vehicle instances present are 1836 which is counted manually. The number of false positives is lower in LiDAR detection even though it has a relatively high false negative. Utilizing the fusion, we have been able to reduce the number of false positives to 47 which is much lower than the false positive count only using camera. Thus, the precision and recall of camera-based method is 90.319% and 94.265% respectively. The LiDAR-based methods have higher precision of 97.002% but the recall is as low as 74.019%. Fusion based approach improves both the precision and the recall respectively to 97.384% and 95.316% which is higher than single sensor-based methods.
Table 3: Comparison of detection counts and performance across different methods

<table>
<thead>
<tr>
<th>Metrics</th>
<th>Camera Only</th>
<th>LiDAR Only</th>
<th>Fusion</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ground Truth</td>
<td></td>
<td>1836</td>
<td></td>
</tr>
<tr>
<td>True Positive</td>
<td>1726</td>
<td>1359</td>
<td>1750</td>
</tr>
<tr>
<td>False Positive</td>
<td>185</td>
<td>42</td>
<td>47</td>
</tr>
<tr>
<td>False Negative</td>
<td>105</td>
<td>477</td>
<td>86</td>
</tr>
<tr>
<td>Precision</td>
<td>90.319%</td>
<td>97.002%</td>
<td>97.384%</td>
</tr>
<tr>
<td>Recall</td>
<td>94.265%</td>
<td>74.019%</td>
<td>95.316%</td>
</tr>
</tbody>
</table>

4.3 Safety Performance

We have calculated the PET for all the tracked vehicles. For each pixel pair, the arrival and departure timestamps are denoted by the frame numbers. Each frame has a time difference of 0.1 seconds. Therefore, the arrival and departure timestamp for different ids is calculated by multiplying the time difference with the frame number difference. In Figure 7, we see the comparison of the number of conflicts in 2 second (7a) and 1 second (7b) thresholds. As expected, we observe that the number of conflicts decreases with the threshold. The difference in camera, LiDAR, and fusion measurement is due to the localization of vehicles in these sensors inside the conflict area.
Figure 7: A Bar chart showing the number of conflicts of different types – sideswipe, rear-end, head-on, and angle calculated by camera, LiDAR, and fusion at (a) PET < 2 seconds and (b) PET < 1-second thresholds for 15 minutes.

In most cases, fused trajectories are giving more conflicts than individual sensors – camera and LiDAR. However, for side swipe the case is different where fusion identifies less sideswipes than camera does. This is explainable from the change in bounding box edges of fusion output. Fused outputs look a lot more like LiDAR than camera for the four corners of the vehicle, hence the angle computation is influenced by the bounding box edges. In total the number of conflicts of Fusion is 95 whereas camera is 67 and LiDAR is 50. Therefore, fusion increases the number of conflicts 41.7% from camera and 90% from LiDAR at PET < 2 threshold.

As we see that most of the conflicts observed are sideswipe and rear-end conflicts. The spatial distribution of conflicts is shown in Figure 8. The spatial density of sideswipe conflicts in the camera is high in the southbound and westbound direction (8a). However, the rear-end conflicts are dense in the northbound and southbound direction (8b). For LiDAR, the sideswipe and rear-end counts are high in density in the westbound direction (8c,8d). The observations from sensor
fusion differ in terms of density. Rear-end conflicts are high in southbound and westbound directions (8e). Sideswipe conflicts are high in westbound and distributed in the intersection (8f).
Figure 8: A Heatmap showing the spatial distribution of conflicts - (a) camera rear-end, (b) camera sideswipe, (c) LiDAR rear-end, (d) LiDAR sideswipe, (e) fusion rear-end & (f) fusion sideswipe conflicts.
CHAPTER 5: CONCLUSION

Our proposed methodology can be easily deployed in real-time for traffic safety assessment. We have increased the redundancy through sensor fusion that decreases the miss rate as well as increase the precision of bounding box detection. We have used the LiDAR detections to generate precise bounding boxes for vehicles. Fusion has increased the overall accuracy of detection counts and precision of vehicle size. Camera and LiDAR provide 90.319% and 97.002% precision rate respectively and fusion detects 97.384% of the vehicle correctly present in the scene. On the other hand, the recall for camera and LiDAR is respectively 94.265% and 74.019%. Using fusion, the recall is 95.316%. Thus, we can utilize the advantage of both sensors and get better precision and recall. Furthermore, we have also shown that our proposed framework is able to compute more conflicts than camera and LiDAR. The number of increased total conflicts in fusion compared to camera and LiDAR is respectively 41.7% and 90%. Our approach has an ease of implementation and can be implemented to any intersection real-time for vehicle-to-vehicle safety assessment.

Our future work aims to experiment with deep learning-based object detection algorithms on point clouds to increase the recall of LiDAR based detections. We aim to extend our safety assessment approach by incorporating vulnerable road users like pedestrian and bicyclists in the analysis. We want to be able to extend our current framework in terms of individual detection performance of both sensors. We expect to tune our framework that is guaranteed to perform well in reduced visibility and unfavorable weather conditions as well. In addition to that, we want to experiment with complex motion models in vehicle tracking to compute safety performance using deceleration-based safety assessment measures like Deceleration Rate (DR) and Deceleration to
Safety Time (DST). Thus, we will be able to present a wholistic safety assessment of different road-users in different metric at an intersection.

BAR-SHALOM, Y., WILLETT, P. K. & TIAN, X. 2011. Tracking and data fusion, YBS publishing Storrs, CT, USA:.


TANG, Z., NAPHADE, M., LIU, M.-Y., YANG, X., BIRCHFIELD, S., WANG, S., KUMAR, R., ANASTASIU, D. & HWANG, J.-N. Cityflow: A city-scale benchmark for multi-

35


