FINAL REPORT

Exploring the use of LIDAR data from Autonomous Cars for Estimating Traffic Flow Parameters and Vehicle Trajectories

Date: October 2017

Mecit Cetin, PhD, Associate Professor, Old Dominion University
Cem Sazara, PhD student, Old Dominion University
Reza Vatani Nezafat, PhD student, Old Dominion University

Prepared by:
Transportation Research Institute (TRI)
Old Dominion University
135 Kaufman Hall
Norfolk, VA 23529

Prepared for:
Mid-Atlantic Transportation Sustainability University Transportation Center

University of Virginia
Charlottesville, VA 22904

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
4. Title and Subtitle Exploring the use of LIDAR data from Autonomous Cars for Estimating Traffic Flow Parameters and Vehicle Trajectories		**5. Report Date** 10/09/2017 **6. Performing Organization Code**
7. Author(s) Mecit Cetin, Cem Sazara, Reza Vatani Nezafat		**8. Performing Organization Report No.**
9. Performing Organization Name and Address Old Dominion University, Norfolk, VA 23529		**10. Work Unit No. (TRAIS** **11. Contract or Grant No.** DTRT13-G-UTC33
12. Sponsoring Agency Name and Address US Department of Transportation Office of the Secretary-Research UTC Program, RDT-30 1200 New Jersey Ave., SE Washington, DC 20590		**13. Type of Report and Period Covered** Final 11/5/15 – 9/30/17 **14. Sponsoring Agency Code**

15. Supplementary Notes

16. Abstract

LIDAR has become one of the major enabling technologies for autonomous vehicles. LIDAR sensors generate 3D point cloud data around the instrumented vehicle. These data enable detailed localization of both fixed and moving objects in the surrounding environment. In this project, various aspects of LIDAR data processing are studied with an emphasis on exploiting such data for estimating traffic flow parameters. To achieve that, LIDAR data are collected in the field by an instrumented vehicle under different traffic conditions. Data processing and machine learning algorithms are developed to detect and classify vehicles based on the point cloud data. These algorithms help track ambient vehicles and extract their trajectories. From the extracted trajectories, both microscopic and macroscopic traffic flow parameters can then be estimated. In the case when some vehicle trajectories are partially observed or missing for a short duration, e.g., due to occlusion, this report shows how such trajectories could be completed by employing car following models. To demonstrate the use of trajectory data for estimating traffic flow parameters, a richer dataset with all vehicle trajectories, i.e., the NGSIM data, are utilized. Models are developed to predict traffic density, or the number of unobserved vehicles between two known trajectories, under congested conditions from a sample of trajectories. Under the dense traffic conditions analyzed, the results show that the number of unobserved vehicles between two probes can be predicted with an accuracy of ±1 vehicle almost always.

17. Key Words	18. Distribution Statement
LIDAR, 3D point cloud, Kalman Filter, microscopic traffic flow, macroscopic traffic flow	No restrictions. This document is available from the National Technical Information Service, Springfield, VA 22161

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price
Unclassified	Unclassified	28	

Acknowledgements

The authors would like to thank Mid-Atlantic Transportation Sustainability Center – Region 3 University Transportation Center (MATS UTC) for funding this project.

Disclaimer

The contents of this report reflect the views of the authors, who are responsible for the facts and the accuracy of the information presented herein. This document is disseminated under the sponsorship of the U.S. Department of Transportation's University Transportation Centers Program, in the interest of information exchange. The U.S. Government assumes no liability for the contents or use thereof.

Table of Contents

1. Problem:

Traditionally, fixed-point detectors such as loop detectors have been the dominant technology for collecting traffic flow data. Fixed point sensors, by definition, only collect data at the specific locations they are installed. On the other hand, vehicle tracking and localization technologies, such as GPS, enable measuring traffic flow characteristics along the paths of so called probe vehicles. Obviously, tracking a GPS-equipped mobile consumer device(s) within the vehicle also enables generating similar data. Data from these sources have been used in studying traffic flow phenomena [1-3]. Mobile sensors provide more diverse information compared to fixed location detectors as they are not limited to certain data collection locations. With the emergence of autonomous vehicles, it is now possible to expect another source data that would become commonly available as these enter the consumer market. Autonomous vehicles are typically equipped with LIDAR or other similar sensors to detect and track obstacles in the surrounding environment. LIDAR also provides a means to detect and track other vehicles around the autonomous car. There is very limited research on how data from such vehicles could be utilized to estimate traffic flow parameters. In this project, 3D point clouds data from a LIDAR installed on a vehicle are used for extracting information relevant for modeling traffic flow. Such data are very rich and have the advantage of defining the environment in more detail but also require fast processing methods due to large size of the data. This report discusses LDIAR data collection and processing to estimate micro and macro traffic flow parameters.

2. Approach and Research Objectives:

The main goal of this study is to estimate traffic flow parameters with the help of LIDAR. The specific goals and the approach of the proposed research are:

1. Collect sample LIDAR data under different traffic conditions on freeways and urban arterials in Hampton Roads
2. Develop algorithms to detect vehicles around a LIDAR-equipped car and classify them based on vehicle size.
3. Develop algorithms to track other vehicles while within the LIDAR range
4. Estimate macroscopic traffic flow parameters based on the detected vehicles along the path of the LIDAR-equipped vehicle

3. Introduction:

Environment detection is one of the key aspect of autonomous driving technologies. Detecting obstacles around the autonomous vehicle, such as moving or parked cars, pedestrians, trees, curbs etc., accurately and reliably is a challenge. To accomplish that, autonomous driving systems use sensor and camera systems to successfully navigate while avoiding obstacles. LIDARs have become an important part of these driving systems. LIDAR is a remote sensing technology that measures distance to a target by sending laser beams and measuring returned beams. LIDARs provide rich spatial information about the environment surrounding the sensor. In this project, 3D Velodyne VLP-16 LIDAR sensor is used [4]. This sensor has 16 laser beams with different pitch angles, in 2 degrees increment, on a 360-degree rotating unit. Figure 1. below illustrates a VLP-16 sensor and the orientation of the 16 laser beams (see also Table 1).

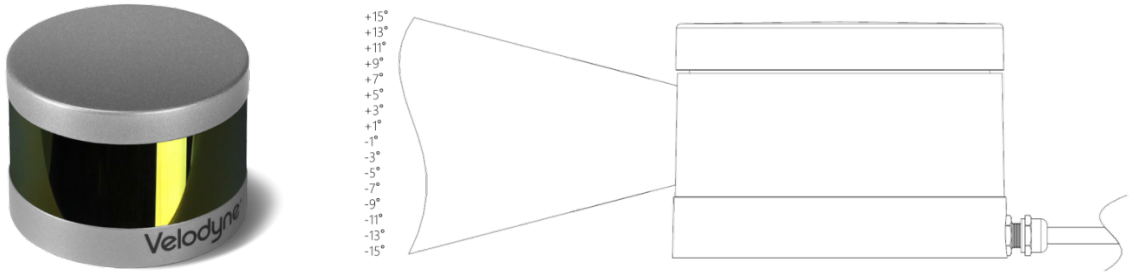

Figure 1. LIDAR and laser structure

The unit allows observing 360-degree horizontal and 30-degree vertical field of view producing approximately 30,000 points in a single scan. It can detect targets up to 100m away within a circle field of detection. Its rotation frequency can be adjusted between 5 to 20 Hz. In this project, it is used at 10 Hz which means a snapshot of the environment is taken in every 0.1 seconds. In this project, data are collected on urban roads with a Velodyne VLP-16 sensor.

Table 1. LIDAR laser vertical angles

Laser ID	Vertical angle
0	-15
1	1
2	-13
3	-3
4	-11
5	5
6	-9
7	7
8	-7
9	9
10	-5
11	11
12	-3
13	13
14	-1
15	15

An illustration of the 3D LIDAR data is in Figure 2. It provides detailed information about surrounding of the vehicle. Different colors designate different intensity levels which increase with reflective surfaces such as metals.

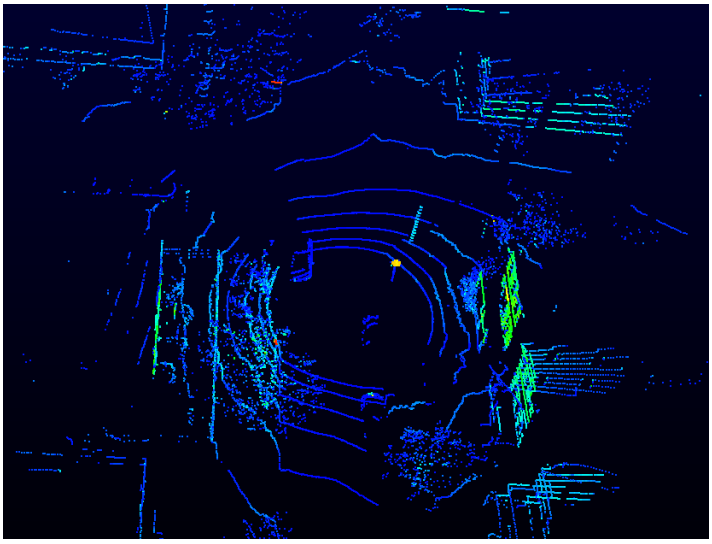

Figure 2. LIDAR data points near an intersection

4. Methodology:

The methodology in this project can be divided into two major parts: (i) Data Collection and Processing and (ii) Data Analysis. These are discussed next.

Data Collection and Processing:

Data collection part is accomplished by mounting the 3D LIDAR on a sedan vehicle and making multiple trips under different traffic conditions.

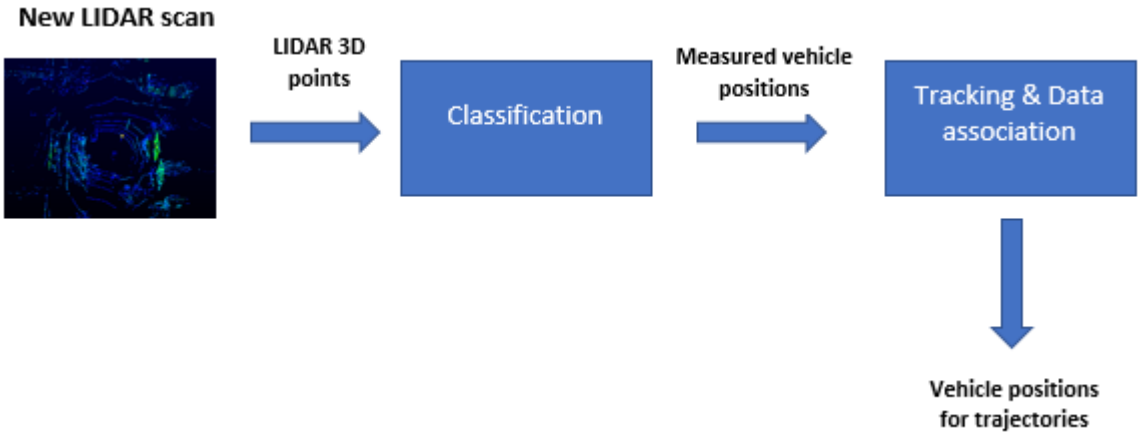

Figure 3. Data processing steps

The data processing step starts with each new LIDAR scan and repeats for every new LIDAR scan. Every LIDAR scan, which is also called a frame, is recorded with 10Hz frequency. This gives 10 LIDAR frames for every second. LIDAR points go through the "Classification" section where vehicles are detected. Then, this detected vehicle information is sent to the "Tracking & Data association" section where measurements are related to the previous measurements and tracking (relating to the previous measurements) is done. In this part, Kalman Filtering is used to track targets and data association is done with Hungarian Algorithm. Output of the "Tracking & Data association" step produces vehicle trajectory information.

Trajectory data collection is crucial as it prepares the basis data for the micro & macro traffic analysis part of the project.

Data Analysis:

In this part, we use the data collected in the previous step and investigate micro-macro traffic parameters. Making trips on different types of roads under different conditions helped us getting a diverse dataset that provides wide range of traffic situations such as stop-go, heavy and free-flow traffic. Data analysis part is divided into two sections: "Microscopic" Traffic Study and "Macroscopic" Traffic Study. In "Microscopic" part, we studied microscopic traffic phenomena which mostly includes car-following models. Collected data is used to calibrate different car following models: Gipps, IDM, Newell and Pipe's. These models provide information about driving behavior. In "Macroscopic" study part, NGSIM dataset is used to create a model for estimating number of vehicles between probe vehicles under stop-and-go conditions. Unsupervised and supervised methods are developed and used to estimate the unknown number of vehicles between probes and then it is used to calculate the macroscopic traffic parameter traffic density. As a future work, this work will be extended to include LIDAR data in addition to the NGSIM data in the estimation methods.

5. Research Findings:

In this section, we explain developed algorithms and methods to address our research objectives.

Data Collection:

Our 3D Velodyne LIDAR is mounted on the roof of our vehicle and 3D LIDAR data is collected. We drove LIDAR equipped vehicles in different traffic conditions on urban roads and highways in Hampton Roads Region in Virginia. Below in Figure 4, our data collection vehicle can be seen.

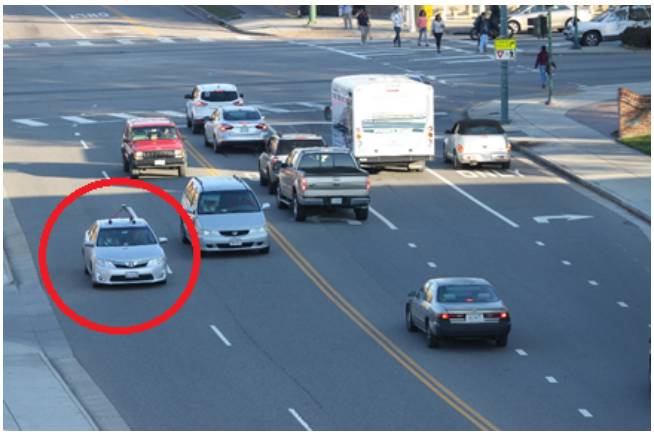

Figure 4. Our vehicle with LIDAR is marked with red circle

Hampton Roads area provides rich traffic information as it includes various types of infrastructures such as bridges and tunnels. LIDAR data is collected with multiple trips on different routes to capture diverse driving conditions. These routes are marked with blue in Figure 5.

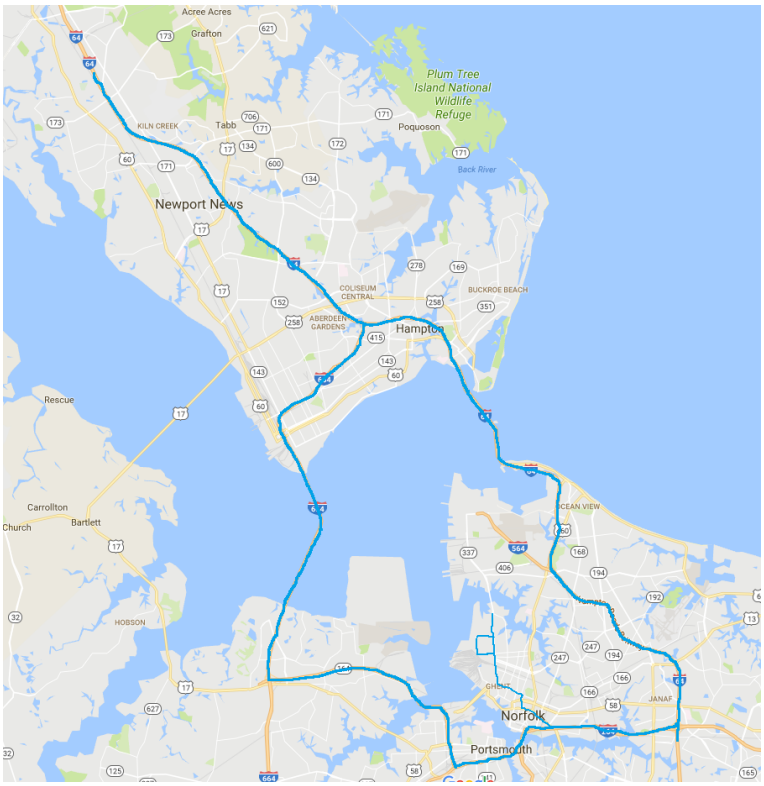

Figure 5. Data collection routes

Vehicle Classification:

Vehicle classification consists of two parts: Ground detection and target classification. Ground detection and target detection are closely related. Data coming from 3D LIDAR is first processed for ground detection and then for target or obstacle detection and classification. In ground detection, ground points are detected and removed from the complete set of points. Then target detection-classification is applied on the remaining points.

Ground Detection:

One of the challenges dealing with LIDAR data is distinguishing ground points from other points. Data coming from 3D LIDAR is first processed for ground detection. Many methods in the literature assumes a flat ground surface and simply applies a height threshold [5-7]. This causes problems in some different situations where the road is not flat. We show an example of that situation from our LIDAR data in Figure 6. As shown in Figure 6 (right), roads and their cross sections are not completely flat, and they may be curved. In [8, 9], ground points are calculated by fitting a plane using RANSAC algorithm. This method has the assumption that ground needs to be planar and has limitations if the road has curves or slope on one side. In [10], authors proposed a grid (cell) based method where ground is detected using average and variance of heights falling into each cell. We applied an approach similar to the method in [10] and developed a simple rule-based method to identify ground points accurately. Advantage of our method is

that we investigate each point in a grid cell separately because each cell may contain both ground and non-ground points.

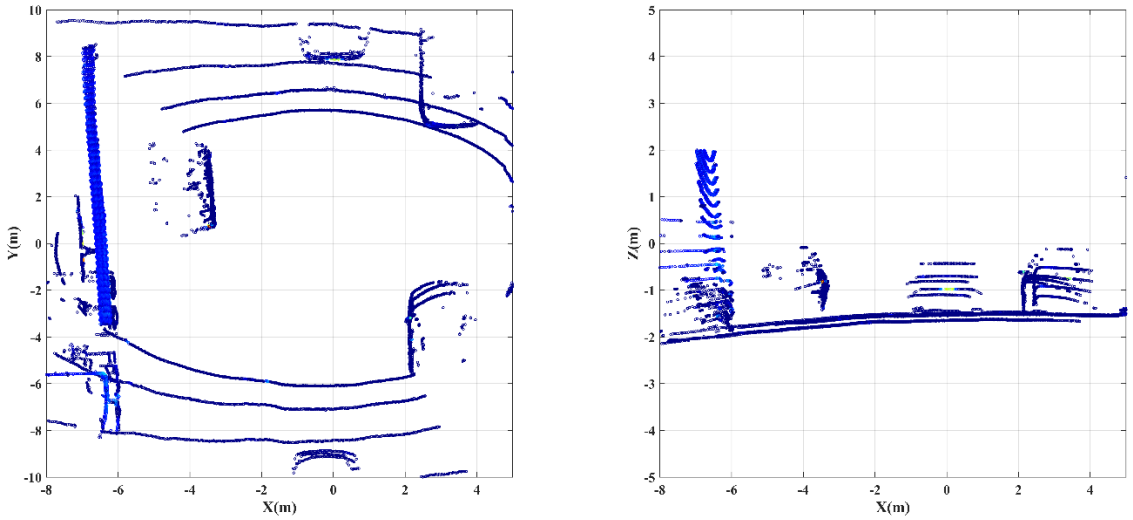

Figure 6. Lidar scan projected into 2D space. On the left projection to x, y axis and right to x, z axis.

We defined a region of interest of 40 meters long and 13 meters wide and vehicle detection-classification is done in this field. 3D LIDAR scan is projected onto x-y space and discretized into 0.25x0.25 meters grid cells which creates a grid with 52x160 cells. Minimum height in each cell was found, and any point with a height less than the minimum height in the cell plus a constant (α) is considered to belong to the ground. This condition is only applied where the minimum height in the cell is less than a constant (β). This procedure is applied on each cell. The result of this method on the same LIDAR scan in Figure 6 is given below in Figure 7. The red colors show the detected ground.

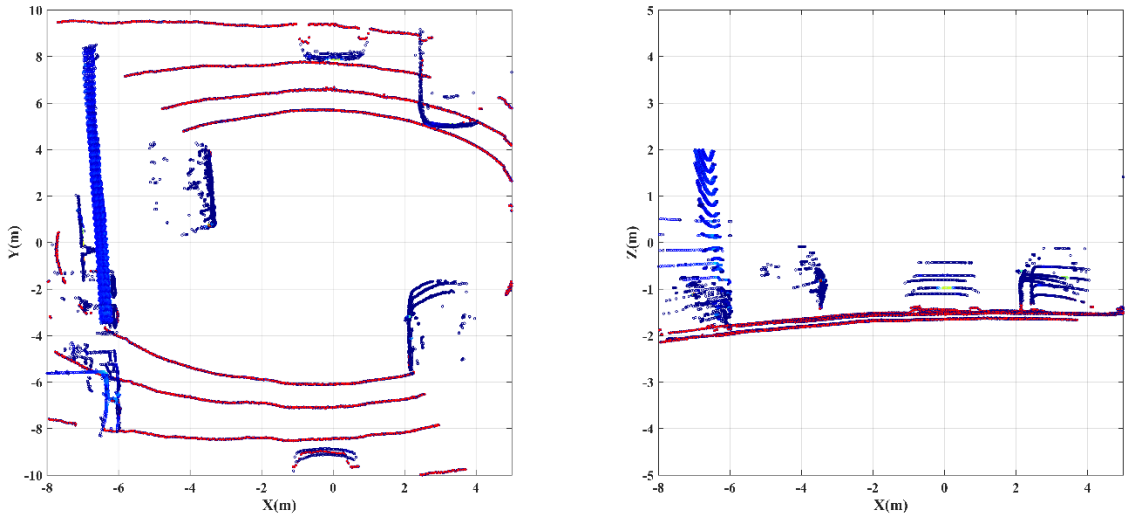

Figure 7. A sample result of ground detection

This ground detection algorithm can be expressed as:

$$p_k = \begin{cases} ground, & height(p_k) < \min\big(height(cell[i,j])\big) + \alpha \; and \; \min\big(height(cell[i,j])\big) < \beta \\ non-ground, & else \end{cases} \quad (1)$$

where p_k is the point in cell[i,j] of location i,j where 1≤i≤n and 1≤i≤m. α is ground height threshold and β is corrective height threshold to make sure parts of the vehicle is not considered as ground. m, n, α and β are selected as 160, 52, 0.25 and -1.35 respectively. Our proposed method allows differentiation between points in each cell. This ground detection algorithm is applied on 125 hand-labeled LIDAR scans and resulted in satisfactory results in terms of precision and recall measures. Precision gives a measure of how many of the selected points are relevant and recall shows how many of the relevant points are successfully selected.

$$precision = \frac{True\ Positive}{True\ Positive + False\ Negative} \quad (2)$$

$$recall = \frac{True\ Positive}{True\ Positive + False\ Positive} \quad (3)$$

Table 2. Precision and accuracy of ground detection

Precision	Recall
0.975	0.99

Target Detection and Classification:

After ground points are filtered out, the remaining sparse points belong to vehicles. We proposed a method to classify vehicles based on their dimensions. The collected 3D LIDAR data contains both trucks and passenger cars. In this research, a simple technique is applied to detect large trucks (e.g., FHWA Class 9) and more comprehensive vehicle classification is left for future research. Therefore, we classified detected vehicles as either a truck or a car. There are two main challenges to be addressed.

1- Clustering: 3D LIDAR points are sparse and points that belong to the same vehicle should be grouped together.
2- Unknown number of clusters: Due to the dynamic nature of traffic, number of vehicles in the region of interest is unknown.

Density-based spatial clustering of applications with noise (DBSCAN) algorithm is used to solve these problems [11]. It groups together points that are near each other. It can also identify points whose nearest neighbors are too far away and mark them as noise. DBSCAN is one of the most cited and popular clustering algorithm in the scientific literature.

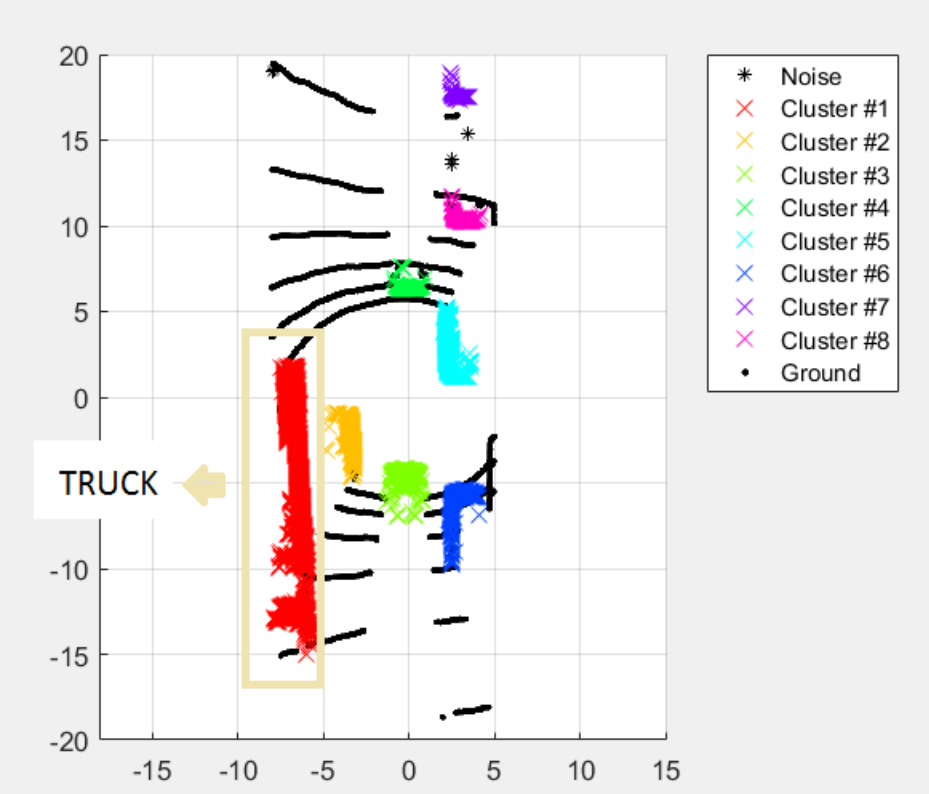

Figure 8. A sample result of clustering of cars and a truck with DBSCAN

This algorithm consists of two parameters. The first one is epsilon which is the diameter of the circle around any point called the neighborhood. The second one is the minimum number of points which can be considered as a cluster. For this study, the diameter is selected as 1.5 meters, and the minimum number of points is set to be 10 points. Figure 8 shows how the algorithm identified all clusters accurately. After this point, it is straight forward to distinguish between trucks and regular vehicles by looking at X-Z profile of each cluster. If the range of height in a cluster is more than 2 meters, it is classified as a truck. In Figure 8, cluster #1 (the red large cluster) is a truck, whereas all other clusters (cluster # 2-8) are cars.

All in all, we employed a two-step vehicle classification method. The first step removes ground points and second step creates clusters and classifies them. Vehicle classification is important because different types of vehicles have different traffic characteristics. Clusters from this part will be used for tracking in the next part of the project and trajectories of the vehicles will be recorded, knowing which trajectory belonging to which type will be useful in the analysis.

Target Tracking:

Target tracking is an essential part of our project. It allows to record vehicle trajectories and add new measurements to the existing trajectories. In this project, we used Kalman Filtering tracking with Hungarian Method for data association.

❖ **Kalman filtering for tracking:**

Kalman Filter has been a widely used method for guidance, control and tracking of vehicles [13, 14]. It consists of two parts: “Prediction” and “Update” [12]. Prediction part uses previous state for the current

state prediction. It doesn't include any observation information, so it is called "a priory" estimate. In the update part, current estimate is updated using the current observation, therefore, it is called "a posteriori" estimate.

State "s" at time "k" calculated by multiplying state transition model "F" at time "k" by state "s" at time "k-1" and adding process noise "w" at time "k".

$$s_k = F_k s_{k-1} + w_k \tag{4}$$

where process noise w_k is drawn from $N(0, Q_k)$ with zero mean and covariance Q_k

Observation "z" at time "k" is calculated by multiplying state "s" at time "k" by observation model "H" at time "k" and adding observation noise "v" at time "k".

$$z_k = H_k s_k + v_k \tag{5}$$

where observation noise v_k is drawn from $N(0, R_k)$ with zero mean and covariance R_k

"Prediction" step:

System state is predicted by

$$\hat{s}_{k/k-1} = F_k \hat{s}_{k-1/k-1} \tag{6}$$

Error covariance matrix is predicted by

$$P_{k/k-1} = F_k P_{k-1/k-1} F^T{}_k + Q_k \tag{7}$$

where "P" denotes covariance matrix which yields how accurate the state estimation is.

"Update" step:

Measurement innovation residual:

$$\tilde{y}_k = z_k - H_k \hat{s}_{k/k-1} \tag{8}$$

Innovation covariance:

$$C_k = R_k + H_k P_{k/k-1} H^T{}_k \tag{9}$$

Optimal Kalman gain:

$$K_k = P_{k/k-1} H^T{}_k C_k^{-1} \tag{10}$$

Updated state estimate:

$$\hat{x}_{k/k} = \hat{x}_{k/k-1} + K_k \tilde{y}_k \tag{11}$$

Updated estimate covariance:

$$P_{k/k} = (I - K_k H_k) P_{k/k-1} \tag{12}$$

In applying this filter to our problem of object tracking, we use data projected onto a 2D-space (x,y). Therefore, state s is given by $[x, y]^T$ where x is x coordinate, y is y coordinate. Observation z is the tracked object's measured location given by $[x, y]^T$

❖ **Multi-target tracking with Hungarian method:**

Frame matching is an important problem in target tracking. Once objects are detected and registered in a scene, next scene will provide new observations (objects). We need to determine which observed object in a new scene correspond to which object in the previous scene (see Figure 9).

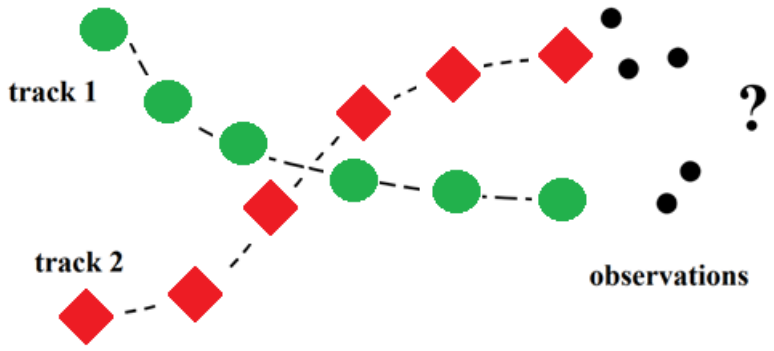

Figure 9. Data association problem

Hungarian method helps solve the assignment problem [15]. It seeks to minimize a global cost in the assignment problem. In our case, we can construct the cost matrix with Euclidian distances between last saved tracks and new measurements. Cost function for n number of tracks and m number of measurements is below

$$C_{i,j} = \begin{bmatrix} C_{1,1} & C_{1,2} & C_{1,3} & C_{1,4} & C_{1,5} & C_{1,6} & C_{1,7} & C_{1,m} \\ C_{2,1} & C_{2,2} & C_{2,3} & C_{2,4} & C_{2,5} & C_{2,6} & C_{2,7} & C_{2,m} \\ C_{3,1} & C_{3,2} & C_{3,3} & C_{3,4} & C_{3,5} & C_{3,6} & C_{3,7} & C_{3,m} \\ C_{4,1} & C_{4,2} & C_{4,3} & C_{4,4} & C_{4,5} & C_{4,6} & C_{4,7} & C_{4,m} \\ C_{n,1} & C_{n,2} & C_{n,3} & C_{n,4} & C_{n,5} & C_{n,6} & C_{n,7} & C_{n,m} \end{bmatrix} \tag{13}$$

where i and j are i^{th} track and j^{th} measurement respectively and 0<i<n and 0<j<m

In the data association part, we try to pair measurements and tracks in the best way such that it minimizes the global cost for a given assignment matrix $As_{i,j} \in \{0,1\}$ and cost function $C_{i,j}$

This can be written as:

$$\min_{As_{i,j}} \sum_{i=1}^{n} As_{i,j} C_{i,j} \tag{14}$$

subject to $\sum_{j=1}^{m} As_{i,j} = 1$, $\sum_{i=1}^{n} As_{i,j} = 1$ and $As_{i,j} = 0$ if and only if $C_{i,j} > r$ where r is distance threshold

Tracking results for 512 LIDAR scans or frames are given Figure 10 and 11 below. Vehicles travelling only in the same direction as our LIDAR-equipped vehicles are considered (forward direction). Data collection is from a two-lane urban road near Old Dominion University campus and 11 vehicles are tracked along our data collection path. In Figures 10 and 11, vertical axis shows the distance between data collection vehicle and target vehicles in the direction of movement. Horizontal axis shows the frame number at which the LIDAR recorded the vehicles. Our LIDAR was used in 10 Hz frequency which means it recorded 10 frames per second.

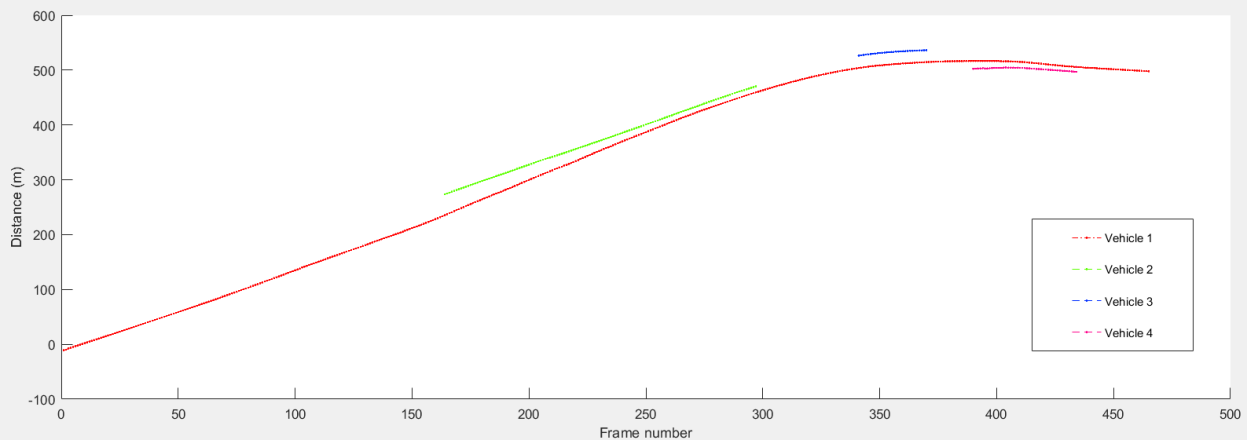

Figure 10. Vehicle trajectories for lane1

Figure 10 shows vehicle trajectories for lane 1 which is the same lane as our data collecting vehicle and it is left lane. Figure 11 below shows the second lane which is the right lane.

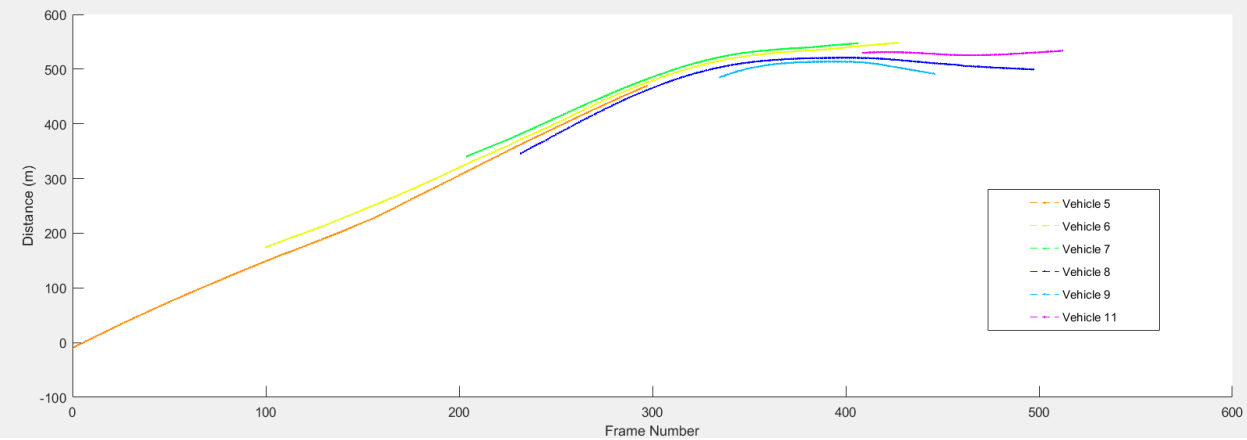

Figure 11. Vehicle trajectories for lane 2

We can see our proposed methods are able to construct vehicle trajectories around our data collection vehicle by using their detection location and our speed information. Vehicle trajectories provide valuable information about traffic and our methods proved to be successful in creating vehicle trajectories from 3D LIDAR point cloud data.

Microscopic and Macroscopic Traffic Parameters

One of the important goals of the project is to capture micro and macro states of the traffic. As the vehicle with LIDAR is moving within the traffic, it can capture and save the current traffic state by detecting vehicles around it. Below, we first present how LIDAR data could be used to construct and complete vehicle trajectories when there is missing observations. After that, we discuss how trajectory data from a sample of vehicles could be used estimate traffic density.

Constructing Vehicle Trajectories:

In this section, car following theories are used to generate complete trajectories and address potential missing data problems. Additional details about this method can be found in [16]. Trajectories collected with 3D Lidars may have gaps or missing points due to range limitations of the sensor where targets may go out of range and enter again. Microscopic traffic models are utilized to solve this problem. We calibrated four different car following models and fill missing trajectory data using the best calibrated models for each gap in the data.

- **Data:**

We used our collected 3D LIDAR data as well as NGSIM data to develop and test our methods. This way, we show that our model works for data from various sources and collected with different technologies.

LIDAR data acquisition and processing:

LIDAR data is collected at various traffic conditions with a platform consists of LIDAR (VLP-16), GPS and dash camera. Detected vehicle is considered as the leader vehicle and our vehicle is the follower vehicle for each trajectory pair. Some manual work is applied by looking at dash cam records to verify that lane change did not occur and the same car was followed for each trajectory.

NGSIM:

We also included NGSIM I-80 dataset [17] to test our missing data recovery model. NGSIM dataset includes an area about 500 meters long and 6 lanes on eastbound Interstate 80 in Emeryville, CA. in 2005. This dataset has served as an important benchmark dataset in traffic theory research. It consists of three parts each 15-minute long. We used the first 15 min part and used vehicle trajectories that are at least 50 seconds long without lane change and not on leftmost (HOV) and rightmost (ramp) lanes.

- **Approach:**

Complete LIDAR trajectories collected with the data collection methods mentioned in this report and NGSIM trajectory data are used to test our method. Random gaps are created deliberately within the trajectories which are then recovered by the algorithms. We assumed that driving behavior stays the same for 5 seconds, and gaps that occur for less than 5 seconds can be recovered with linear interpolation. Gaps longer than 5 seconds need more complicated solutions. We used our method to recover those gaps that

are more than 5 seconds long. Our method makes use of microscopic car following models for calibration of vehicle trajectories. These car following models help capturing driving behavior and leader-follower vehicle interactions. We calibrated different car following models using vehicle trajectories from our LIDAR data and NGSIM. Calibrated trajectories are compared with the actual trajectory values for the random created gaps. Later, we introduced an approach which we call "smooth transition algorithm" to fix un-matched trajectory end points with smooth transition along the gap.

❖ **Car following models:**

Car following models reside in microscopic traffic models and they use microscopic properties such as velocity, acceleration, deceleration and position. In this project, we studied Gipps', Intelligent Driving Model, Pipe's and Newell's car following models [18]. Brief background about these models are provided below.

Gipps' model uses reaction time and safety parameters such as "safe speed" and "safe distance" parameters to make sure collision is avoided. Its formula is given by

$$v(t+\Delta t) = \min(v + a\Delta t, v_0, v_{\text{safe}}(s, v_l)) \tag{15}$$

$$\mathrm{v} = \min(\mathrm{v}_0, \mathrm{v}_{\text{safe}}) \tag{16}$$

$$\mathrm{v} = \min(\mathrm{v}_0, \text{-b } \Delta\mathrm{t} + \sqrt{\mathrm{b}^2\Delta t^2 + v^2 + 2b(s - s_0)}) \tag{17}$$

where v: speed, v_0: desired speed, v_{safe}: safe speed, b: deceleration, Δt: reaction time, s: distance and s_0: minimum distance.

The Intelligent Driver Model (IDM) uses a smooth acceleration and deceleration policy and a smooth transition between them [19]. It is given by

$$\dot{\mathrm{v}} = a\left[1 - \left(\frac{v}{v_0}\right)^{\delta} - \left(\frac{s^*(v, \Delta v)}{s}\right)^2\right] \tag{18}$$

with the parameters speed v, speed change Δv, desired speed v0, acceleration exponent δ, acceleration a and desired distance s* and current distance s. Desired distance is given by

$$s^*(v, \Delta v) = s_0 + \max(0, vT + \frac{v\Delta v}{2\sqrt{ab}}) \tag{19}$$

where s_0: minimum distance, T: time gap, a: acceleration and b: deceleration. This term allows intelligent following distance.

Pipe's Model uses a "safe distance" of at least the length of a car for every ten miles per hour between vehicle pairs [20]. It is given by

$$x(n) = x(n+1) + b + L(n) + s_{\min}(n+1) \tag{20}$$

where x(n) and x(n+1) are the positions of the leader and follower vehicles respectively, b: standstill distance between the vehicles and L(n): length of the leader vehicle. s_{min}(n+1): the minimum distance between the vehicles given by

$$s_{\min}(n+1) = T.V(n+1) \tag{21}$$

where v(n + 1): speed of the follower vehicle and T: travel time of the minimum distance.

Newell model assumes leader and follower trajectories are linear translations of one another in time and space [21]. Newell's model assumes follower will keep a minimum distance with the leader under congested conditions.

$$x_n(t+\tau_n) = x_{n-1}(t) - d_n \tag{22}$$

where τn and dn are time and distance translations, respectively.

❖ **Calibration methodology:**

Optimum parameters for the car following models are found by solving an optimization problem. Genetic algorithm is used as the solving method. Genetic algorithms are widely used in solving constraint and unconstrained optimization problems for defined cost functions. We used a weighted cost function where points closer to the end points in the gaps have more effect on the overall cost. Tri-cube (bell shape) weight function (Eq. 24) is selected and sum of the difference between calibrated distance headway (from models) and known trajectory points headway values is multiplied with the corresponding weight based on the location around the gap. Cost function is given by

$$C = \sum_{i=1}^{N} w_i abs(s_{i,\mathrm{model}} - s_{i,trajectory}) \tag{23}$$

where N: total number of points around the gap, $s_{i,model}$: car following model headway at point I, $s_{i,trajectory}$: headway of known data point i and w_i is weight for point i. The weight for each point i is calculated by

$$w_i(d,L) = \begin{cases} (1-(d/L)^3)^3 & for\,(d/L) < 1 \\ 0 & for\,(d/L) \geq 1 \end{cases} \tag{24}$$

where d is the distance from the edge of the gap and L is the length of before and after gap areas. Figure. 12 shows these areas in detail.

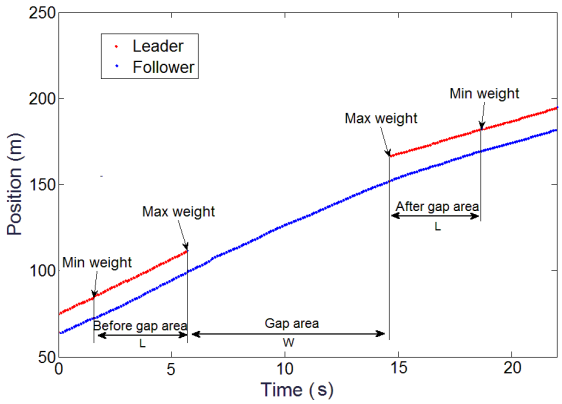

Figure 12. Trajectory gap and before-after areas

Max weight points contribute the most and points beyond the min weight points are disregarded in the method. Points between min and max weight points are weighted by the tri-cube weight function in Eq. 24. Figure 13 below shows the result of this locally weighted calibration. The discontinuity at the end of the calibrated gap can be seen.

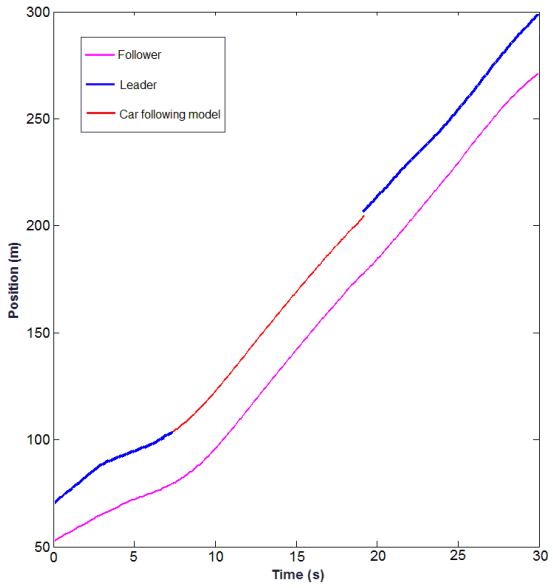

Figure 13. Result of calibration

The discontinuity in Figure 13 is expected as the car following models use the beginning of the gap as start point. We proposed a smooth transition algorithm to achieve continuous end points. Details of the algorithm are below.

❖ **Smooth transition algorithm:**

- **Step 1:** Straight lines from the ending edge of the gap are drawn to each estimated point starting from the closer points. We used a threshold value of slope (speed) and the first line to meet the requirement is selected. Several lines are drawn until a line meets the requirement. This allows a smooth transition at that part of the trajectory.

- **Step 2:** Reshaping operation is applied on the selected straight line from step 1. This operation uses a varying ratio between the car following model value and the straight-line value given in Eq 25.

$$y_{stm}[n] = w_n . y_{cfm}[n] + (1 - w_n) . y_{line}[n] \quad (25)$$

where $y_{stm}[n]$ is the smooth transition model value, $y_{cfm}[n]$ is car following model value, $y_{line}[n]$ is line value and w_n is the weight at point n.

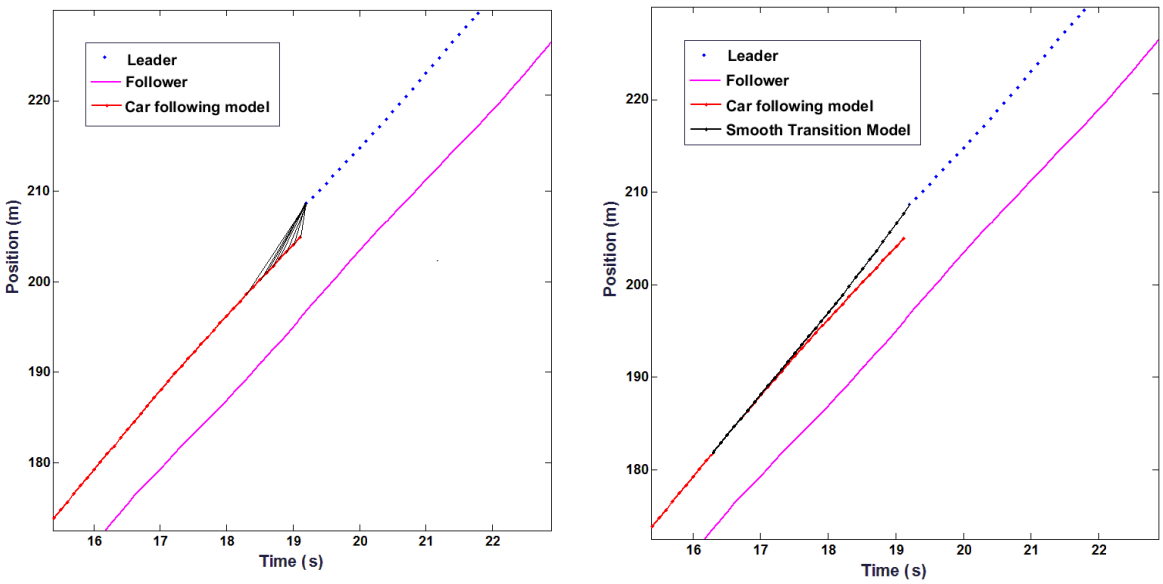

Figure 14. Straight lines drawn in the gap area in step 1 (left) and result of the algorithm (right)

Result of the algorithm for recovering a sample gap is given in Figure 14. This method is applied on the gaps created on our dataset. Gaps are created with random length between 5 and 15 seconds. We created 112 gaps in NGSIM and 24 gaps in the LIDAR data. We selected before and after gap areas (L) of 5 seconds on both sides of the gap. This allows to capture intra-driver behavior before and after the gap. Our optimization method is Genetic algorithm with roulette wheel selection, crossover rate 0.7, mutation rate 0.1, population 20 and 50 number of iterations. Root mean square error (RMSE) and mean absolute percent error (MAPE) of the joint datasets are given in Figure 15 below.

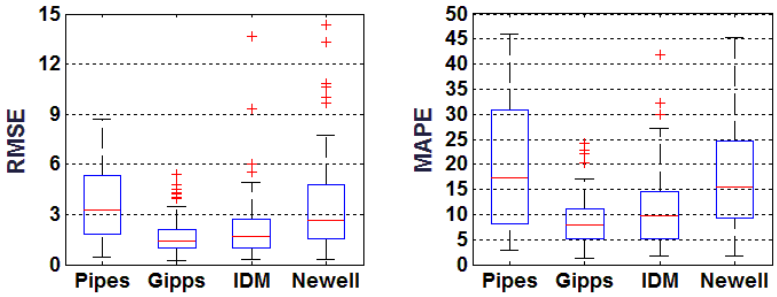

Figure 15. MAPE and RMSE (meters) for completed gaps in both LIDAR and NGSIM data

Table 3. Error Statistics for LIDAR and NGSIM Data

	Car following models							
	Pipes		***Gipps***		***IDM***		***Newell***	
	% MAPE	*RMSE*	% MAPE	*RMSE*	% MAPE	*RMSE*	% MAPE	*RMSE*
Min	2.87	0.48	1.35	0.22	1.78	0.34	1.63	0.29
Max	46.1	8.71	24.13	5.41	41.78	13.67	45.29	14.36
Average	20.27	3.63	8.95	1.75	11.21	2.26	17.79	3.65
Median	17.4	3.29	7.82	1.42	9.66	1.68	15.57	2.66
Standard diviation	13.37	2.12	5.36	1.15	7.58	2.04	10.51	2.99

Additional error statistics are summarized in Table 3. Gipps and IDM yield better results than other models for MAPE and RMSE measures. From Table 3, Gipps model has lower standard deviation than IDM, so it can provide more stable estimations. In this part, we investigated microscopic traffic models and estimated their parameters using a locally weighted calibration method. We applied this method on recovering missing trajectory data. Full trajectories are important as they provide valuable information about traffic state. As a future work, we plan to extend our method to work on multiple vehicle pairs around our data collection vehicle.

Estimating traffic density from trajectories:

Using the algorithms presented above, one can extract vehicle trajectories from LIDAR data. If a reasonably large number of vehicle are equipped, data from these vehicles could be integrated to estimate traffic density or other parameters for a given time-space region of interest. Since the LIDAR data collected in study come from a single vehicle, estimating density from such a limited data would not be possible. Instead, NGSIM data are employed to demonstrate how trajectories of some “probe” vehicles could be utilized to infer the number of unobserved vehicles or density. In particular, traffic density is estimated under congested traffic since vehicles interactions are more prevalent which makes the inference more feasible. As explained in [22], probe vehicle data under stop-go conditions can be extracted to predict the number of unobserved vehicles between them. For this purpose, NGSIM data trajectory data collected on I-80 and US-101 [17] are used. This dataset includes periods of heavy congestion with stop-and-go shockwaves. This characteristic is used to estimate number of unobserved vehicles between probe vehicles. A subset of the data is used to develop and train the model and the rest is used for testing. Training involves estimating mean and standard deviation for the probability density functions (pdfs) of gaps between probe vehicles. Queue length estimation at signalized intersections have been studied in the literature with statistical [23,24] or shockwave theory [25-28]. This work provides a simple approach and proves the theory that traffic flow parameters can be estimated using probe vehicle data with a good accuracy for stop-and-go traffic situations. This work will be extended using LIDAR data as a future project where probe vehicles and LIDAR equipped vehicle will be used together.

❖ **Data Processing:**

An example of trajectories is given in Figure 16. Red circles indicate the points at which vehicles stop and green circles indicate the points where vehicles start moving.

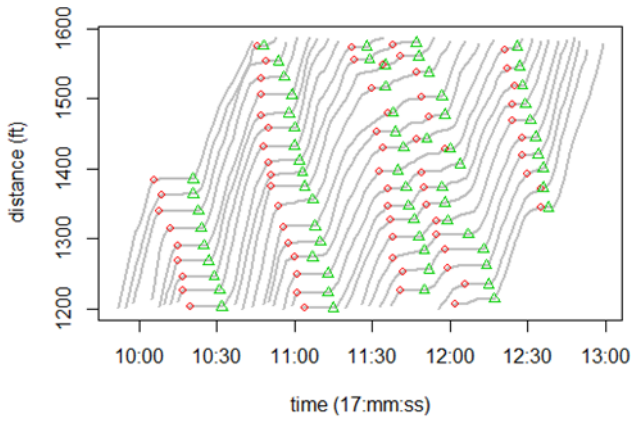

Figure 16. Sample trajectories and five stop-and-go shockwaves from lane 4 of I-80

Stop-and-go points are obtained using the following steps:

- **Step1:** For each trajectory, find points where vehicle speed is below a threshold (e.g., 0.1 fps) and label with a unique stopID. Add these stopIDs to a set $\mathcal{S}$.
- **Step2:** Find the stopID with the longest duration in $\mathcal{S}$ and define (t_s, x_s) and (t_e, x_e) as the start and end t-s coordinates of this stopID, respectively. Search between $(t_s - \tau, x_s)$ and $(t_e + \tau, x_e)$, where τ is buffer size (e.g., 5 sec).
- **Step3:** For the selected trajectories in Step 2, find the stop points ('go' points) where the vehicle decelerates (accelerates) the most.
- **Step4:** Remove extracted data in Step2 from $\mathcal{S}$ and go back to Step2 until $\mathcal{S}$ is empty.

Distance headway between any two vehicles is defined as,

$$x_{i,j}^k = x_i^k - x_j^k \quad \forall i, j \in I_k, i < j \tag{26}$$

In this project, the main problem is predicting the number of unobserved vehicles *n* between probe vehicles i and j based on the distance headway $x_{i,j}^k$, where *n* is defined simply as,

$$n = j - i \tag{27}$$

From the available dataset, 2004 distance different headways $x_{i,j}^k$ are extracted. A sample dataset is given in Table 4.

Table 4. Sample distance headways

Row #	n	$x_{i,j}^k$(ft.)	Road	Lane
1	1	21.572	US101	1
2	10	267.178	US101	2
3	4	100.706	I80	4
…	…	…	…	
2004	4	110.538	I80	5

❖ **Prediction Model and Training:**

In this study, distance headways between probe vehicles are assumed to be independent and follow a Gaussian probability density function (pdf) with parameters $\theta_n = (n\mu, n\sigma^2)$; and θ_1 represents the mean and variance of the Gaussian pdf when n = 1. A Naïve Bayes classifier is used to predict n given the distance headway.

$$\tilde{n} = \underset{m}{\operatorname{argmax}}\, f\left(x_{i,j}^k \middle| \theta_m\right) \qquad m = 1,2,\ldots \tag{28}$$

where, $\tilde{n}$ is the predicted *n*, *f* is the Gaussian pdf, $\theta_m = (m\mu, m\sigma^2)$

Unknown parameters mean and variance of the distance headway between two consecutive probe vehicles are determined with both a supervised and unsupervised method.

Supervised learning method:

In this method, it is assumed that distance headways from two consecutive probes are available. The main steps as applied to the data in Table 4 are as follows:

- Randomly select %50 of the data with n=1 from Table 4
- Estimate θ_1 (μ_1, σ_1^2) where $\mu_1 = avg(S)$ and $\sigma_1^2 = var(S)$

Unsupervised learning method:

For the unsupervised method, a subset of distance headways less than a predefined upper bound threshold (UB) are selected so that most of the observations with n ≤ 2 can be covered. The actual labels (n) need not be known in the unsupervised method. A Gaussian mixture model with two components is then fit to these data. UB is adjusted by the following steps until a stable value is found.

1. Initialize the upper bound (UB) for $x_{i,j}^k$.
2. Randomly select a given size of sample (e.g., 50%) from Table 4 where $x_{i,j}^k$ < UB.
3. A two-component Gaussian Mixture Model is fit with parameters $\theta_2 = 2\theta_1 = (2\mu_1, 2\,\sigma_1^2)$. A sample fit is shown in Figure 17.

4. The upper bound is updated to be three standard deviations larger than the second mean: $UB = 2\mu_1 + 3\sqrt{2\sigma_1^2}$
5. Steps 2-4 are repeated until UB converges. UB converges between 10-20 steps.

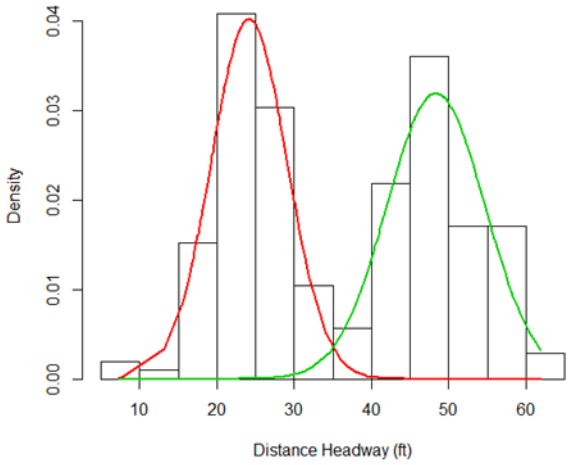

Figure 17. A two component Gaussian mixture model to fit the data

❖ **Results:**

Supervised and unsupervised methods are applied on the test data to predict unobserved number of vehicles between probe vehicles by estimating the two model parameters (μ_1, σ_1^2). Results are in Figure 18 below. Please note that data used for training is excluded in the graphs.

For the supervised method, 50% of all distance headways for which *n* = 1 are used for training and rest is used for testing. For *n* = 1, there are a total of 230 samples in the data. Since 115 of them are used for model training, they are not included in the top chart of Figure 18. Hence, the 115 samples on the first bar. For the unsupervised method, samples are selected based on the UB criterion as discussed before. Most of the random sample for training in this case comes from either *n* = 1 or *n* = 2 conditions while 11 data points (198-187=11) are from *n* = 3. The GMM model is then fit to this mixed sample of points to estimate the model parameters. The results shown in Figure 18 are for the remaining data points (i.e., training data not included). From Figure 18, we can see that supervised and unsupervised methods performed similar. Horizontal axes show the unobserved number of vehicles between probes. The number above each bar shows number of available data for testing corresponding to n. The numbers in white areas represent correct prediction, whereas numbers in gray areas represent under predictions (i.e., $\tilde{n} = n - 1$) or over predictions (i.e., $\tilde{n} = n + 1$). In very few instances, the over/under-predictions are by ±2, which are indicated by red shading. The Blue lines show prediction accuracy which is calculated by numbers in white area divided by the corresponding sample size (numbers above bars).

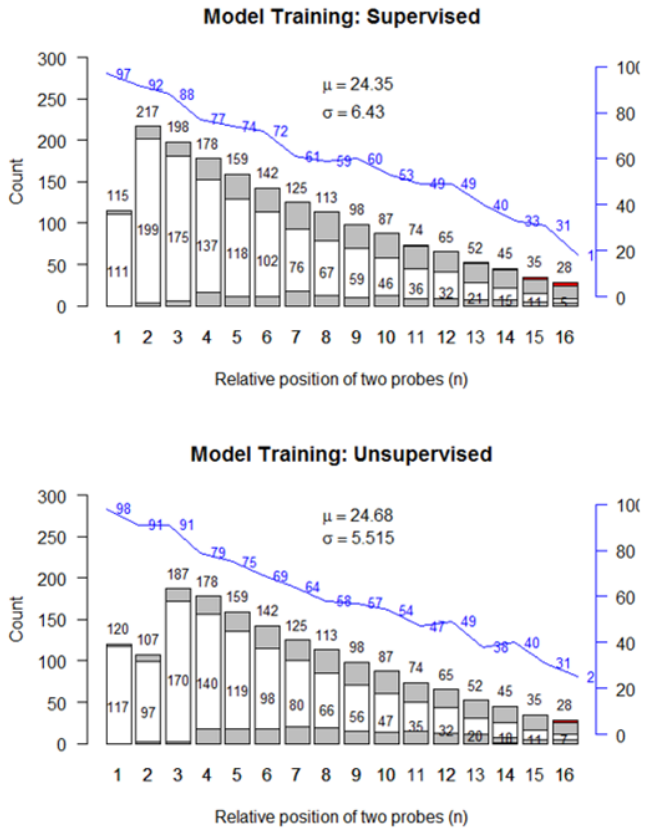

Figure 18. Accuracy of the supervised and unsupervised learning methods applied to the distance headway data.

❖ **Estimating macroscopic traffic density:**

Traffic density is calculated by dividing the prediction $\tilde{n}$ by headway distance $x_{i,j}^{k}$ which is $\tilde{n}/x_{i,j}^{k}$. Accuracy of this approach is computed by the following error measure:

$$\varepsilon = \frac{\left(\frac{\tilde{n}}{x} - \frac{n}{x}\right)}{\frac{n}{x}} * 100 \tag{29}$$

Where subscripts are omitted for simplicity. This equation can be written in a simpler way by

$$\varepsilon = \frac{\tilde{n} - n}{n} * 100 \tag{30}$$

This error measure is used to calculate the accuracy of the traffic density calculation. Since we used a random a selection for training and testing. Each model is trained and tested 30 times to account for variations. Figure 19 show the result of these 30 runs: the average estimated, actual densities and the standard deviation ε. The blue line shows the standard deviation of ε, the green and red ones show the actual and estimated average densities, respectively. The macroscopic density is expected to be high due to the data coming from stop-and-go or dense traffic conditions and Figure 19 shows the same result except the first bin where headway is less than 20 ft. Standard deviation of error for the first bin is zero since unrealistic small headways when x<20 ft results in both n and $\tilde{n}$ to be 1. For headways larger than

20ft, standard deviation of error starts high and decreases as headway or n increases. While accuracy of predicting n decreases with increasing n in Figure 18.

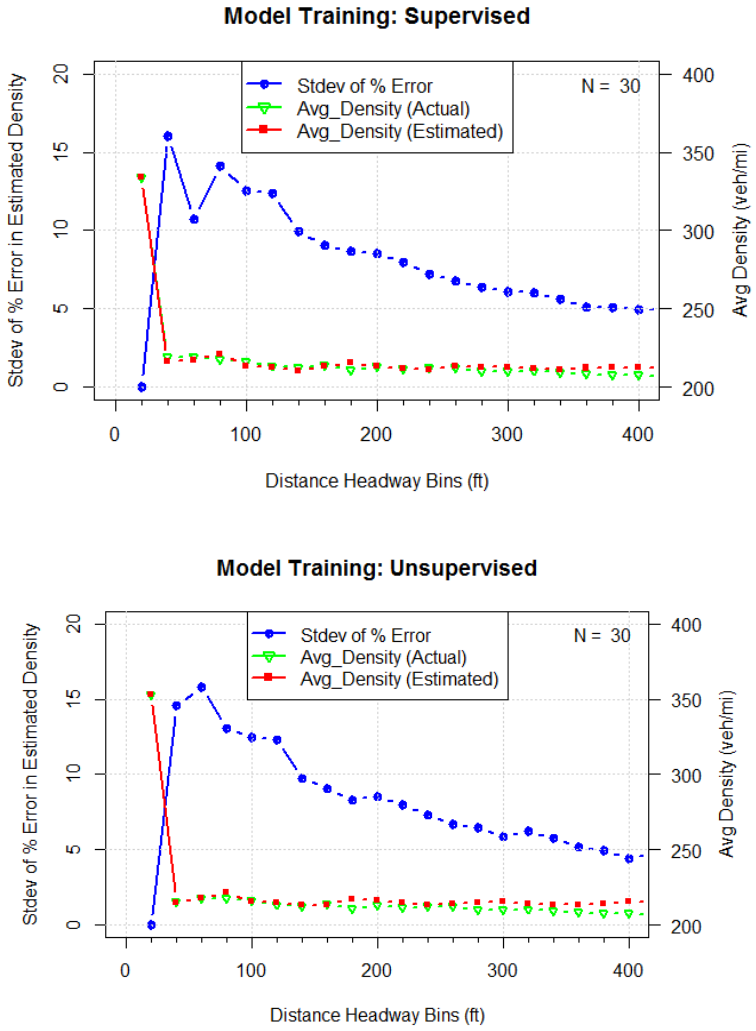

Figure 19. Accuracy for supervised and unsupervised methods. The results of 30 trials are aggregated by distance headways bins in increments of 20 ft.

In this study, number of vehicles between probe vehicles are estimated using distance information between probe vehicles. Naïve Bayes model with Gaussian pdfs are used and their parameters are estimated with a supervised and unsupervised method. These methods are tested with NGSIM trajectory data and resulted in good prediction results with accuracy of ±1 vehicle almost always. Later, predicted number is used to calculate traffic density by the equation 30. From Figure 19, we can see that models can be used to calculate macroscopic traffic density. Accuracy increases as number of vehicles or distance headway increases. This work proposes two different methods for traffic flow parameter estimation from probe vehicle trajectory. This work will be extended using LIDAR data.

6. Conclusions:

In this project, authors developed models and algorithms for traffic flow parameters estimation from 3D LIDAR data problem. Authors addressed the research objectives of the project: LIDAR data is collected under different conditions on urban and freeway roads, vehicles are classified based on their sizes, vehicles are tracked and their trajectories are extracted and microscopic and macroscopic traffic flow parameters are calculated from vehicle trajectories. We successfully applied our methods on the data we collected with our vehicle on urban and freeway roads. Detected vehicles are classified based on their geometry and then they are processed by a tracking system consisting of Kalman Filter with Hungarian Algorithm for data association. Using the tracking information, we constructed vehicle trajectories which have valuable information about the traffic state along the data collection path. Then, collected vehicle trajectories are used in a car-following models to predict missing data points in trajectories. This method gave very satisfactory results when applied on known trajectories from LIDAR and NGSIM datasets. It is also demonstrated that traffic density can be estimated accurately under congested conditions using a small sample of probe vehicles in the traffic stream.

7. References:

[1] J. C. Herrera, D. Work, R. Herring, J. Ban, Q. Jacobson, and A. Bayen. “Evaluation of traffic data obtained via GPS-enabled mobile phones: the Mobile Century experiment,” Transportation Research Part C, 2009

[2] M. Brackstone and M. McDonald, "Dynamic Behavioural Data Collection Using an Instrumented Vehicle," Transp. Res. Rec., vol. 1689, pp. 9-17, 1999.

[3] Goodall, N., B. Smith, and B. Park. “Microscopic Estimation of Freeway Vehicle Positions Using Mobile Sensors”, In TRB 91st Annual Meeting. 2011.

[4] http://velodynelidar.com/

[5] D. Pfeiffer, U. Franke, Efficient representation of traffic scenes by means of dynamic stixels, in: Intelligent Vehicles Symposium (IV), 2010 IEEE, pp. 217–224, 2010

[6] A. Broggi, S. Cattani, M. Patander, M. Sabbatelli, P. Zani, A full-3d voxel-based dynamic obstacle detection for urban scenario using stereo vision, in: Intelligent Transportation Systems - (ITSC), 2013 16th International IEEE Conference on, pp. 71–76, 2013

[7] A. Azim, O. Aycard, Layer-based supervised classification of moving objects in outdoor dynamic environment using 3d laser scanner, in: Intelligent Vehicles Symposium Proceedings, 2014 IEEE, pp. 1408– 1414, 2014

[8] M. Oliveira, V. Santos, A. Sappa, P.Dias, Scene representations for autonomous driving: an approach based on polygonal primitives, in: 2nd Iberian Robotics Conference, 2015

[9] F. Oniga, S. Nedevschi, Processing dense stereo data using elevation maps: Road surface, traffic isle, and obstacle detection, Vehicular Technology, IEEE Transactions on 59 (3) (2010) 1172–1182.

[10] A. Asvadi, P. Peixoto, U. Nunes, Detection and tracking of moving objects using 2.5d motion grids, in: Intelligent Transportation Systems (ITSC), 2015 IEEE 18th International Conference on, 2015, pp. 788–793.

[11] M. Ester, H. Kriegel, J. Sander, and X. Xu, “A density-based algorithm for discovering clusters in large spatial databases with noise,” in Proc. 2nd Int. Conf. Knowledge Discovery and Data Mining (KDD’96), pp. 226–231, 1996

[12] R. E. Kalman. “A new approach to linear filtering and prediction problems” In Trans. ASMEVJ. Basic Eng, pages ser.D, vol. 82, pp.35-45, 1960
[13] Z. Luo, S. Habibi and M. V. Mohrenschildt, “LiDAR Based Real Time Multiple Vehicle Detection and Tracking”, International Journal of Computer, Electrical, Automation, Control and Information Engineering, vol. 10, pp. 1125-1132, 2016
[14] A. Macaveiu, A. Campeanu, and I. Nafornita, “Kalman-based tracker for multiple radar targets” in COMM 2014 International Conference on Communications, Bucharest, Romania, 2014.
[15] H. W. Kuhn, “The Hungarian method for the assignment problem”, Naval Research Logistics, Vol. 52, pp. 7-21, 2005.
[16] C. Sazara, R. V. Nezafat, M. Cetin, “Offline reconstruction of missing vehicle trajectory data from 3D LIDAR”, Intelligent Vehicles Symposium (IV), 2017 IEEE, pp. 792-797, 2017
[17] FHWA. (2017, 9/20/2017). Next Generation Simulation (NGSIM). Available: https://ops.fhwa.dot.gov/trafficanalysistools/ngsim.htm
[18] M. Treiber and A. Kesting, Traffic flow dynamics: data, models and simulation. Heidelberg; New York: Springer, 2013.
[19] D. H. M. Treiber, “Congested traffic states in empirical observations and microscopic simulations,” Phys Rev. E 62, 1805, 2000.
[20] Pipes LA, “An operational analysis of traffic dynamics” J Appl Phys 24(3): pp. 274–281, 1953
[21] Newell, G., A simplified car-following theory: a lower order model. Transportation Research Part B, 36, 195–205, 2002
[22] M. Cetin and K.A. Anuar, “Using Probe Vehicle Trajectories in Stop-and-Go Waves for Inferring Unobserved Vehicles” 5th IEEE International Conference on Models and Technologies for ITS, Naples, Italy, June 26-28 2017.
[23] G. Comert and M. Cetin, "Queue length estimation from probe vehicle location and the impacts of sample size," European Journal of Operational Research, vol. 197, pp. 196-202, 2009.
[24] G. Comert and M. Cetin, "Analytical evaluation of the error in queue length estimation at traffic signals from probe vehicledData," IEEE Transactions on Intelligent Transportation Systems, vol. 12, pp. 563-573, 2011.
[25] J. Anderson, B. Ran, J. Jin, X. Qin, and Y. Cheng, "Cycle-by-Cycle Queue Length Estimation for Signalized Intersections Using Sampled Trajectory Data," Transportation Research Record: Journal of the Transportation Research Board, vol. 2257, pp. 87-94, 2011.
[26] Q. Cai, Z. Wang, L. Zheng, B. Wu, and Y. Wang, "Shock Wave Approach for Estimating Queue Length at Signalized Intersections by Fusing Data from Point and Mobile Sensors," Transportation Research Record: Journal of the Transportation Research Board, vol. 2422, pp. 79-87, 2014.
[27] M. Cetin, "Estimating queue dynamics at signalized intersections from probe vehicle data," Transportation Research Record: Journal of the Transportation Research Board, vol. 2315, pp. 164-172, 2012.
[28] S. Y. R. Rompis, M. Cetin, and F. Habtemichael, "Probe Vehicle Lane Identification for Queue Length Estimation at Intersections," Journal of Intelligent Transportation Systems, pp. 0-0, 2017.