FINAL REPORT

Submitted to
THE FLORIDA DEPARTMENT OF TRANSPORTATION
On Project

Extended Development and Testing of Optimized Signal Control with Autonomous and Connected Vehicles

FDOT Contract BDV31-977-109

Submitted by

Lily Elefteriadou, Ph.D.

Sanjay Ranka, Ph.D.

Carl Crane, Ph.D.

Luan Staichak Carvalho

Patrick Emami

Christopher Mauldin

Pruthvi Manjunatha, Ph.D.

September 14, 2021

UF | Transportation Institute
UNIVERSITY of FLORIDA

DISCLAIMER

The opinions, findings, and conclusions expressed in this publication are those of the authors and not necessarily those of the State of Florida Department of Transportation.

METRIC CONVERSION CHART

U.S. UNITS TO METRIC (SI) UNITS

LENGTH

SYMBOL	WHEN YOU KNOW	MULTIPLY BY	TO FIND	SYMBOL
in	inches	25.4	millimeters	mm
ft	feet	0.305	meters	m
yd	yards	0.914	meters	m
mi	miles	1.61	kilometers	km

METRIC (SI) UNITS TO U.S. UNITS

LENGTH

SYMBOL	WHEN YOU KNOW	MULTIPLY BY	TO FIND	SYMBOL
mm	millimeters	0.039	inches	in
m	meters	3.28	feet	ft
m	meters	1.09	yards	yd
km	kilometers	0.621	miles	mi

TECHNICAL REPORT DOCUMENTATION PAGE

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
4. Title and Subtitle Extended Development and Testing of Optimized Signal Control with Autonomous and Connected Vehicles		5. Report Date September 14, 2021 6. Performing Organization Code
7. Author(s) Lily Elefteriadou, Sanjay Ranka, Carl Crane, Luan Carvalho Staichack, Patrick Emami, Christopher Mauldin, Pruthvi Manjunatha		8. Performing Organization Report No.
9. Performing Organization Name and Address University of Florida Transportation Institute University of Florida 365 Weil Hall, PO Box 116580 Gainesville, FL 32611-6580		10. Work Unit No. (TRAIS) 11. Contract or Grant No. FDOT Contract BDV31-977-109
12. Sponsoring Agency Name and Address Florida Department of Transportation 605 Suwannee Street, MS 30 Tallahassee, FL 32399		13. Type of Report and Period Covered Final 14. Sponsoring Agency Code
15. Supplementary Notes		

16. Abstract

Previous work by the University of Florida and FDOT focused on the development of algorithms, software, and hardware solutions to enhance traffic signal control operations simultaneously with vehicle trajectories. These tests focused on the integration of the technology in a mixed traffic stream of autonomous, connected, and conventional vehicles. A natural extension of this work is to consider pedestrian movements and the fusion of additional existing sensors to refine the algorithm performance and further prepare it for field implementation. This report discusses the development of sensor fusion and LiDAR detection for pedestrians at a signalized intersection, and a new approach to data sharing between the infrastructure and the autonomous vehicle to maximize safety based on increased information regarding the surrounding environment. It also presents the procedure developed to accommodate pedestrians within the signal control optimization environment and the field testing conducted at FDOT's Traffic Engineering and Research Lab (TERL) to evaluate the hardware and software system. The scenarios tested in the field provided feedback on how the system developed responds to different traffic conditions, and showed that it can serve both vehicle and pedestrian demand in all cases evaluated. It was observed that the detection method affects the timing of when a vehicle is detected, and therefore DSRC can detect vehicles from a longer distance than video. Therefore, connected vehicles may be able to place a call to the controller and get the right-of-way sooner. As a next step, this algorithm could be implemented at one or more signalized intersections for further evaluation and refinement.

17. Key Words	18. Distribution Statement
Signal control optimization; autonomous and connected vehicles	No restrictions

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price
Unclassified	Unclassified	102	

EXECUTIVE SUMMARY

Previous work by the University of Florida and FDOT focused on the development of algorithms, software, and hardware solutions to enhance traffic signal control operations simultaneously with vehicle trajectories. The Real-time Intersection Optimizer (RIO) is an algorithm developed by the UFTI to optimize Signal Phase and Timing (SigPT) and vehicle trajectories. However, the original version of the algorithm does not serve pedestrian demand, nor does it optimize vehicular throughput considering pedestrian presence. A natural extension of this work is to consider pedestrian movements and the fusion of additional existing sensors to refine the algorithm performance and further prepare it for field implementation.

There were four primary objectives for this project:

1. Extend the scope of the previously developed traffic intersection sensor fusion system to incorporate pedestrians, bicyclists, and scooters. This system, developed in previous research, was not able to process these vulnerable road users. Adding the ability to account for these users using LiDAR allows the traffic signal optimization system to more accurately account for all vulnerable road users.

2. Investigate how the safety and efficiency of the connected and autonomous vehicles (CAV) can be enhanced as they navigate in the vicinity of an intersection and within a realistic environment which includes pedestrians, bicycles, and scooters. This project developed a framework for communicating between the intersection controller (IC) and the CAV, and involves both static data (for example, intersection design and pole locations) and dynamic data (for example, optimal trajectories and the location of pedestrians).

3. Extend RIO to use real-time anonymous tracking information of vehicles and consider the presence of pedestrians and other modes when optimizing signal control.

4. Implement the extended optimization algorithm at FDOT's Traffic Engineering and Research Laboratory (TERL) facilities and evaluate its operation with a mix of vehicles and pedestrians.

The research team designed, implemented, and integrated a LiDAR-based pedestrian detection system into RIO. However, the team was unable to test the system with bicycles and scooters due to an inability to collect suitable data because of the COVID-19 pandemic. Nevertheless, the method developed is general enough to include additional road users in the future. Overall, the performance of this detection system was found to be satisfactory. Pedestrians standing near the LiDAR at the intersection corner were reliably detected automatically, and the information was entered into the signal control optimizer.

A message structure was designed to allow for source-independent communication of detected dynamic objects. Testing of this message structure showed low latency for a single detected object. It was concluded that further testing is required to determine the scalability of

the message in this respect. In cases where the dynamic object is being observed by both the intersection and the CAV, the large error between the two position measurements suggests incoming messages may not be useable for course planning by the CAV.

The research team added pedestrian detection to RIO, along with pedestrian phasing capabilities. The LiDAR algorithm can detect and send the counts of pedestrians to the signal control optimizer with a delay of roughly 2 seconds. RIO then uses these counts to assign suitable signal phasing and times to pedestrians.

Testing at the TERL provided field data regarding the response of the signal control optimizer to different traffic conditions. It was concluded that the system can serve both vehicle and pedestrian demand in all cases evaluated. It was also observed that RIO can provide reasonable signal phasing and timing even when detection errors occur, such as when the video feed provides false positives or duplicates of vehicles. The research team also observed that the detection method affects the timing of when a vehicle is detected: Dedicated Short-Range Communications (DSRC) can detect vehicles at a longer distance. As a next step, this algorithm could be implemented at one or more signalized intersections for further evaluation and refinement. In order to deploy this system in the field, suitable detection methods are required for all approaches in order to identify vehicles, pedestrians, and other users.

Implementation Requirements

In order to implement this system, the following detection components should ideally be installed:

- Video-cameras with line of sight of each approach
- LiDARs for pedestrian detection (pedestrian push buttons can provide pedestrian input for waiting areas outside of LiDAR coverage)
- Road Side Unit (RSU)/DSRC for providing signalization information to connected vehicles.

Figure 1 shows a schematic of the system's components and data flow. All data processing occurs at the intersection computer, which is located near or in the cabinet. The LiDAR readings for pedestrians are processed on a Intel NUC 6 that connects wirelessly to the Intelligent Intersection Controller (IIC) which allows the LiDAR to be deployed at a distance from the signal cabinet. The IIC processes the vehicle inputs and generates recommended vehicle trajectories and the signalization plan.

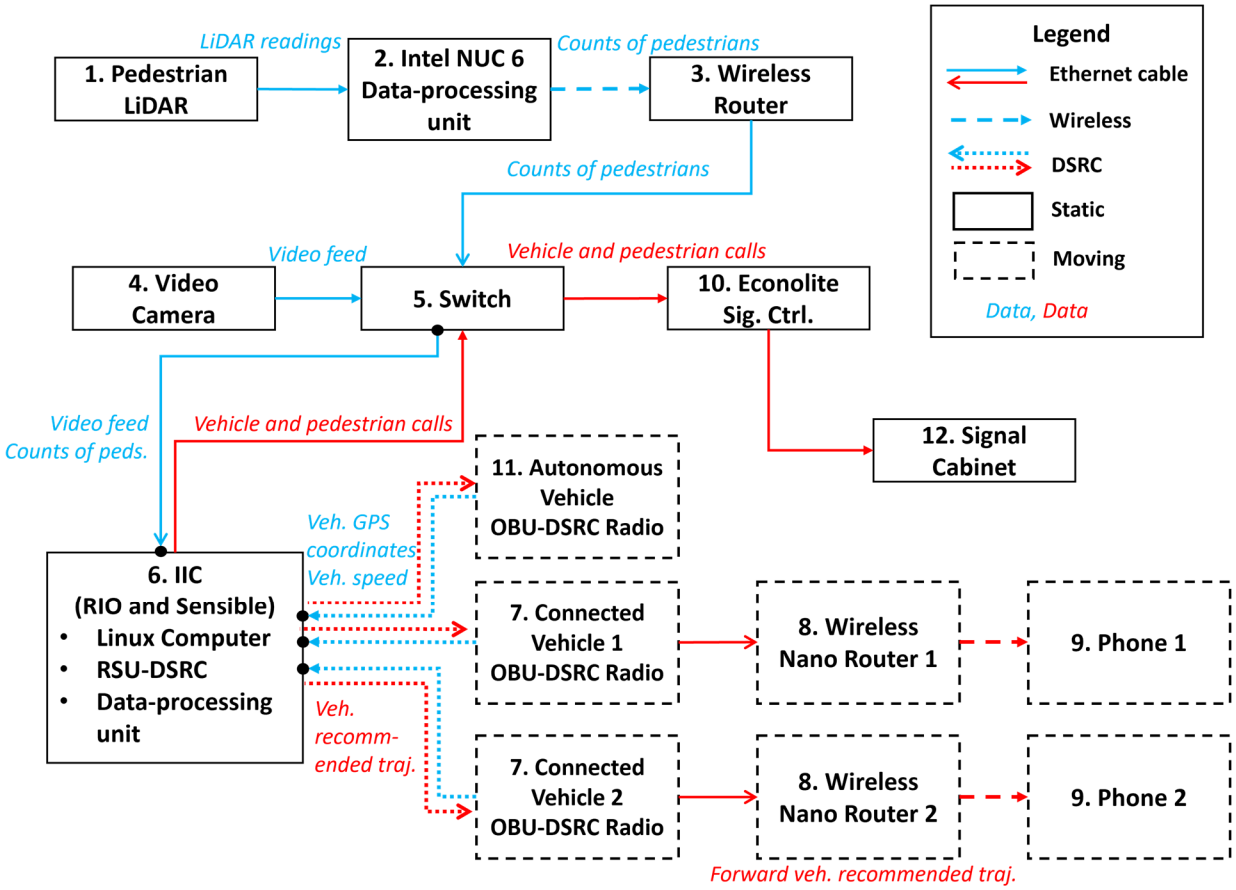


Figure 1: System components and data flow.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

1.1 Background

Vehicle automation has long been a part of production vehicles, and the level of automation has increased with time. Auto manufacturers have included some aspect of vehicle automation in production vehicles for decades, starting with anti-lock brake systems, cruise control systems, and automatic transmissions. Production vehicles are now available with automated parking, lane keeping, adaptive cruise control, and automatic emergency braking. Fully autonomous vehicles (AVs) may soon be available to consumers for retail purchase. It is unknown when a significant market share of AVs will be operating on our highways, but it is predicted that AV sales will total 21 million globally by the year 2035.

Connected vehicles (CNVs) can communicate with surrounding vehicles and infrastructure using wireless communication. The USDOT CNV research program aims to enable wireless communications among vehicles, infrastructure, and personal communication devices (https://www.its.dot.gov/pilots/). The program is currently focusing on pilot deployments to implement existing research concepts and encourage further innovation. CNV technology seeks to warn drivers of impending dangers while the vehicle is still controlled by a human driver. AVs may also have connectivity capabilities; such vehicles are called connected and automated vehicles (CAVs). These types of vehicles would provide the most improvements in safety and mobility as they would provide additional information that is not easily obtainable by video and LiDAR sensors on the vehicle (for example, intent of other vehicles, pedestrians and bicycles, and potential obstacles that are not within the line of sight of the vehicle).

With the advent of AVs, CNVs, and CAVs, traffic management strategies and infrastructure must adapt to accommodate evolving technology. While it is possible for autonomous vehicles to operate in the current system, it is advantageous to utilize this technology to improve traffic operations and safety. Conflict areas such as signalized intersections are primary sources of traffic congestion and incidents. Receiving real-time data about all road users can allow intelligent traffic management systems to process and update traffic control to ensure the safety of pedestrians and drivers. These recommendations may include signal timing, speed recommendations, in-vehicle warnings, or microscopic vehicle trajectories.

It is likely that soon both connected and automated vehicles will operate side-by-side in large numbers in our nation’s roads, along with conventional vehicles and pedestrians. This creates many opportunities in improving surface transportation efficiency and safety. For example, the USDOT Multimodal Intelligent Traffic Signal Systems initiative aims to provide a comprehensive traffic information framework to service all modes of transportation, including passenger vehicles, transit, emergency vehicles, freight fleets, and pedestrians and bicyclists in a CNV environment.

Previous work by the University of Florida and FDOT focused on the development of algorithms, software, and hardware solutions to enhance traffic signal control operations simultaneously with vehicle trajectories. The Real-time Intersection Optimizer (RIO) is an algorithm developed by the UFTI to optimize Signal Phase and Timing (SigPT) and vehicle trajectories. The original version of RIO receives real-time positioning of CNVs or CAVs, determines SigPT in order to better serve the current demand in the intersection, and sends back optimized trajectories such that vehicles depart from the stop bar at the saturation rate and at the maximum possible speed. However, the original version of the algorithm does not serve pedestrian demand, nor does it optimize vehicular throughput considering pedestrian presence. A natural extension of this work is to consider pedestrian movements and the fusion of additional existing sensors to refine the algorithm performance and further prepare it for field implementation.

1.2 Project objectives

There are four primary objectives for this project:

1. Extend the scope of the previously developed traffic intersection sensor fusion system to incorporate pedestrians, bicyclists, and scooters. This system, developed in our previous research, is not able to process these vulnerable road users. Adding the ability to account for these users using LiDAR allows the traffic signal optimization system to more accurately account for all vulnerable road users. This project develops robust algorithms that allow us to recognize all modes of transport at the intersection.

2. Investigate how the safety and efficiency of the CAV can be enhanced as it navigates in the vicinity of the intersection and within a realistic environment which includes pedestrians, bicycles, and scooters. The research team developed a framework for communicating between the intersection controller (IC) and the CAV, and involves both static data (for example, intersection design and pole locations) and dynamic data (for example optimal trajectories and the location of pedestrians).

3. Extend the optimization algorithm (i.e., the intersection manager) to use real-time anonymous tracking information of vehicles and consider the presence of pedestrians and other modes when optimizing signal control.

4. Implement the extended optimization algorithm at FDOT's TERL laboratory facilities and evaluate its operation with a mix of vehicles as well as pedestrians, bicycles, and scooters.

1.3 Report organization

The second chapter discusses the use of sensor fusion and LiDAR detection for pedestrians at a signalized intersection, while the third chapter discusses data sharing between the infrastructure and the autonomous vehicle to maximize safety based on increased information regarding the surrounding environment. The fourth chapter presents the procedure developed to accommodate

pedestrians within the signal control optimization environment. The fifth chapter summarizes the field testing conducted at the TERL to evaluate the hardware and software system. The last chapter provides an overview of the research along with conclusions and recommendations.

2 MULTIMODAL SENSOR FUSION FOR DETECTION OF VEHICLES AND PEDESTRIANS

2.1 Overview

The objective of this part of the research is to develop robust algorithms that allow us to identify pedestrians at traffic intersections. This system will produce localization information in real-time that can be used by our signal and trajectory optimization algorithm to make predictions that increase the safety and efficiency of traffic management at intersections.

To accommodate the identification of all types of road users, we have introduced a novel algorithm for fusing video data with other sensor modalities. We use the Velodyne and Ouster LiDAR sensors to integrate the detection of pedestrians into our multi-sensor framework. We have access to one Velodyne LiDAR that the FDOT has purchased for another project conducted by UF (this equipment has been transferred from BDV31-977-57 and was recently sent to the manufacturer for repair). We have also purchased an Ouster OS-1 sensor for a different project that we can use for additional data collection. These sensors are built for short-range, high-precision detection and tracking of pedestrians and vehicles. The data extracted from a LiDAR can be used in real-time with video, radar, and vehicle-to-infrastructure communication data to produce a comprehensive picture of the surrounding environment.

Most vision-based traffic surveillance systems are highly adept at either only pedestrian detection or vehicle detection, and most struggle with occlusion, changes in target appearance, depth estimation, and poor weather conditions, or are too inefficient to run on the edge in real-time. To address these challenges, we are leveraging tried-and-true LiDAR classification algorithms that are highly efficient for real-time detection to supplement our state-of-the-art real-time deep learning-based vision and radar fusion tracking framework. To ensure our LiDAR detector is highly accurate, we prioritize the collection and annotation of a large dataset of pedestrians in-the-wild to train the detector. This is a crucial contribution of the proposed research, as there does not yet exist freely available labeled datasets for LiDAR pedestrian detection at urban traffic intersections. A proof-of-concept implementation of the LiDAR pedestrian detector will be integrated into our multi-sensor fusion system and deployed in our intelligent intersection control system.

In summary, the key objectives of this part of the project are:

- Design and implement a real-time and accurate pedestrian detection system for LiDAR
- Design and implement the integration of the LiDAR into the larger multi-sensor fusion framework for the intelligent intersection control system.
- Collect and annotate pedestrian examples in unstructured LiDAR data from a traffic intersection for detector training and validation.
- Analyze the performance of the detector on our collected dataset to characterize strengths and weaknesses.

2.2 Literature review

A reliable pipeline for real-time LiDAR processing breaks down the problem into multiple steps. First, unnecessary points belonging to flat planes such as the ground can be removed. Then, the remaining points can be clustered into a list of candidate objects. Finally, detection and tracking of the relevant classes of objects can be efficiently handled at the level of clustered objects.

The most closely related project to ours in the literature is "Detection and tracking of pedestrians and vehicles using roadside LiDAR sensors" (Zhao et al., 2019). This paper includes a comprehensive analysis of a project where a LiDAR is positioned at the roadside and simple road user classification is performed. Similar to our approach, they use a multi-stage processing pipeline to convert the problem into that of classifying point cloud object clusters. They use a simple three neural/one-layer multi-layer perceptron classifier for their method with hand-crafted features extracted from the point cloud. This classification strategy achieves a similar level of accuracy to the one considered in this work.

2.2.1 Ground removal

Point clouds that belong to the ground surface constitute most of the point cloud and their removal significantly reduces the number of points involved in the subsequent computations. Typically, this step can be addressed with geometric methods. A simple approach considers the maximum absolute difference between point heights in a grid representation of the environment (Thrun et al., 2006). Scan line segmentation is leveraged by (Wang and Zhang, 2016) to identify flat planes from airborne LiDAR point cloud data. The most robust approaches, including the one considered in this work (Zermas et al., 2017), use regression techniques to quickly perform plane fitting based on the raw point cloud data. Fast plane fitting techniques based on robust regression are popular in a wide variety of applications, and hence many off-the-shelf solutions exist that are low complexity and highly accurate.

2.2.2 Point cloud object clustering

A clustering method needs to divide an unorganized point cloud model into smaller parts so that the overall processing time is significantly reduced. A simple data clustering approach in a Euclidean sense can be implemented by making use of a 3D subdivision of the space using fixed-width boxes, or more generally, an octree data structure. This representation is very fast to build and is useful for situations where either a volumetric representation of the occupied space is needed, or the data in each resultant 3D box (or octree leaf) can be approximated with a different structure. We can make use of nearest neighbors and implement a clustering technique that is essentially like a flood fill algorithm. The Point Cloud Library (PCL) provides a fast and effective implementation of Euclidean clustering for point cloud data.

The other main approaches for point cloud clustering include DBSCAN (Ester et al., 1996) and Mean-Shift (Comaniciu and Meer, 2002), which we did not find to be necessary for our application as Euclidean clustering already provided reasonable performance with low complexity.

2.2.3 Cluster classification

Several pattern recognition methods exist to classify pedestrian clusters. Here we briefly review the two most promising choices for our application: the support vector machine (SVM) (Cortes and Vapnik, 1995) and deep neural networks (DNNs).

SVM requires us to find a hyperplane that best divides a feature plane. For example, in the two-class case shown in Figure 2-1, the task is to find a hyperplane to best divide the blue and green dots. In this figure, the red hyperplane is the best one that achieves maximum margin between the hyperplane and the support vectors (parallel lines upon which blue and green dots fall on that are closest to the red hyperplane). The SVM method requires hand-made features. SVMs have been used previously as highly performant classifiers for human detection in 3D LiDAR (Navarro-Serment et al., 2010, Kidono et al., 2011, Yan et al., 2017) and excel in the low data regime.

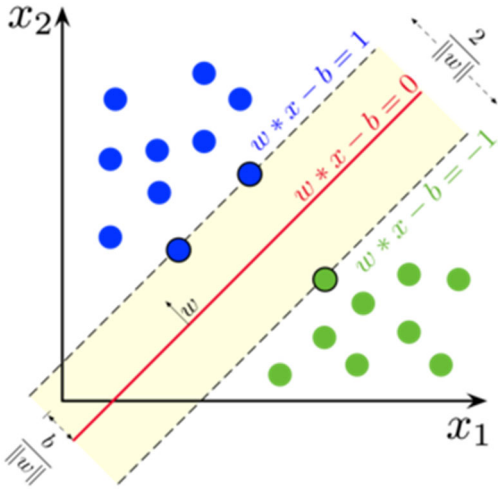


Figure 2-1: Basic binary SVM classifier.

Point clouds are an important type of geometric data structure. Due to its irregular format, most researchers transform such data to regular 3D voxel grids or collections of images. This, however, renders data unnecessarily voluminous and causes issues. In (Qi et al., 2017) and (Qi et al., 2018) they design a novel type of neural network that directly consumes point clouds, which adheres to the permutation invariance of points in the input. The network, named PointNet (Figure 2-2), provides a unified architecture for applications ranging from object classification, part segmentation, to scene semantic parsing. Empirically, it shows strong performance on par or even better than state of the art. To deal with an unordered input set, the key to this approach is the use of a single symmetric function, max pooling. Effectively, the network learns to select interesting or informative points of the point cloud and encode the reason for their selection. The final fully connected layers of the network aggregate these learnt optimal values into the global descriptor for the entire shape as mentioned above (shape classification) or are used to predict per point labels (shape segmentation).

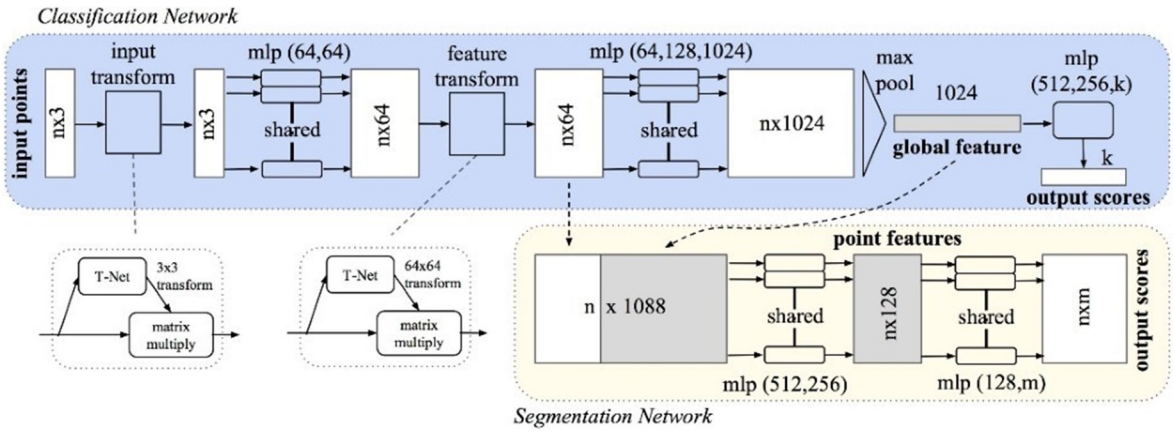


Figure 2-2: PointNet architecture.

The main drawback of this approach is that the number of parameters in the model is very high (in the millions), and therefore the training dataset must be very large (tens of thousands or hundreds of thousands of examples). Due to the scarcity of labeled data for the considered problem, we chose to use the SVM classifier.

2.3 Data collection

This section describes the steps taken to collect the LiDAR dataset of annotated pedestrians at an urban traffic intersection, as well as the details of our classification method. Then, we discuss how the detections from the LiDAR sensor can be provided to a multimodal sensor framework that also leverages video and radar data streams. An intelligent traffic intersection control system uses the multimodal sensor fusion outputs to compute optimal signal control and vehicle trajectories simultaneously.

2.3.1 LiDAR data collection

We recorded two sequences at the intersection at Gale Lemerand Dr. and Stadium Rd. in Gainesville, FL using the Ouster OS-1 16 beam LiDAR (Figure 2-3).

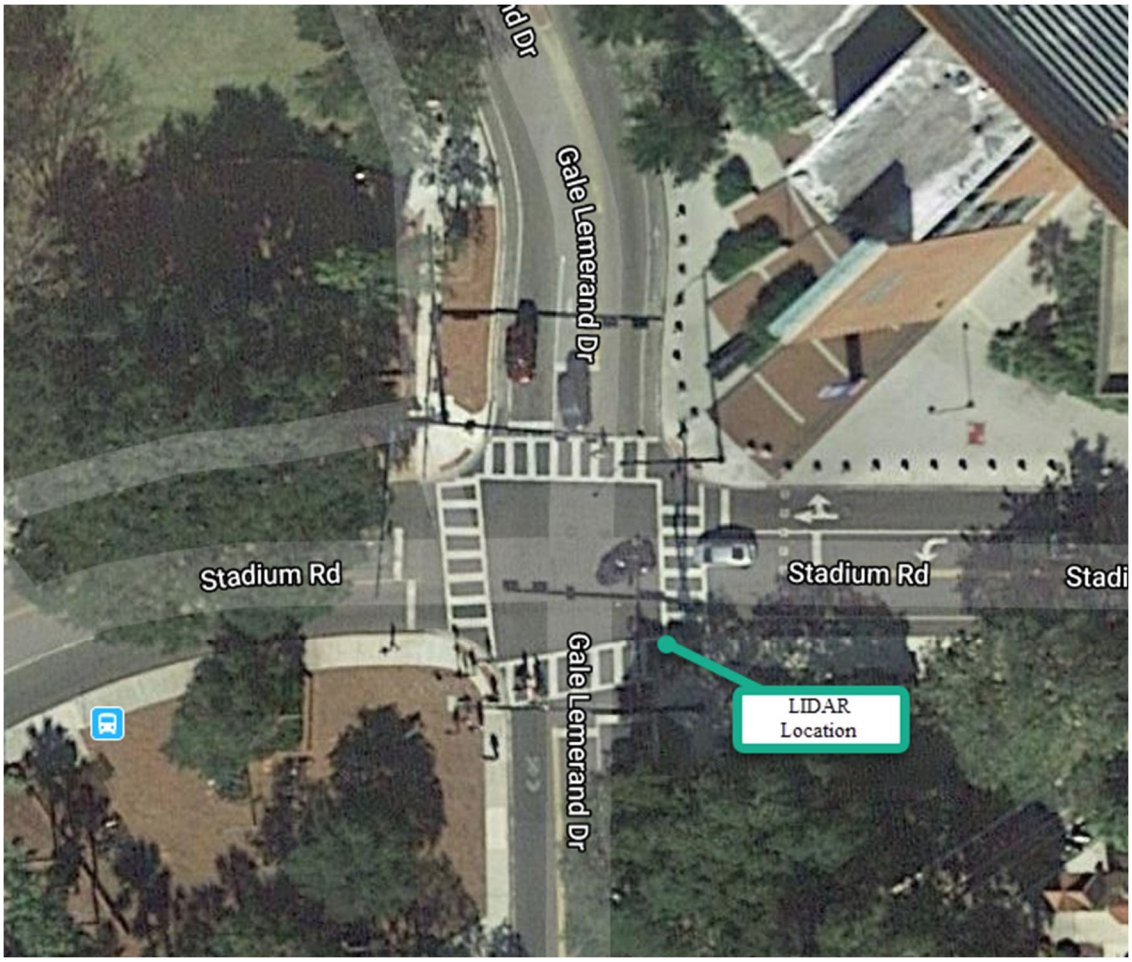


Figure 2-3: LiDAR setup.

The sensor was mounted on a tripod at 2 m from the ground and connected to a rig running on an Intel NUC 6. The first sequence is 4 minutes long and has 2,303 frames. The second recorded sequence is 20 mins with 11,228 frames. It is a busy intersection with cars, trucks, motorcycles, bicycles, and pedestrians present. Also, at times there were a few big groups of pedestrians crossing. There are also stationary objects including walls and trees. The OS-1 LIDAR created two files for each sequence of raw point clouds data: the first is 3.7 Gb, and the second 18.2 Gb. Figure 2-4 provides an example visualization of a point cloud.

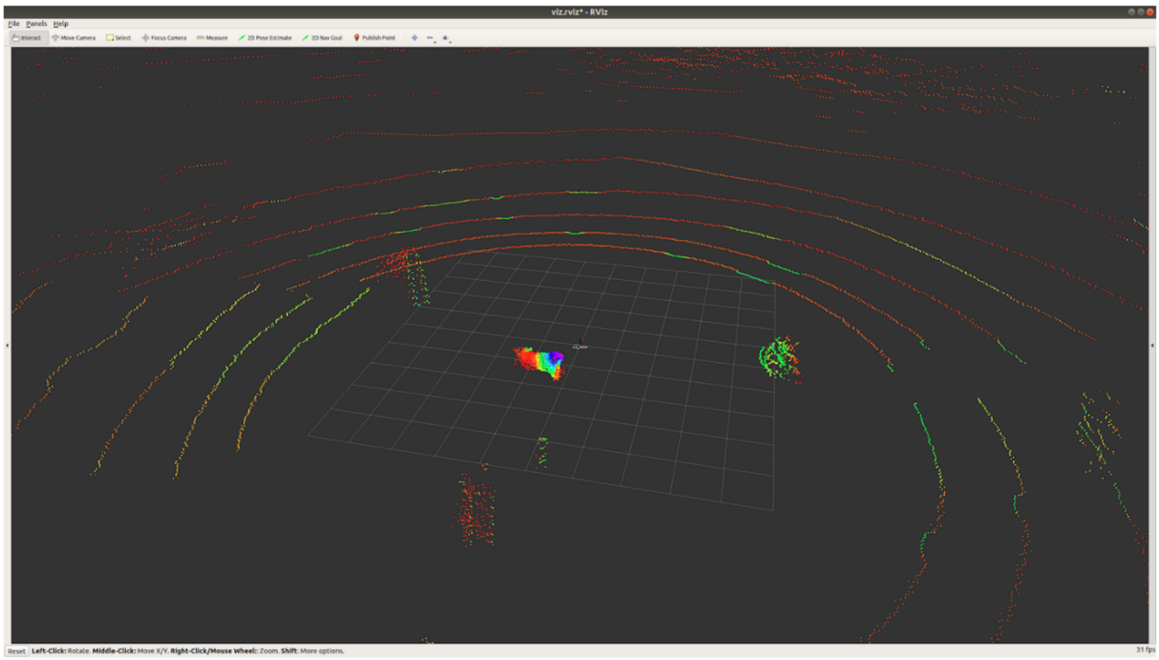


Figure 2-4: Raw point cloud visualization.

2.3.2 Pedestrian detection dataset

For both recorded sequences we extracted each frame and stored them as Point Cloud Data (PCD) files. Each PCD file contains two sections:

- A human-readable header that defines the number, size, dimensionality, and data type of the point cloud.
- A data section which can be in ASCII or a binary, non-human-readable format.

We load the PCD files in a Cloud Annotation Tool (Yan et al., 2017). We had to carefully go through each frame in order to label pedestrians (positive data) as well as non-pedestrians (negative data). LIDAR pedestrian point clouds look very different depending on how far a person is from the detector. Figure 2-5 shows an example of a human (1.68 m high) detected by OS-1 16-beam LIDAR at different distances.

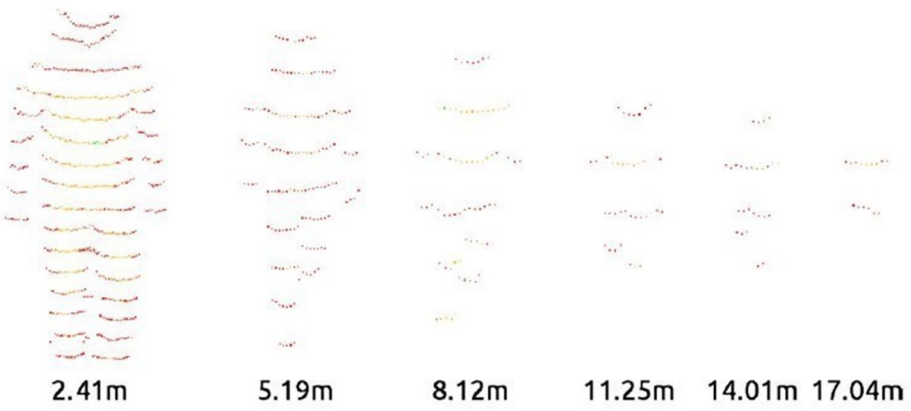


Figure 2-5: Examples of a human detected at different distances (Yan et al., 2017)

In total we have 3,691 labels, 1,811 positives (Pedestrians) and 1,880 negatives (Non-Pedestrians). Each label refers to either a pedestrian or an object that is not a pedestrian in a single frame. One frame may have more than one label since there could be multiple pedestrians visible to the LiDAR, and there may be multiple vehicles or other non-pedestrian objects visible as well. Hence, out of all considered frames, we identified 3,691 objects (total sum of pedestrians and non-pedestrians) to use for training the SVM. To split the dataset into a training set and test, we set aside 70% of the positive and negative labels for training (Train = 2,471) and the remaining 30% for testing (Test = 1,218).

2.4 Methods

The analysis of the data consists of the following three steps:

- Ground segmentation
- Non-ground clustering
- Clustering classification

For **Ground segmentation**, we followed (Zermas et al., 2017). Generally, a single plane model is insufficient for the representation of the real ground surface as the ground points do not form a perfect plane and the LiDAR measurements introduce significant noise for long distance measurements. (Zermas et al., 2017) introduced a novel method that out-performed RANSAC on this ground plane segmentation, called Lowest Point Representative (LPR). Initially we need point extraction to make it converge faster than RANSAC. Then we find points with low height and use them to estimate an initial ground plane. After that we find the distance between each point and its orthogonal projection on the candidate plane. Then we must compare the distance with a user-defined threshold. If smaller, we treat it as ground. We refine the estimation of a new plane and the process repeats for N iterations.

For **Non-ground clustering**, we adopted steps from the PCL software. Assume we have a point cloud with a table and objects on top of it. We want to find and segment the individual object point clusters lying on the plane. If we use a Kd-tree structure for finding the nearest neighbors, the algorithmic steps for that would be as follows: (a) create a Kd-tree representation for the input point cloud dataset; (b) set up an empty list of clusters and a queue of the points that need to be checked; (c) for every point, calculate its nearest neighbor; (d) terminate the algorithm when all points have been processed and are now part of the list of point clusters.

In our case, this method is applicable but must be optimized. Due to the sparse feature of the LiDAR point cloud, directly using the PCL Euclidian cluster extraction would yield poor performance because the distance of reflections between two neighbor beams is positively correlated with the distance from the objects to the sensor. The sparse density challenge could be easily solved by mapping all the points into an x-y plane, taking advantage of the fact that objects at the traffic intersection barely interface each other on the z direction (vertical to the ground). As the constraint of points' z-distances are weakened, the algorithm could cluster points in x-y 2D world, and then return the indices of points of different clusters to the original 3D data.

Finally, for **Clustering classification**, we used a binary SVM classifier. The SVM technique requires us to find a hyperplane that best divides feature planes. The SVM method also requires hand-made features. From Yan et al. (2017), six features with a total of 61 dimensions are extracted from the clusters for human classification, as shown in Table 2-1. The set of feature values of each sample Cj forms a vector fj = (f1, . . ., f6). Features from f1 to f4 were introduced by Navarro-Serment et al. (2010), while features f5 and f6 were proposed by Kidono et al. (2011). A binary classifier is trained for pedestrian detection (i.e. pedestrian or non-pedestrian), based on the above features, using LIBSVM, the standard software library for the support vector machine (Chang and Lin, 2011).

Table 2-1: Features for human classification.

Features	Description	Dimension
f1	Number of points included in the cluster	1
f2	Minimum cluster distance from the sensor	1
f3	3D covariance matrix of the cluster	6
f4	Normalized moment of the inertia tensor	6
f5	Slice feature for the cluster (10 slices, x-y occupancy grid information)	20
f6	Reflection intensity's distribution (25 bins of normalized 1D histogram, mean and standard deviation)	27

Figure 2-6 provides the visual results for parameter tuning while searching for the best C (penalty parameter) and gamma (how quickly dissipation of class boundary occurs).

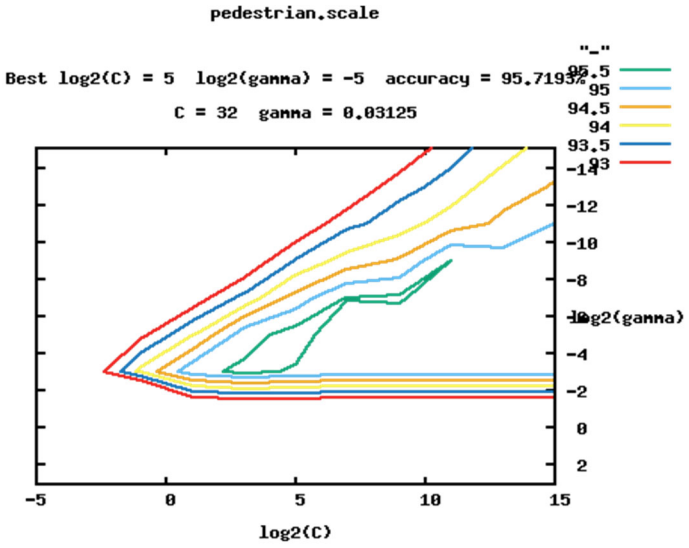


Figure 2-6: LibSVM scaling.

Table 2-2 provides the details for our SVM model.

Table 2-2: SVM pedestrian model.

Parameter	Value/Type
svm_type	c_svc
kernel_type	rbf
gamma	0.03125
nr_class	2
total_sv	752
rho	1.99823
label	1, -1
nr_sv	408, 344

The parameters listed in Table 2-2 are parameters that help describe the details of the support vector machine binary classifier we trained on our dataset.

- svm_type: The flavor of SVM we use. We use a support vector machine classifier (c_svc).
- kernel_type: The family of functions we employ from which we hope to find a good fit for our training data. We use the radial basis function kernel (rbf).

- gamma: This is a tuning parameter for the radial basis function kernel used by the SVM. The value of 0.03125 has no other meaning except that at this value, the SVM implementation with rbf kernel achieved best cross-validation performance.
- nr_class: How many classes there are in the dataset. We have two. Pedestrian, and non-pedestrian.
- total_sv: The total number of support vectors after training the SVM. The value (752) can be interpreted as the number of data points that are crucial to the SVM's decision-making process when trying to classify a novel data point. Using 752 out of 1219 data points as support vectors is fine. Effectively, ~62% of the training data points are being used as support vectors.
- rho: This is a "bias" term in the computation of the predicted class for a data point. It is not related to test performance (e.g., correlation). The value of 1.99823 is what the SVM estimated as the best bias value to achieve the best cross-validation accuracy.
- label: The labels in the training data.
- nr_sv: The number of support vectors per class.

2.4.1 LiDAR integration into multimodal sensor fusion system

This subsection describes the output of the LiDAR detection system and how it is post-processed for use within our intelligent intersection control system.

The LiDAR pedestrian detection system identifies candidate clusters of 3D points and assigns them a binary label corresponding to whether it is a pedestrian or not. Based on this information, for each identified pedestrian, we can output the cluster centroids as a list of 3D points. This can be efficiently sent in real-time to the computing node running the centralized multi-sensor fusion framework. Based on known geometry about the intersection, the list of pedestrians is filtered by distance to the intersection center and organized by proximity to one of the four intersection corners. After this post-processing step, we have an approximate estimate for the pedestrian counts at each of the four corners of the intersection, which we can use as a proxy for the number of pedestrians waiting to cross.

Our multimodal sensor system allows for integration with diverse sensor types by providing a basic API for users to easily add new sensors and send data from the sensor to the main processing unit. For the LiDAR, we implement a driver node in Python that forwards the outputs of the detection over a network socket. Then, the fusion system parses the detection results and can either attempt to combine the list of pedestrian cluster centroids with its own set of detections from video and radar sensors using simple data association and fusion via linear assignment, or it can directly produce the relevant pedestrian counts at each intersection corner and send this information to the optimization algorithm.

2.5 Results and analysis

In this section, we provide a quantitative and qualitative analysis of the performance of our trained pedestrian detector. First, we introduce the relevant metrics for measuring the accuracy of the detector on our labeled dataset. Then, we detail the numerical results. Finally, we discuss the strengths and weaknesses of our method.

2.5.1 Evaluation metrics

Since this problem is treated as binary classification with an SVM, the classifier predicts binary labels for each provided cluster. By comparing each predicted label with the corresponding ground truth label, we can evaluate the ability of the classifier to successfully identify pedestrians. Mainly, we consider **precision**, or positive predictive value, which is defined based on the relationship between true positives (TP) and false positives (FP) using the following expression: (TP / TP + FP). The precision parameter identifies approximately the ratio of correctly detected pedestrians to the total number of objects perceived by the detector to be a pedestrian. A precision close to 1 signifies that few objects (e.g., cars, trucks, signposts) were confused for a pedestrian. The **recall** measures the fraction of pedestrians correctly identified out of all pedestrians in the dataset. Therefore, it answers the question "how many do we miss?" A recall close to 1 signifies that we do not miss any pedestrians. Finally, the **F1-score** is the harmonic mean of precision and recall and provides a holistic performance measure. An F1-score of 1 would be an ideal classifier.

2.5.2 Key results

Table 2-3 provides the results of the trained SVM on the test data.

Table 2-3: SVM test results.

	Precision	**Recall**	**F1-score**	**Support (N)**
Pedestrian	0.95	0.97	0.96	636
Non-Pedestrian	0.96	0.94	0.95	583
accuracy			**0.95**	1219
macro avg	0.95	0.95	0.95	1219
weighted avg	0.95	0.95	0.95	1219

Figure 2-7 through Figure 2-9 depict point cloud data correctly classified as pedestrians. As shown, the middle sections of some pedestrians have missing points, due to incorrect distance settings during the ground and plain removal stages.

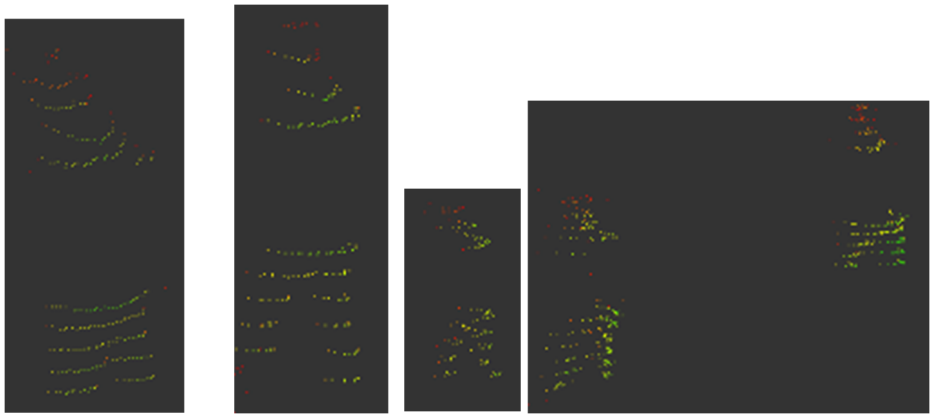


Figure 2-7: Correctly classified pedestrians.

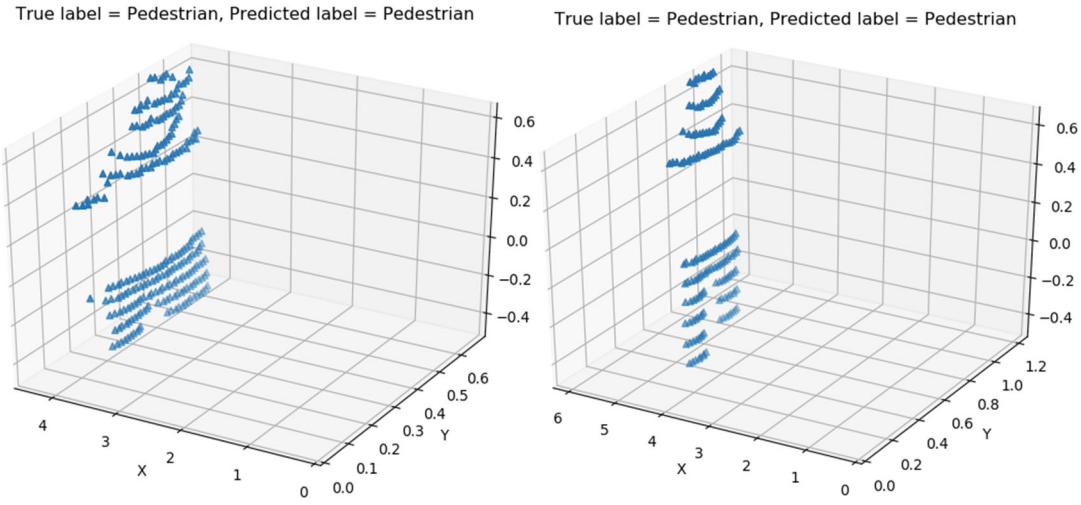


Figure 2-8: Correctly classified pedestrians.

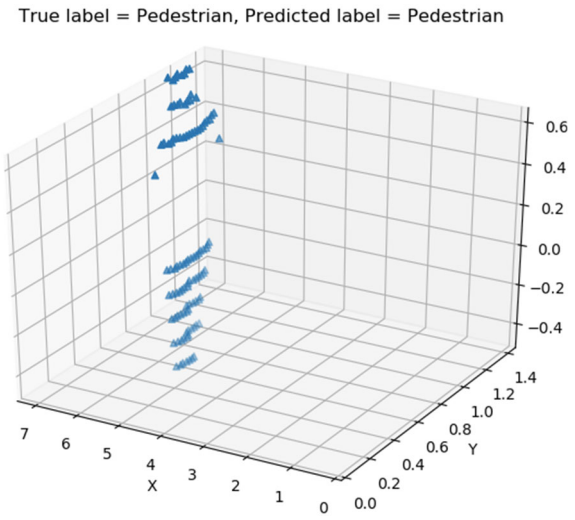


Figure 2-9: Correctly classified pedestrians.

Figure 2-10 shows point cloud data *incorrectly* classified as a pedestrian (false positives). On the left, the object is a pole. The object on the right appears to be an artifact (e.g., from a tree or sign).

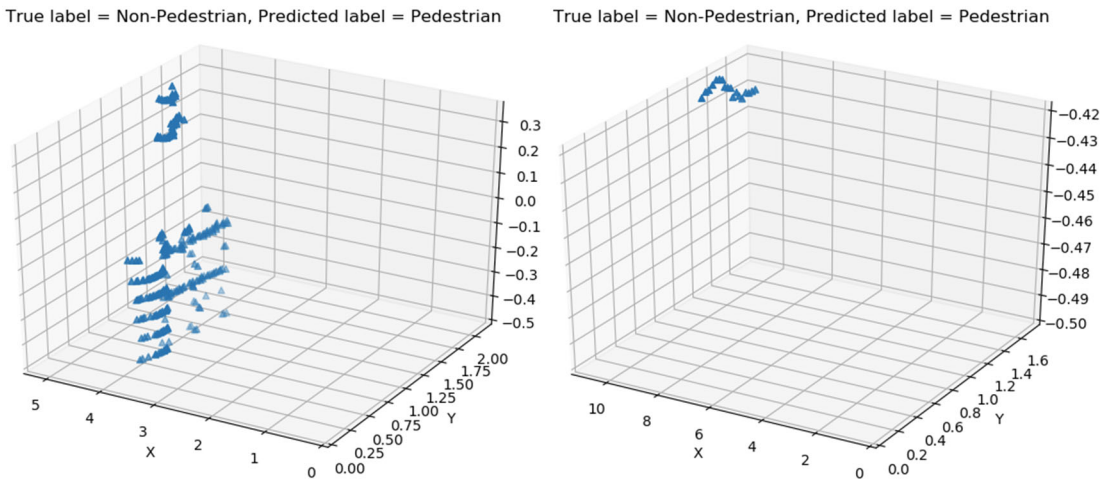


Figure 2-10: Examples of false positive classifications.

Figure 2-11 shows an example point cloud snippet that is *incorrectly* classified as a non-pedestrian. The classifier confused it with another object such as a pole, tree, or signpost.

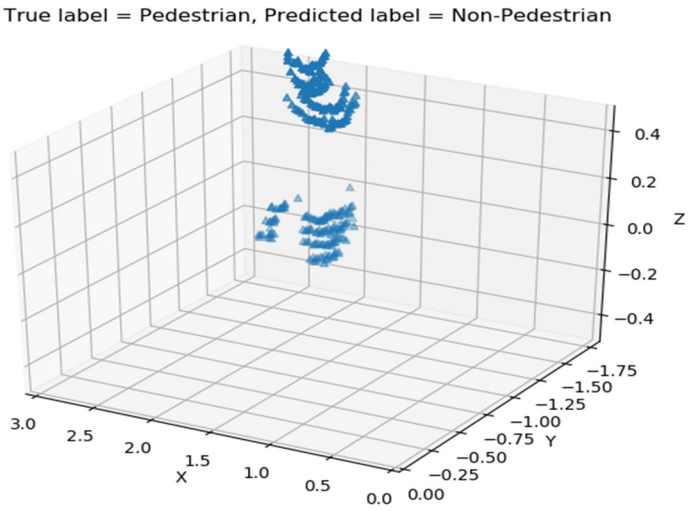


Figure 2-11: Example of a false negative classification.

Figure 2-12 illustrates the pedestrian identification with the addition of marker labels into an example point cloud.

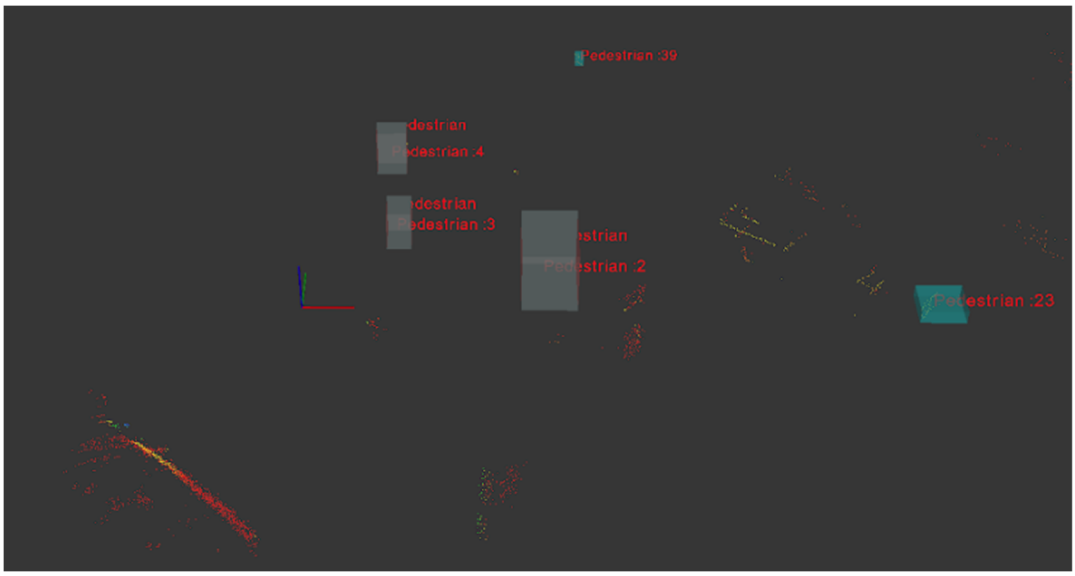


Figure 2-12: Prediction on processed point cloud.

From the SVM results on our dataset, we achieve test set performance of a 95% F1-score, which is highly accurate. However, if the height of the sensor changes, the 3D covariance matrix of the cluster (f3) and the slice features (f5) will also change. So far, our data have been obtained from a LiDAR mounted on a tripod, but ultimately, we hope to mount it in a more permanent way. Every time the height of the sensor changes, we will need to re-train the SVM. This can be accomplished through an 'online learning' algorithm (Yan et al., 2017) and by removing the dependence on features that are specific to a particular installation.

2.5.3 Discussion

Our current results are promising in that we can achieve a 95% F1-score on our held-out test set of examples. By addressing the issue of dropped points around the torsos of many of the pedestrians, we believe we can further improve the reliability of our detector. Since the main goal of this part of the project is to develop a LiDAR-based pedestrian detection system to provide inputs to a traffic signal optimizer, an accuracy of 95-97% is reasonable. The occasional false detection or missed detection will not have an immediate impact on the safety of vulnerable road users. Rather, the time allocated for crossing pedestrians may be slightly too much or too little.

2.6 Conclusions

This part of the research developed a LiDAR-based pedestrian detection system that is integrated into a multi-sensor framework in service of an optimization algorithm for intelligent traffic intersection control. We demonstrated the ability of the detector to achieve high reliability on pedestrian data collected "in-the-wild". Our detector achieves a test set F1-score of 95%, which is highly accurate.

The main obstacles to widespread deployment of this type of system are related to hardware concerns. LiDAR is more difficult to install and maintain at traffic intersections than video and radar, as they cannot be so easily mounted on mast arms. LiDARs have more limited range, and so must be within 20-30 m of the objects of interest. As LiDAR gets smaller and cheaper, however, they will become an increasingly viable option for improving sensor capabilities at the edge due to the precision and range of coverage of point cloud data.

3 ENHANCED DATA-SHARING BETWEEN AUTONOMOUS VEHICLE AND INFRASTRUCTURE

3.1 Overview

In the original version of RIO, a CAV 'checked in' with the IC and provided its position, speed, and intent. This information was used by the IC to optimize the intersection signal control to maximize intersection capacity and minimize travel times. The IC sent the CAV a trajectory for its approach to and passage through the intersection. It also sent the traffic signal controller optimized signal timing settings. The IC consists of two main algorithms, Sensible and RIO. It assembles information from several sensors and fuses it to provide vehicle arrival information to the optimization algorithm. The Real-time Intersection Optimizer (RIO) is an algorithm developed by UFTI under the same NSF funded project, which optimizes Signal Phase and Timing (SigPT) and vehicle trajectories.

The IC operates on a Linux computer equipped with a roadside unit (RSU) DSRC radio to communicate with autonomous and connected vehicles. The IC is also connected to an Econolite signal controller (it can also be connected to other NEMA TS2 controllers). This setup allows the SPaT optimized by RIO to be deployed in real-time at a signalized intersection.

The objective of this task is to investigate how the safety and efficiency of the CAV can be enhanced by exchanging additional information that can benefit both the vehicle and the intersection operations by expanding their field of detections. To achieve this, two additional classes of information are communicated between the CAV and the IC. The first class of information that is sent from the IC to the CAV is the static data that represents the major landmarks associated with the intersection. These data include, for example, the location and the size of poles and signs as well as the type and the location of traffic lights. Lane marking information are also provided. This is a static data that is assumed to not change. Therefore, there is no real time computational burden on the IC.

The second class of information that is sent between the IC and the CAV is dynamic data. This is a list of identified moving, or potentially moving, objects in the vicinity of the intersection or the vehicle. Objects are classified as vehicles, bicycles/scooters, or pedestrians. Location and current velocities are communicated between the IC and the CAV. The motion profiles of any other CAVs (or other sensed connected vehicles), as calculated by the IC, are also sent to the CAV.

The primary contributions of this research component are (1) the development of the necessary message sets that are needed to extend the current Dedicated Short-Range Communications (DSRC) standard data sets and (2) evaluation of the impact of this enhanced data exchange between the CAV and the IC as it pertains to traffic flow and the CAV operational safety.

Note that in the previous FDOT project (BDV31-977-45), two messages were produced to communicate two classes of information between the CAV and the IC. The first message was sent from the CAV to the IC. The vehicle would report its position and intent. The second message was sent from the IC to the CAV. This message would give the vehicle a commanded motion profile (list of position vs. time) as it approaches the intersection.

3.2 Development of new messages

3.2.1 Static message

At the start of the project, it was believed that the CAV's knowledge base would be improved if it knew the location and classification of static objects in the vicinity of the intersection. The plan was for this data to be obtained by the IC and communicated to the CAV. A change, however, was made to the CAV where it determines its pose by comparing current LiDAR data to pre-mapped LiDAR data of the intersection. By already having the pre-mapped data, the CAV has knowledge of the static objects, and it is no longer necessary to communicate this information from the IC to the CAV. For this reason, communication of the static information from the IC to the CAV was redundant and not performed here.

3.2.2 Dynamic message

The structure of the dynamic message has been designed to be source-independent, allowing it to originate from either the CAV or the IC without need for source-specific modifications to the structure. The message is composed of a header and an object description which is repeated for each object being tracked up until a system specified maximum batch size limit. For this application, the batch size was limited to five objects per message. In the case that the number of objects being tracked exceeds the batch size limit, the total set of tracked objects is divided into subsets. Each subset describes a number of objects up to the batch size limit, and a message is generated and sent for each subset. The batch size limit is imposed to avoid problems of scale that occur in large intersections or those that have high pedestrian traffic.

The header describes the time of message generation, the estimated latency in object detection, the location of the source's origin in latitude and longitude, the total number of dynamic objects being tracked by the source, and the number of objects reported in the message. The structure of the message header is shown in Table 3-1.

Table 3-1: Object tracking message header.

Information Type	Dedicated Memory
Message ID	1 Byte
Message Number	2 Bytes
Source ID	4 Bytes
Current Hour	1 Byte
Current Minute	1 Byte
Current Second	2 bytes
Uncertainty in Time	2 Bytes
Source Latitude	4 Bytes
Source Longitude	4 Bytes
Total # of Objects	2 Bytes
Batch Size	1 Byte
Total Size	24 Bytes

The time of message generation is reported in Coordinated Universal Time (UTC) and is taken from the source's GPS. The current hour is reported as a value from 0 to 23 where 0 is designated as midnight. The values of 24 to 30 are reserved. The value 31 is sent when the current hour is unavailable. The current minute is reported as a value from 0 to 59. The values 60 to 62 are reserved. The value 63 is sent when the current minute is unavailable. The current second and the uncertainty in time is reported in milliseconds. The current second is reported in the range of 0 to 60,999 milliseconds. The values between 61,000 and 65,535 are reserved. The value 65,536 is sent when the current second is unavailable.

The location of the source's origin is reported in latitude and longitude. It is used to establish an easting-northing coordinate system in which the source will report location of tracked dynamic objects.

The latitude of the origin is reported as a value between [-9E8:9E8] resulting in a resolution of $1/10^{th}$ of a micro degree. The value 900,000,001 is sent when the latitude of the source centroid is unavailable.

The longitude of the origin is reported as a value between [-18E8:18E8] resulting in a resolution of $1/10^{th}$ of a micro degree. The value 1,800,000,001 is sent when the longitude of the source is unavailable.

The total number of tracked objects is reported using two bytes in order to accommodate large intersections with high foot traffic.

The number of objects being tracked is represented with a single byte. The number of objects in a batch is limited by a user-designated maximum.

The object description provides the object type, position, orientation, bounding box, velocity, and uncertainty in the position of an object detected by the message source. This structure is repeated within the message up until the number of objects included reaches the maximum batch size. The structure of the dynamic object's representation is shown in Table 3-2.

Table 3-2: Object representation structure.

Information Type	Dedicated Memory
Object ID	4 Bytes
Object Class	1 Byte
Object Detection Hour	1 Bytes
Object Detection Minute	1 Bytes
Object Detection Second	2 Bytes
Delta X	2 Bytes
Delta Y	2 Bytes
Orientation	2 Bytes
Length	2 Bytes
Width	2 Bytes
X Velocity	2 Bytes
Y Velocity	2 Bytes
Angular Velocity	2 Bytes
Uncertainty in X	2 Bytes
Uncertainty in Y	2 Bytes
Total Size per Object	29 Bytes

The time of object detection is reported in the same fashion as the time of message generation. The object representation includes a classification for each object (vehicle, bicycle/scooter, pedestrian.) When an object has left the detection range of the message source, the object classification may be set to a "No Longer Tracking" case. The enumeration of dynamic object classification is shown in Table 3-3.

Table 3-3: Object label enumeration.

Dynamic Object	Class Value
No Longer Tracking	0
Passenger Vehicle	1
Motorcycle/Scooter	2
Pedestrian	3
Tractor Trailer	4
Unidentified	255

The centroid of each object is represented as a distance in X and Y measured from the origin specified in the message header. These distances are reported in units of centimeters.

The orientation of the object is reported in the range of -3,600 to 3599 degrees measured from the X-axis. The resolution of this measurement is 0.05 degrees.

The length and width of the object is reported in the range of 0 to 2,000 and is measured in centimeters. This results in a maximum vehicle dimension of 20 meters.

The velocity of the object is reported as the X and Y components of velocity. These values are reported using two signed bytes and is measured in units of centimeters per second. The range of expected values is -2,000 to 1,999 centimeters per second, corresponding to a maximum velocity of approximately forty-five miles per hour.

The angular velocity about the Z axis of each object is reported using two signed bytes and is measured in radians per second.

The uncertainty in the components of the centroid position is reported using two unsigned bytes and is measured in centimeters.

Figure 3-1 shows the flow of data from detection to reception. As shown, information flows both from the intersection to the vehicle, and from the vehicle to the intersection. Box 1 consists of the video camera which is aimed at one of the intersection approaches and acts as the IC's (Box 2) primary sensor for the approach. The IC then forwards the position, orientation, object class, and detection time of any dynamic objects detected in the camera image to the COHDA road-side unit (RSU) (Box 3). The RSU then parses object details into the OTM format, given a transmission timestamp, and transmits to the CAV's on-beard unit (Box 4). The received OTM is given a reception timestamp and forwarded to the CAV's Computer (Box 5). The CAV may then use this information to expand its field of detection.

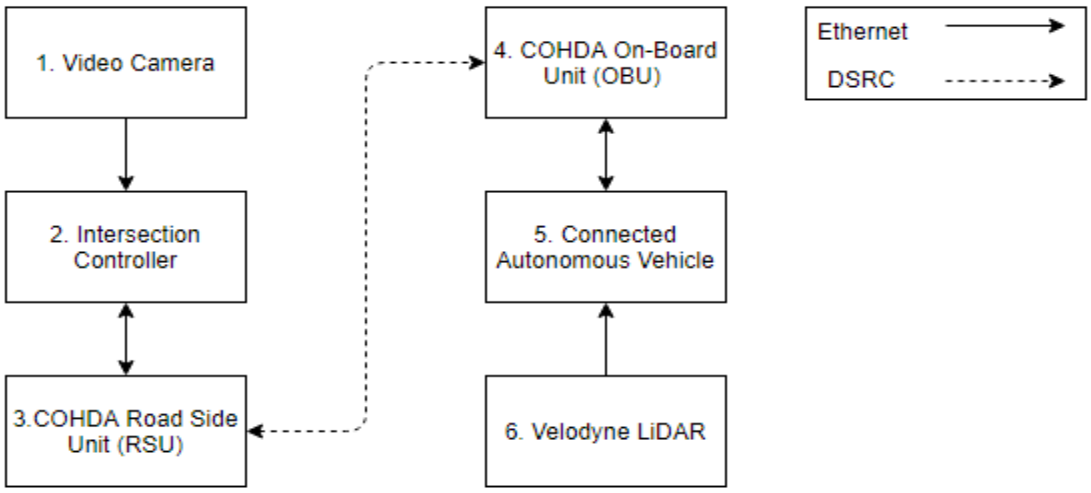


Figure 3-1: Data flow schematic.

Box 6 consists of the Velodyne LiDAR which is mounted to the top of the CAV and forwards point cloud detections to the CAV (Box 5). The CAV then forwards the position, orientation, object class and detection time of any dynamic objects detected in the point cloud to the COHDA on-board unit (OBU) (Box 4). The OBU then parses object details into the OTM format, given a transmission timestamp, and transmits to the IC's RSU (Box 3). The IC may then use this information to expand its field of detection.

3.3 Testing

Initial testing of the object tracking message was preformed using a simulation of the autoware.ai autonomous vehicle software to generate dynamic object messages in a repeatable environment. The simulation applies Euclidian cluster detection to stored LiDAR and GPS data to estimate the position of objects relative to the vehicle.

Figure 3-2 shows the top-down view of the relevant section of the prerecorded LiDAR point cloud map of the TERL facility.

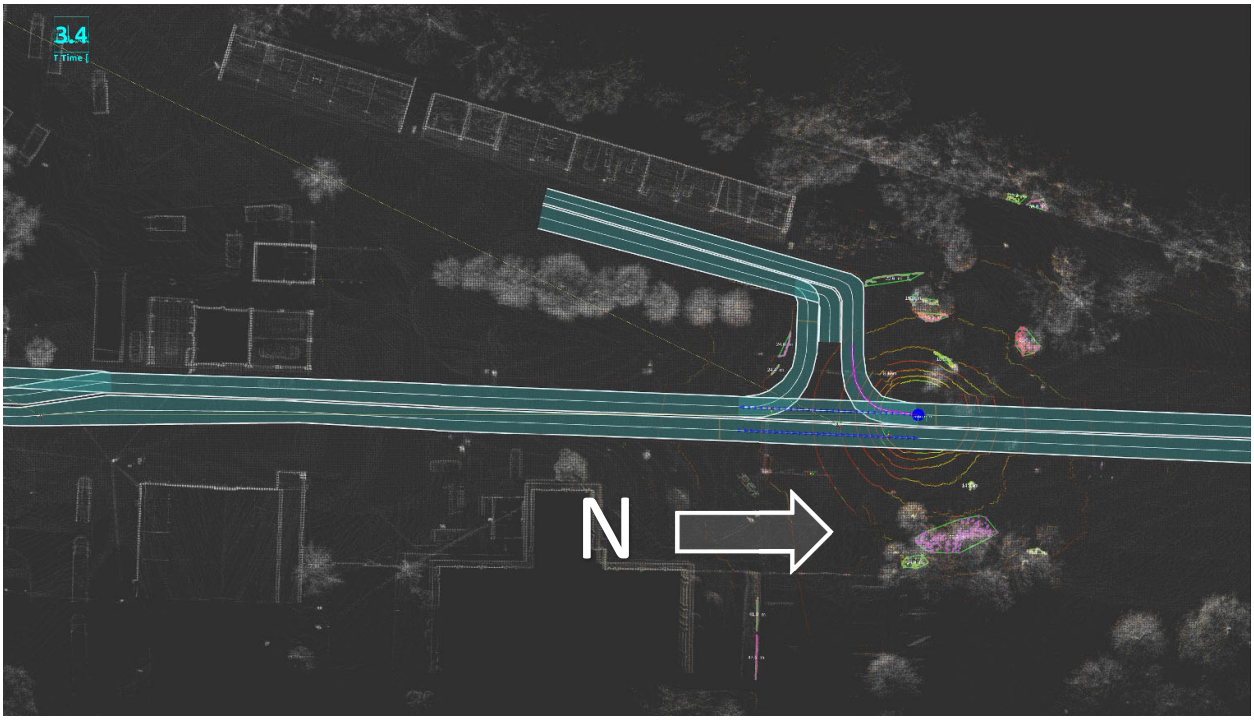


Figure 3-2: TERL test intersection and its point cloud map.

Field testing of the dynamic message was conducted at the Traffic Engineering and Research Laboratory (TERL) in Tallahassee, Florida. The CAV was parked at the stop bar in the intersection's southbound lane (Figure 3-2). A conventional vehicle approached the intersection in the northbound lane. As the conventional vehicle passed through the field of detection of the intersection, object-tracking messages were populated and transmitted to the CAV from the IC. Evaluation of the viability of the object tracking message (OTM) consisted of examining latency of the message transmission and reception and the accuracy of the intersection's reported object poses relative to the CAV's LiDAR-based detections.

Figure 3-3 shows the top-down view of the relevant section of the prerecorded LiDAR point cloud map of the TERL facility. The green polygon on the map surrounds clusters of active LiDAR detections that the CAV has classified as obstruction in real time. The CAV is capable of

consistently identifying cluster obstructions within 25 meters of the Velodyne LiDAR [16 beams] and sporadically identifying cluster obstructions within 50 meters. This ability to identify clusters beyond 25 meters is dependent on the size and profile of the obstruction. In subsequent figures the blue marker represents a two-meter radius around the Velodyne sensor, and the red markers show the zone where detected objects may be located. This zone represents a two-meter radius of uncertainty in the Sensible's estimation of the position of the detected object. It assembles information from several sensors and fuses it in order to provide vehicle arrival information to the IC.

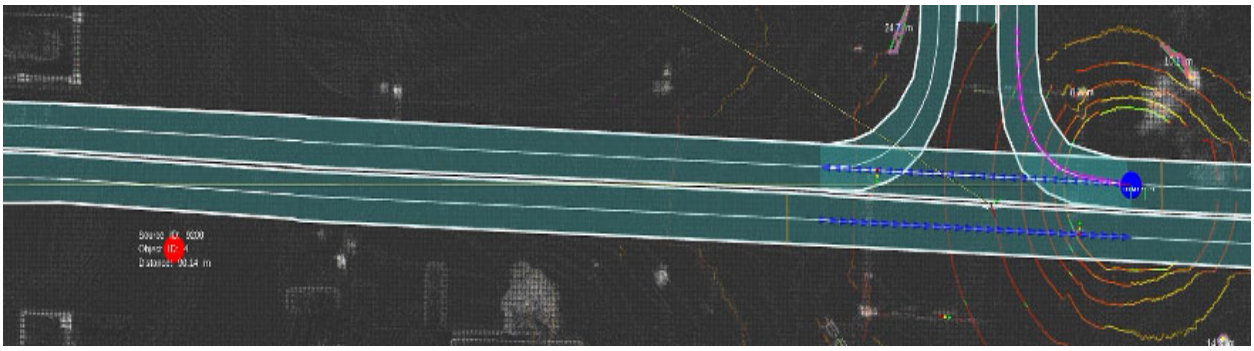


Figure 3-3: First detection of the conventional vehicle by the IC.

The CAV assigns OTM - based markers a one-second lifetime and will remove the object marker unless it is updated within that lifetime. The first received message place the oncoming vehicle approximately ninety meters south of the stationary CAV. In Figure 3-3, the marker representing a Sensible-detected object falls outside of the roadway. This object remains outside of the roadway for the entirety of its detection.

Figure 3-4 and Figure 3-5 show the movement of the conventional vehicle as detected by the CAV's LiDAR over the course of the last received OTM's one second lifetime. There is an error in total position of approximately 20 m between the intersection and the LiDAR- based detections. This error may be due to inaccuracies in the point cloud's latitude and longitude alignment and/or due to latency in Sensible's detection of the object. The research team is conducting additional tests to determine the reason for this inconsistency and attempt to correct it.

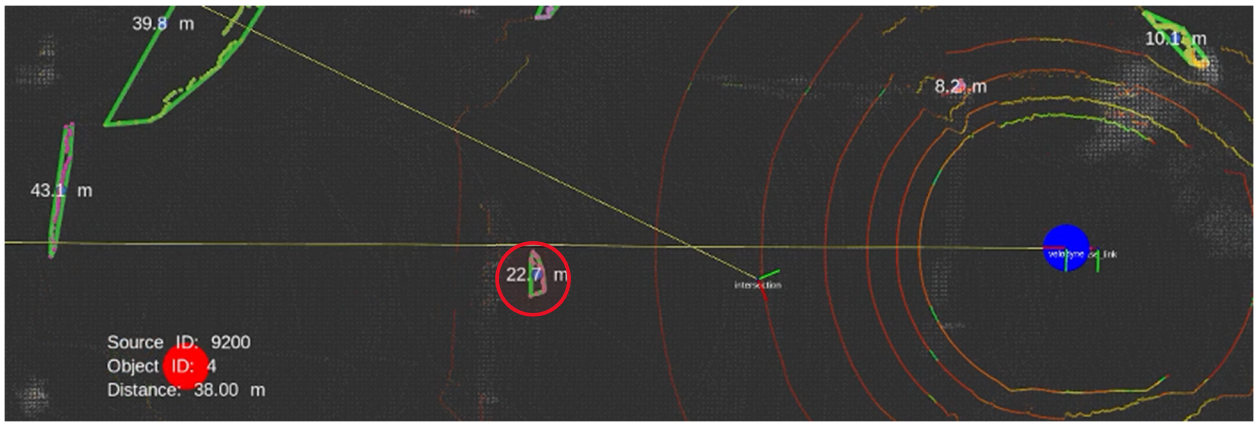


Figure 3-4: First detection of the conventional vehicle by the CAV LiDAR.

Figure 3-5: Last detection of the conventional vehicle by the IC.

Examination of Sensible's object detection time stamps and the CAV's reception time stamps indicate an average lapse of 0.13 seconds between object detection and reception of the OTM. However, an error in the COHDA roadside unit (RSU) resulted in sporadic and potentially unreliable transmission timestamps. For example, the radio-assigned time stamps were shifted by several hours relative to the time of day and to each other. This shift in time stamps was caused by a momentary loss of GPS signal. Thus, this latency may be subject to errors associated with desync between the clocks of RSU and the CAV's COHDA on board unit (OBU).

3.4 Potential impact of the new messages

The hypothesis is that the autonomous vehicle will be able to make better decisions if it has more information about the position and velocity of other vehicles at the intersection. Two examples of actual events that the UF researchers experienced at the DARPA Urban Challenge event are presented here. In both cases, the vehicle made an inappropriate decision based on incomplete or

faulty data. Having the intersection communicate additional information would have resulted in the correct outcome.

The DARPA Urban Challenge was a competition where autonomous vehicles were to navigate on streets while obeying standard traffic laws. The final event was held on 3 Nov 2007 in Victorville, California. The autonomous vehicles were given a road network database file and a mission file. The road network database defined how roadways were connected. The mission file contained reference points that the vehicle was to visit. Vehicles had to obey all traffic regulations while negotiating with other traffic and obstacles and merging into traffic. The vehicle had to make 'intelligent' decisions in real time based on the actions of other vehicles. What was new in this competition was that vehicles were moving through a more cluttered urban environment as compared to just performing highway driving where there is little interaction between the vehicles. Here, vehicles had to perform more complex tasks such as negotiating a four-way stop intersection and navigating in a parking space area.

In the first case, the autonomous vehicle approached a four-way-stop intersection. Two vehicles were at the intersection, one behind the other, before the autonomous vehicle arrived. Figure 3-6 shows the scene after the autonomous vehicle arrived.

Figure 3-6: Autonomous vehicle at four-way-stop intersection.

Since the other two vehicles had arrived at the intersection before the autonomous vehicle, the correct behavior would have been for the autonomous vehicle to wait until the white van proceeded through the intersection. The autonomous vehicle would go next. It wanted to make a right turn at the intersection. Finally, the pick-up truck would move up to the stop line and then proceed through the intersection.

Instead of this, the autonomous vehicle arrived at the intersection, briefly paused, and then turned right before the white van moved (see Figure 3-7). The reason for this was that based on LiDAR data, the van and truck were classified by the autonomous vehicle as one long vehicle. The center of mass of the 'long vehicle' was determined. Assuming an average vehicle length of 15 feet, the calculated front of the vehicle was not yet at the stop line. Since the 'long vehicle' was not yet at the stop line, the autonomous vehicle acted as if it was the first vehicle to arrive at its stop line and proceeded accordingly.

Figure 3-7: Autonomous vehicle performs incorrect behavior.

The fix that was implemented after this occurred was to use the LiDAR data to measure the length of the 'long vehicle' upon which it would have been determined that the 'long vehicle' was indeed at the stop line before the autonomous vehicle arrived. The autonomous vehicle would have waited until the 'long vehicle' moved through the intersection. Of course, it would have had to reevaluate the scenario when only half of the 'long vehicle'. i,e. the van proceeded through the intersection.

Today, autonomous vehicles are much more adept at interpreting a scene such as this correctly. Multi-beam LiDAR and vision can quite accurately identify the two separate vehicles. However, some interpretation errors may occur if parts of the 'long vehicle' are occluded. Having the intersection sensors correctly identify the two vehicles and then communicating this information to the autonomous vehicle would have resolved the problem.

The second scenario occurred at the DARPA Urban Challenge qualification round in Victorville, California. Figure 3-8 provides an overview of the scenario. Multiple human driven vehicles travelled in both lanes around the loop at a speed of 10 mph. The autonomous vehicle was to drive the half-loop indicated by the red and yellow arrows. At the intersection marked by the stop sign, the autonomous vehicle had to stop, but the human driven vehicles did not.

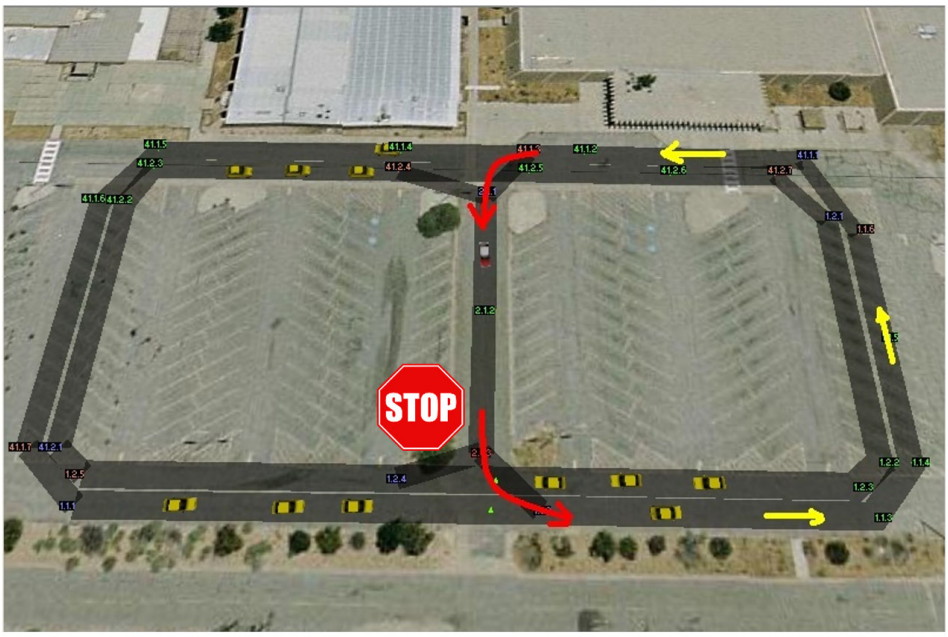


Figure 3-8: DARPA Urban Challenge scenario.

Figure 3-9 shows the autonomous vehicle at the stop line, waiting for the opportunity to turn left and merge onto the outer loop. The UF vehicle operated correctly until one particular scenario occurred.

Figure 3-9: Autonomous vehicle waiting at stop line.

Figure 3-10 shows the autonomous vehicle at the stop line and a human driven vehicle traveling in the outer loop. At the instant of the picture, the autonomous vehicle has detected this human driven vehicle and is tracking it accordingly.

Figure 3-10: Autonomous vehicle waiting at stop line.

Figure 3-11 shows the scene approximately ten seconds later. Two closely following vehicles in the inner loop have occluded the 'problem vehicle' so that the autonomous vehicle cannot sense it. The autonomous vehicle software was programmed to delete any vehicle that has not been sensed for three seconds. Thus, at the instant of Figure 3-10, the autonomous vehicle has removed from memory that the 'problem vehicle' ever existed and has determined that it is safe to proceed through the intersection. This was an incorrect behavior.

Figure 3-11: Two vehicles on inner loop occlude the 'problem vehicle' on out loop.

Figure 3-12 shows the autonomous vehicle pulling out in front of the human-driven vehicle. Once the autonomous vehicle determined it was safe to proceed (because the 'problem vehicle' was occluded for more than three seconds), it began to enter the intersection. Once the 'problem vehicle' was re-detected, it was too late.

Figure 3-12: Autonomous vehicle pulls out in front of problem vehicle.

This scenario would have been avoided had there been communication between the autonomous vehicle and the intersection controller which is sensing and reporting the location and velocities of vehicles and pedestrians near the intersection. This represents the true impact of this work. Increased information provided to the autonomous vehicle will result in much safer operation.

3.5 Conclusions

Initial testing of the object tracking message show viability due to its low latency. The knowledge base of the CAV was increased as other vehicles became known to the CAV at distances that far exceeded the CAV's onboard detection range. The testing did show the importance of accurately referencing the CAV's pre-mapped point cloud (which the CAV's position is determined from) and the coordinate reference of the IC system. An accuracy of less than 10 cm, which is obtainable, should be sufficient.

It is important that the location of vehicles and pedestrians be accurately determined and recorded by the IC. Similarly, the CAV must accurately determine its location and the location of other vehicles and pedestrians. It is possible to obtain RTK GPS accuracies on the order of 2 cm (using for example the FDOT FPRN system) which would adequately determine the IC reference point and the CAV position. The accuracy of determining the position of other vehicles and pedestrians is then primarily a function of the accuracy of the sensors used for detection. Typical LiDAR range accuracy is 1 to 5 cm with a range of 120 m. Radar resolution is on the order of 1 m with a range of 100m. Vision sensor range accuracy is a function of the camera resolution. From the point of view of the CAV, the accuracy of the additional knowledge gained by communicating data from the IC is adequate. Knowing that a vehicle or pedestrian is present is what is important, and accuracies to less than a few cm do not seem required. When

the CAV is in the near vicinity of these objects, it will use its onboard sensors to accurately map its relative position.

Future testing of the OTM will implement scenarios using increasing numbers of conventional vehicles to determine the impact on message latency. Additionally, improvements to the CAV's hardware, such as increasing the number of LiDAR beams from 16 to 32, will allow for an improved evaluation of pose accuracy.

4 INTERSECTION OPTIMIZER EXTENSION TO ACCOMMODATE PEDESTRIANS

4.1 Overview

Serving pedestrian demand within a signalized intersection requires addressing safety and delay concerns for all users of the intersection. While using several protected phases for pedestrians greatly increases safety, it can also impose a significant increase in vehicular delay. Emerging technologies for smart intersections and autonomous vehicles allow for fine-tuning SigPT and vehicle trajectories, providing solutions that maintain high safety standards while reducing overall vehicle and pedestrian delay.

The objective of this part of the research is to add and enhance algorithms to the original version of RIO in order to consider pedestrian presence. The revised algorithm can:

- Receive pedestrian arrival and presence information using existing and enhanced signal and detection infrastructure.
- Keep pedestrian right-of-way status updated in real-time at the intersection manager.
- Use pedestrian arrival and presence information in the SigPT optimization solver.
- Serve pedestrians safely at each intersection crosswalk.

RIO functions either in *real-time mode* or in *simulation mode*. The implementation of the real-time mode is made possible by receiving counts of pedestrians in waiting areas around the intersection through sensor fusion and a pedestrian push-button Graphical User Interface (GUI). The simulation mode uses pedestrian arrival rates at the waiting areas based on pre-built demand scenarios.

The management of pedestrian status and the consideration of pedestrians within the SigPT solver is accomplished by adapting the vehicle trajectory models currently implemented in RIO to replicate pedestrian movement. The safety of pedestrians as they cross is enhanced by reducing conflicting movements between vehicle and pedestrian movements to a minimum, and also by allowing the SigPT solver to adjust the pedestrian phase according to the time required for a pedestrian to complete the crossing at a comfortable speed.

RIO consists of two main modules: SigPT solver and trajectory solver. These modules will be referred to as RIO-SS (signal solver) and RIO-TS (trajectory solver), respectively, throughout this report.

4.2 Methodology development

Due to the way RIO manages individual entities within the network, and in order to prepare the programming environment to handle both present and future functionalities related to pedestrian

demand and behavior, it is desirable to manage pedestrians as individual entities within the network. In order to identify and supply pedestrian demand at an intersection, several new steps were added to RIO. These steps work both for real-time and for simulation modes, but with different sources of input (sensor information vs. pre-built scenarios):

- create a *data structure* to input and manage pedestrians as individual entities with their own "properties" (namely: arrival time, waiting area, accumulated delay);
- receive different types of *information inputs* (real-time mode: push-button GUI and counts from sensor fusion; simulation mode: pre-built scenarios with arrival times and locations) and translate it into pedestrian entries within RIO; and
- keep track of *pedestrian status and management*, and serve pedestrians during the corresponding pedestrian phase.

Each of these three steps is discussed in the following sections.

4.3 Pedestrian data structure

Before the addition of pedestrians, four main data structures were used in RIO:

- Vehicle class: used to manage CNV and CAV, storing vehicle properties and space-time parameters (speed, position within lane, trajectory).
- Lanes class: used to group vehicles by lane.
- Signal class: used to manage and optimize SigPT.
- Intersection class: store network geometry features and specifications such as speed limit, allowable signal phases, etc.

To accommodate pedestrians within RIO's framework, two of these data structures were modified: the first relates to the individual entries (formerly vehicles) and the second relates to the space occupied by individual entries (formerly lanes). When a vehicle enters the optimization region in its lane, the algorithm computes the earliest possible departure time for this vehicle. The SigPT optimizer (RIO-SS) then uses this information to determine the optimal phase order and length in order to serve the vehicles in the network.

Since the earliest departure time is the key parameter for assigning a signal phase to serve an entity in the network, the pedestrians were added as a subclass of the vehicle class. This approach has several benefits. First, it allows the SigPT solver to maintain the same linear program that solves the phase order and length, as long as a departure time is assigned to each pedestrian (this is discussed in more detail in subsection 4.5). Second, the input from either the real-time or the simulated mode can be translated into a common framework of individual pedestrian entries. Using the same structure for both real-time and simulation modes maintains RIO's consistency when running under different scenarios (with varying intersection geometry) and with different available inputs (each corner of an intersection may or may not have the

required infrastructure to provide counts of pedestrians and/or pedestrian push buttons). Also, such flexibility allows different real-time conditions to be tested using the simulation mode. Third, the modules that manage vehicle car-following, trajectory optimization, and overall movement can be used to manage pedestrians (they can either be deactivated or modified to suit pedestrian movement). For this implementation, pedestrian movement is not tracked, and therefore there is no need for pedestrian trajectory estimation. However, future expansions related to pedestrian trajectories can be easily plugged into RIO's mainframe through these module entry points.

Similar to vehicles grouped based on their arrival lanes, pedestrians are gathered according to the waiting area that is closest to the crosswalk they intend to cross. The waiting areas behave similarly to a lane class. In other words, they have geometric features (a delimited area close to the crosswalk observed by the sensor fusion to determine whether a pedestrian wishes to use a specific crosswalk) and, when inserted into the algorithm, they provide a reliable way of organizing and managing pedestrian entities occupying the same region within the network. Moreover, waiting areas that serve the same crosswalk are paired, allowing the signal controller and the SigPT solver to properly serve the demand of pedestrians moving in opposite directions and using the same sidewalk.

4.4 Input for pedestrian entries under real-time mode and under simulation mode

Three different input types can be used to add pedestrians to a network in RIO: two are used for the real-time mode and one for the simulation mode.

In real-time mode, pedestrians can be detected by RIO by either pressing the push button or through detection by video and/or radar. When the push button is pressed, the algorithm checks whether the waiting area that matches the push button already contains one or more pedestrians. If not, a single pedestrian is added to that waiting area. In this mode, the counts of pedestrians detected within each waiting area are sent from *SENSIBLE* – an algorithm that combines video, radar, and LiDAR inputs to provide tracking and counts of vehicles and pedestrians at the intersection. The counts of pedestrians detected within each waiting area are sent from *SENSIBLE* to RIO through a user datagram protocol (UDP) message. The counts provided by the sensor fusion are likely to have some noise, either missing some entries due to occlusion or overestimating the total number of pedestrians. To mitigate these potential errors, an auxiliary variable named *ghost pedestrians tracker* (GPT) is used. This time variable keeps track of inconsistencies that may occur between the sensor fusion count and RIO count. If this value reaches a time threshold (currently set to five seconds), then RIO adjusts its count to match the sensor fusion count.

The management of pedestrians in real time is handled as follows. At each time step, when RIO receives the counts message, the algorithm compares, for each waiting area, the count supplied by the sensor fusion with the total number of pedestrians already registered within the waiting area class. If the sensor fusion count is higher than RIO's count, the algorithm generates

a number of pedestrians equal to the difference between counts, adds those to the total number of pedestrians, and GPT is set to zero. If the sensor fusion count is lower than RIO's count, then the signal phase for that waiting area is checked. If the corresponding pedestrian phase is green, then the difference in counts is likely because pedestrians are currently crossing the road, thus no adjustment is made, and GPT is set to zero. However, if the signal phase is red, then the lower count provided by the sensor fusion could be caused because one or more pedestrians decided to leave the waiting area, or the sensor fusion missed one or more pedestrians. In either case, one second is added to the GPT. If the GPT reaches a pre-configured threshold, RIO has continuously kept a higher count of pedestrians than what is provided by the sensor fusion. Thus, the excess number of pedestrians is removed from the network, starting from the latest arrival. A diagram of how pedestrian counts are processed is presented in Figure 4-1.

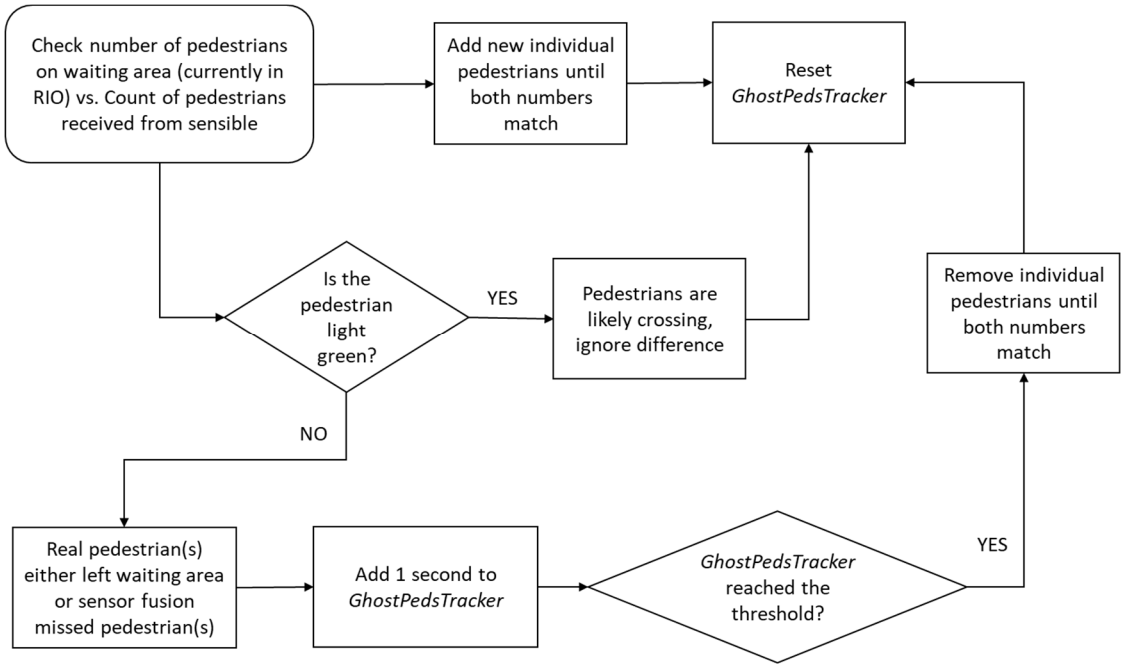


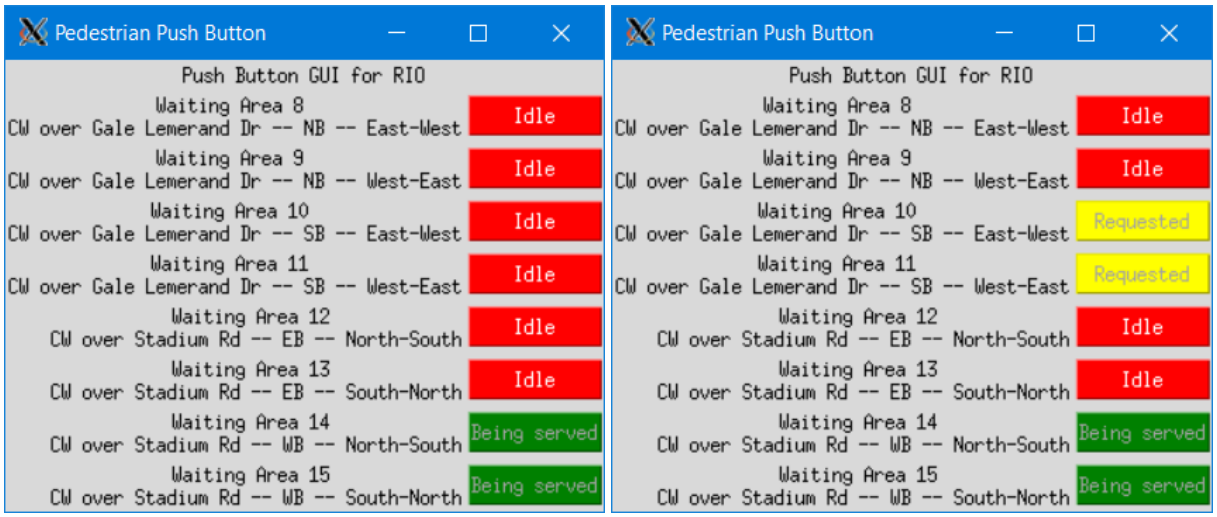
Figure 4-1: Pedestrian input and control based on automatic detection.

The value for the GPT threshold can be manually adjusted in order to accommodate the particular features and quality of sensing of each intersection. The threshold should be adjusted at an intersection based on the noise observed in the counts of pedestrians obtained with the LiDAR sensors, and values between 3 and 10 seconds are recommended. Generally, the lower the accuracy of the detection, and the lower the number of pedestrians at the intersection, the lower this threshold should be. Obstructions such as trees, weather conditions, and light intensity are examples of factors that would likely justify reducing the GPT threshold to generate a faster adjustment of the signal plan, given that these factors may obscure pedestrians or generate false positives. Intersections with high diagonal crossing of pedestrians would have pedestrians switching waiting areas more often. In this case, it may be more suitable to set the GPT threshold higher to avoid modifying the signal plan too often.

Our initial tests at the FDOT facility indicated that a GPT threshold of 5 seconds was appropriate for low pedestrian demand during clear and cloudy weather. It is worth reinforcing that only the binary condition (no pedestrians at waiting area *n* | one or more pedestrians at waiting area *n*) is currently used within RIO's method of assigning and optimizing the signal plan. The total count of pedestrians is used only for tracking pedestrian and network user delay.

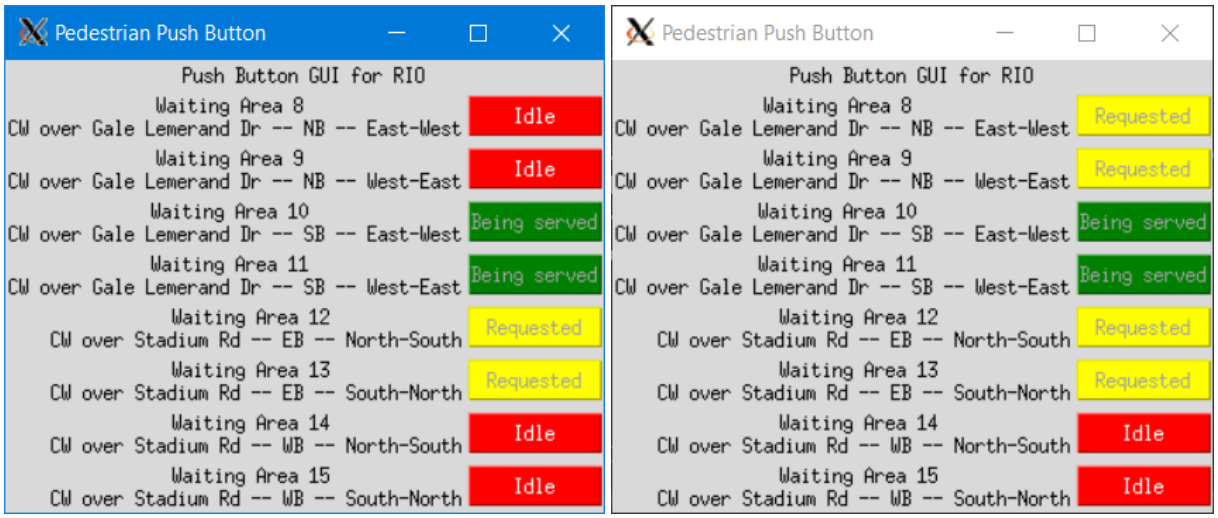
Since both the push button and the sensor fusion inputs of pedestrians account for the current number of pedestrians within the waiting area in RIO, these two input methods are not mutually exclusive, hence they can be used simultaneously. Moreover, should one of the methods need to be deactivated, RIO can continue to manage the intersection while still serving pedestrian demand. Since the LiDAR detection method relies on the pedestrian staying inside the boundaries of the waiting areas, push buttons may be kept at specific locations where pedestrians are likely to avoid the waiting area boundaries for any reason, including rain, sun, noise, safety, emissions, etc. In simulation mode, to allow the testing of the push-button functionality, a Graphical User Interface (GUI) was created, and a comma-separated values (csv) file is used to input both vehicles and pedestrians. While a vehicle entry requires several parameters to be provided (for example, speed, lane, length, and distance to stop bar), a pedestrian entry requires only the time of arrival and the waiting area where the pedestrian arrives.

Figure 4-2 and Figure 4-3 show four different scenarios using the pedestrian GUI. In the first scenario, RIO has just been initialized in real-time mode with an initial green phase serving waiting areas 14 and 15. Some seconds later, in the second image, the button for waiting area 11 is pressed, which represents a new pedestrian arrival. At this point, the button is changed to "request" mode and locked. Because waiting areas 10 and 11 are the opposite corners of the same crosswalk, the button for area 11 has the same modifications as button 10. The third image shows the push-button GUI after about 10 seconds. The phase for areas 14 and 15 is terminated, and the button becomes "idle" and can be pressed again. The phase for areas 10 and 11 is now green, while a new pedestrian arrived at area 12 and pressed the button, causing a new request to be sent to RIO. Approximately 2 seconds later, in the fourth image, another pedestrian arrives at area 9, resulting in another call. In this situation, the recently arrived pedestrian will have to wait for two phases (the one currently serving 10 and 11, and the next which will serve 12 and 13) before (s)he may cross.



(1) (2)

Figure 4-2: Pedestrian push-button GUI (1) idle and (2) with one walk light request.



(3) (4)

Figure 4-3: Pedestrian push-button GUI with (3) one and (4) two walk light requests.

4.5 Pedestrian status and management

When a pedestrian is added to the network through any of the input modes previously discussed, (s)he is immediately assigned to the appropriate waiting area, which means that (s)he is ready to cross. Within RIO's operation, the detection time for a pedestrian is equal to his/her earliest departure time. During each time step, the SigPT solver determines the best phase order and time duration to serve the demand of both vehicles and pedestrians based on the earliest departure time for each entity.

Similar to the SigPT optimization procedure previously implemented for vehicles, the *Pedestrian Minimum Green Time* (PMGT) is used to determine the minimum length of each phase:

- If the phase contains only pedestrian movements, the minimum length of this phase is the maximum value of PMGT among all pedestrian movements included in that phase.

- If the phase contains both vehicle and pedestrian movements, the minimum phase length is the maximum between (*i*) the PMGT for all pedestrian movements in the phase and (*ii*) the minimum green times for all vehicle movements in that phase.

For both cases, the minimum length of the phase must guarantee that any user (vehicle or pedestrian) of that phase can perform its movement safely. After the SigPT is updated, the status of each entity in the network is updated. Since pedestrians have no trajectory estimated for them, the decision of whether the pedestrian can cross the road or not is guided only by properties of the SigPT.

During the same time step that a pedestrian is added to the network, after the SigPT is updated, the algorithm checks whether the corresponding pedestrian phase is green. If yes, then the algorithm checks whether there is enough green time left for the pedestrian to safely cross. If both checks are positive, then it is assumed that the pedestrian has crossed the road, and thus (s)he is removed from the network. Otherwise, the pedestrian waits until his/her signal phase is green in order to exit the network. The diagram in Figure 4-4 provides an overview of how newly arriving pedestrians are managed after the SigPT optimization.

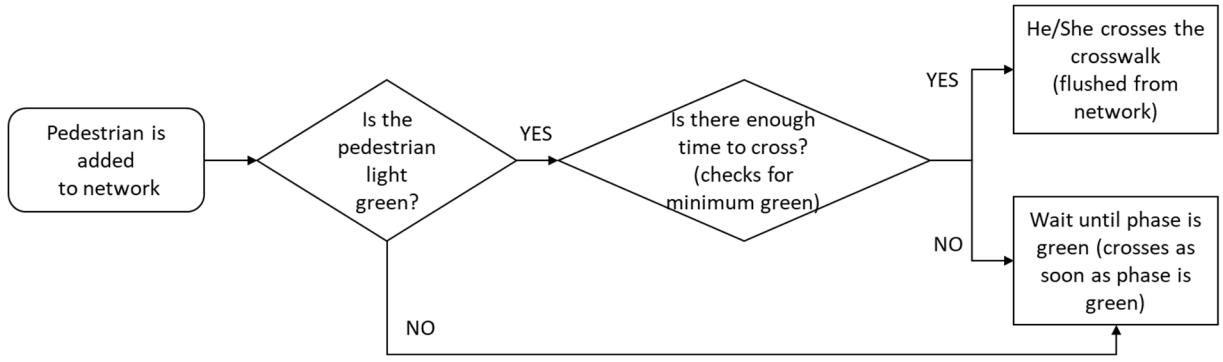


Figure 4-4: Management of pedestrian status within the network

The decision related to the safety of crossing is made using the PMGT parameter, which can be adjusted for each crosswalk. The PMGT is intended to be adjusted according to not only the width of the road, but also according to local preferences for expected pedestrian speed. The parameter and the length of the pedestrian phase should consider the prevailing conditions including the expected level of pedestrian demand at a particular location, as well as the expected

population types (for example, speeds would likely be different in a university campus vs. around a retirement community.)

4.6 Simulation testing

A case study was developed to evaluate how vehicle and pedestrian delay is impacted by RIO-SS under different demand patterns for both pedestrians and vehicles if compared to an actuated signal. The simulated intersection used for this analysis was modeled based on a real intersection located in the city of Gainesville, FL (Gale Lemerand Drive and Stadium Road). The lane coordinates were added to RIO using Universal Transverse Mercator (UTM) coordinates. There are two lanes per approach. Two waiting areas for pedestrians are present on each corner of the intersection, totaling eight waiting areas. Figure 4-5 provides a schematic of the network layout and the identifiers for each lane and waiting area.

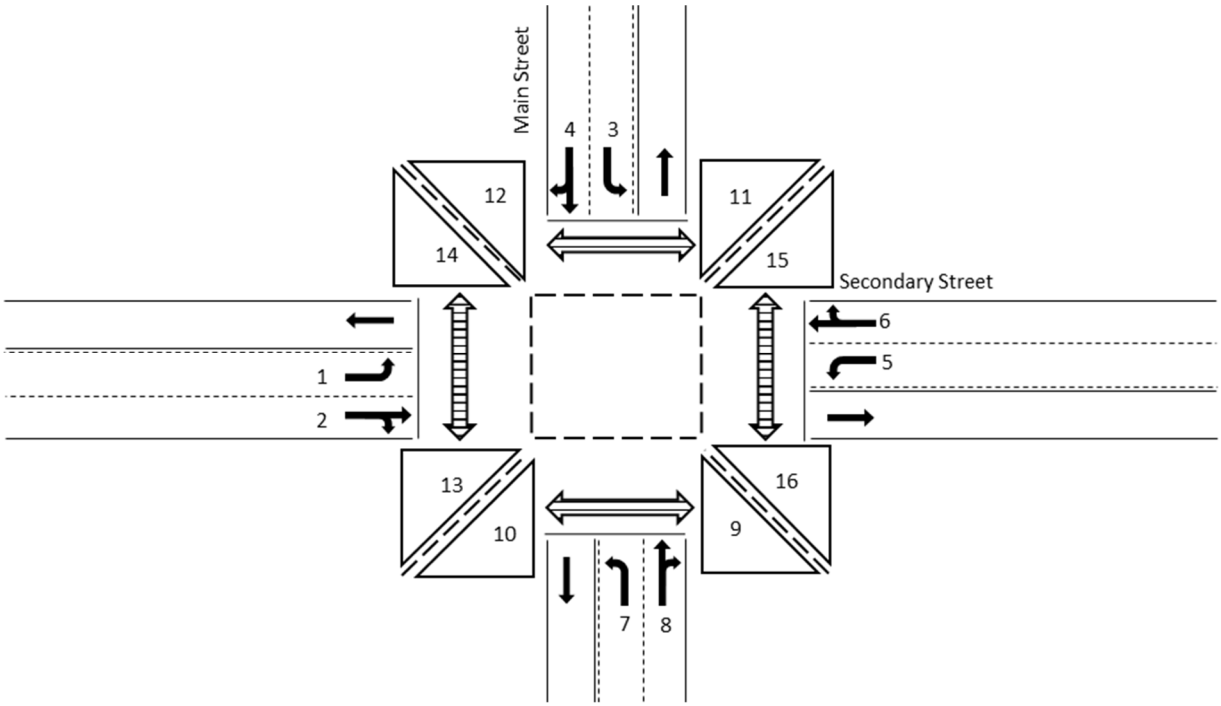


Figure 4-5: Case study intersection layout.

Twelve scenarios were designed to evaluate how the SigPT solver performs under different patterns of demand for pedestrians and vehicles. These scenarios are grouped in four sets according to four demand patterns, with three scenarios in each group.

In demand pattern 1, all vehicle and pedestrian approaches have the same demand on the main street, and the demands on the side street approaches are half the corresponding vehicle and pedestrian demands for the main street. In demand pattern 2, the pedestrian and vehicle demands are equal on all approaches for both main and side streets. These two sets of demand scenarios

were developed to determine whether RIO prioritizes approaches with higher demand or whether some approaches have higher delay in order to reduce the overall delay at the intersection. In demand pattern 3 all vehicle approaches have the same demand, and the pedestrian demand in all approaches is half the vehicle demand. Demand pattern 4 uses the same vehicle demand as pattern 3, but the pedestrian demand is twice the vehicle demand. These two sets of demand scenarios were developed to evaluate how delay varies for each mode (pedestrian/vehicle) as the other mode's demand changes.

Within the same demand pattern group, each scenario uses slightly different average headways for vehicles and/or pedestrians (which results in different levels of demand for each of the three scenarios in each demand pattern). A Poisson distribution was used with these average headways to generate stochastic vehicle and pedestrian arrivals throughout the simulation period. For each approach (lane or waiting area), the arrivals occur between time 0 seconds and the total simulation duration (a_N). The first arrival time (a_1) is obtained by using an exponential distribution to draw a random time value based on the headway for that approach, and the arrival time is bounded as $0 \leq a_1 \leq a_N$. The next arrival time must occur after the first, and it must also respect a minimum headway, defined as 2 seconds. Thus, another arrival time (a_2^*) is drawn with boundary set to $a_2^* \geq a_1$, and a_2 is defined as $a_2 = \max(a_2^*, a_1 + 2)$. Thus, the general model for generating arrivals is $a_n = \max(0, a_n^*, a_{n-1} + 2)$.. New arrivals are generated until $a_n \geq a_N$, because after this time the new arrivals would not be used by the simulation. Although pedestrians in real life are not bound by the headway restriction, this feature was kept the same for vehicles and pedestrians for the sake of simplicity. The input for each scenario is presented in Table 4-1, where the IDs for the lanes and waiting areas follow the same numbering as Figure 4-5.

Each simulation scenario was run for 1800 seconds. Because the vehicle generation follows a stochastic model, each run yields different results. Thus, multiple runs of each scenario were executed, and the resulting average delay and its standard deviation were obtained. The sample size was obtained based on the standard deviations of preliminary runs with five iterations for each scenario, in order to obtain results within a 95% confidence interval. Each scenario was executed 25 times using RIO-SS and another 25 times using actuated signal control.

The actuated signalization framework was developed and added to RIO specifically for this case study, to serve as a basis to determine the differences in delay obtained with RIO-SS compared to the state-of-the-art actuated control. Both signalization frameworks (RIO-SS and actuated control) use two phases. Each phase serves two vehicle approaches (northbound and southbound; eastbound and westbound) and also the parallel crosswalks. For example, in Figure 4-5, phase 1 serves lanes 1 and 3, and also waiting areas 9, 10, 11 and 12. The yellow and all-red times were set for both signal plans as 4 and 1 seconds, respectively. The PMGT was set to 7 seconds for all phases.

Table 4-1: Headway and hourly flow rate for each scenario.

			Demand Pattern 1			Demand Pattern 2				Demand Pattern 3			Demand Pattern 4		
		Demand Pattern Description	Different headways for main/side roads			Same headway for all approaches				Vehicle headway is lower than pedestrian's			Vehicle headway is higher than pedestrian's		
		Scenario ID	1	2	3	4	5	6		7	8	9	10	11	12
		Main St. Avg. Headway (s)	60	80	100	90	120	150	**Vehicle Headway (s)**	60	80	100	60	80	100
		Second. St. Avg. Headway (s)	120	160	200	90	120	150	**Pedestrian Headway (s)**	120	160	200	30	40	50
	Lane/Waiting Area ID	**Road Classification**	**Hourly Flow (veh/h or ped/h)**												
Lane	1	Main	30	22.5	18	20	15	12		30	22.5	18	30	22.5	18
	2	Main	30	22.5	18	20	15	12		30	22.5	18	30	22.5	18
	3	Secondary	15	11.25	9	20	15	12		30	22.5	18	30	22.5	18
	4	Secondary	15	11.25	9	20	15	12		30	22.5	18	30	22.5	18
	5	Main	30	22.5	18	20	15	12		30	22.5	18	30	22.5	18
	6	Main	30	22.5	18	20	15	12		30	22.5	18	30	22.5	18
	7	Secondary	15	11.25	9	20	15	12		30	22.5	18	30	22.5	18
	8	Secondary	15	11.25	9	20	15	12		30	22.5	18	30	22.5	18
Pedestrian Waiting Area	9	Main	30	22.5	18	20	15	12		15	11.25	9	60	45	36
	10	Main	30	22.5	18	20	15	12		15	11.25	9	60	45	36
	11	Main	30	22.5	18	20	15	12		15	11.25	9	60	45	36
	12	Main	30	22.5	18	20	15	12		15	11.25	9	60	45	36
	13	Secondary	15	11.25	9	20	15	12		15	11.25	9	60	45	36
	14	Secondary	15	11.25	9	20	15	12		15	11.25	9	60	45	36
	15	Secondary	15	11.25	9	20	15	12		15	11.25	9	60	45	36
	16	Secondary	15	11.25	9	20	15	12		15	11.25	9	60	45	36

Using RIO's customizable phasing structure, for this simulation pedestrian movements are protected, except those that conflict with permitted left turns. For RIO-SS simulations, two vehicle types are used. The first vehicle type are CAVs. These obtain optimal trajectories through RIO-TS, and are able to execute them using autonomy capabilities. When CAVs follow another vehicle, they aim to keep a short headway from the vehicle ahead. The second vehicle type are CNVs. It is assumed that these vehicles have access to an app indicating the optimal speed to cross the stop bar as the signal turns green. When CNVs follow another vehicle, the simulation assumes they do so according to the Gipps car-following model.

For actuated signal control simulations, vehicles are not connected to the intersection signal infrastructure and have no awareness of the SigPT beyond the current signal indication for that vehicle's lane. Leader CNVs in this scenario aim to drive at the speed limit while followers use the same Gipps car-following model used for RIO-SS. Autonomous vehicles are not used in this scenario.

In order to properly compare RIO-SS with the actuated signal plan, a unique set of traffic arrivals was generated for each iteration of the simulation. Twenty-five replications of each scenario described Table 4-1 were performed, each with a unique table of vehicle and pedestrian arrivals. Each replication uses a unique random seed (a starting number used to generate subsequent pseudorandom numbers) to generate the arrival times stochastically. For example, the set of arrivals #2 is generated using seed #102. Also, the set of arrivals #3, generated with seed #103, is different from the set of arrivals #2.

Autonomous vehicles' penetration rates of 0, 25, 50, 75 and 100% were tested for RIO-SS scenarios. Thus, the total number of simulation runs is (12 actuated scenarios × 25 repetitions) + (12 RIO-SS scenarios × 5 penetration rates × 25 repetitions) = 1800 iterations. For each combination of scenario, signalization and penetration rate, the average delay per lane/waiting area of the 25 iterations was obtained. Table 4-2 and Table 4-3 summarize the assumptions made for RIO-SS and actuated modes.

Table 4-2: Specifications for case study – vehicles.

Parameter group	Parameter	Value	
		RIO-SS	Actuated
Vehicle specifications	Vehicle length	4.8 meters	
	Maximum speed	8.94 m/s (20 mph)	
	Maximum acceleration	2 m/s²	
	Maximum deceleration	-6 m/s²	
	Minimum CAV headway	1.5 seconds	
	Minimum CNV headway	2 seconds	
	Min distance to stop bar (for trajectory re-optimization)	5 meters	
	Lag on green	1 second	

Table 4-3: Specifications for case study – SigPT.

Parameter group	Parameter	Value	
		RIO-SS	Actuated
Signal timing	Initial green	5 seconds	5 seconds
	Passage green	-	3.5 seconds
	Maximum Green	25 seconds	30 seconds
	Pedestrian Walk	7 seconds	7 seconds
	Yellow	4 seconds	4 seconds
	All Red	1 second	1 second
	Upstream Detectors position	-	25 meters (all approaches)
Signal phasing	Phase 1	NB and SB	
	Phase 2	EB and WB	

4.7 Results and analysis

Table 4-4 and Table 4-5 show the differences and ratios, respectively, between the average delays obtained with the actuated signal plan and with the RIO-SS, for each of the 12 scenarios, at different penetration rates. The results are shown separately for vehicles, pedestrians, and users. Each vehicle is equal to 1.2 users, while each pedestrian is equal to 1 user. The total number of vehicles and pedestrians simulated within 1800 seconds is shown for the sake of comparison of the different demands in each scenario.

Two statistical tests were performed. The significance of the average network delay of each scenario, given the sample size of 25 iterations, was tested assuming margin of error of 10% and confidence interval of 95%. All the actuated scenarios results are statistically significant. Some of RIO-SS scenarios' results are not significant according to this test, and they are marked with a double underline. The statistical significance of the difference between RIO-SS and actuated results were tested using a p-test with confidence interval of 95%, with significant results highlighted in bold (all scenario results passed the test). The delay difference is calculated as *actuated delay* - *RIO-SS delay*, results shown in seconds. Thus, positive results mean that RIO provides a network delay reduction. These results are highlighted with green, while negative results are highlighted with red. Results that failed either statistical test are highlighted with gray to be easily discernable.

Table 4-4: Delay differences between RIO-SS and actuated signal control.

Demand Pattern Description	Demand Pattern 1 Different headways for main/side roads			Demand Pattern 2 Same headway for all approaches			Demand Pattern 3 Vehicle headway is lower than pedest.'s			Demand Pattern 4 Vehicle headway is higher than pedest.'s		
Scenario ID	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
Total sim. vehicles	90	68	54	80	60	48	120	90	72	120	90	72
Total sim. pedestr.	90	68	54	80	60	48	60	45	36	240	180	144
CAV Pen. Rate on RIO	**Vehicle delay difference (Actuated Delay - RIO Delay) [seconds]**											
0%	**11.6**	**10.8**	**9.0**	**11.3**	**11.6**	**7.7**	**11.3**	**9.1**	**9.1**	**14.1**	**13.2**	**13.2**
25%	**12.4**	**11.5**	**10.6**	**12.0**	**11.3**	**9.2**	**11.3**	**10.5**	**10.4**	**14.3**	**14.4**	**13.7**
50%	**13.6**	**11.7**	**10.3**	**12.7**	**12.2**	**10.2**	**12.3**	**11.8**	**10.6**	**14.5**	**15.8**	**14.2**
75%	**14.1**	**12.6**	**11.2**	**13.8**	**12.3**	**11.6**	**12.4**	**11.4**	**11.0**	**14.8**	**15.4**	**15.8**
100%	**14.6**	**12.7**	**11.9**	**11.4**	**13.7**	**10.6**	**10.3**	**12.3**	**12.1**	**15.1**	**15.9**	**17.4**
CAV Pen. Rate on RIO	**Pedestrian delay difference (Actuated Delay - RIO Delay) [seconds]**											
0%	**-2.3**	**-1.9**	**-3.3**	**-2.0**	**-2.2**	**-3.9**	**-2.1**	**-3.2**	**-3.9**	**-2.1**	**-1.7**	**-1.6**
25%	**-2.3**	**-2.0**	**-2.9**	**-2.2**	**-2.4**	**-3.4**	**-2.3**	**-3.5**	**-3.4**	**-1.9**	**-1.5**	**-1.9**
50%	**-2.0**	**-1.8**	**-2.7**	**-1.9**	**-2.0**	**-2.3**	**-2.0**	**-2.7**	**-3.2**	**-2.0**	**-1.4**	**-1.3**
75%	**-1.9**	**-1.5**	**-3.4**	**-1.8**	**-2.7**	**-3.1**	**-1.6**	**-2.5**	**-3.7**	**-2.1**	**-1.4**	**-1.1**
100%	**-1.4**	**-1.3**	**-2.6**	**-1.6**	**-2.3**	**-3.3**	**-1.8**	**-2.5**	**-3.3**	**-2.1**	**-1.2**	**-0.9**
CAV Pen. Rate on RIO	**User delay difference (Actuated Delay - RIO Delay) (1 ped = 1 user; 1 veh = 1.2 users) [seconds]**											
0%	**5.8**	**5.6**	**3.8**	**5.8**	**5.9**	**2.7**	**8.3**	**6.3**	**6.0**	**4.2**	**4.2**	**4.2**
25%	**6.3**	**5.9**	**5.0**	**6.1**	**5.6**	**3.8**	**8.3**	**7.2**	**7.2**	**4.4**	**4.8**	**4.2**
50%	**7.2**	**6.2**	**4.8**	**6.6**	**6.3**	**4.9**	**9.2**	**8.6**	**7.4**	**4.5**	**5.4**	**4.8**
75%	**7.5**	**6.8**	**5.0**	**7.4**	**6.0**	**5.4**	**9.4**	**8.3**	**7.5**	**4.5**	**5.3**	**5.5**
100%	**8.1**	**7.0**	**5.9**	**6.0**	**7.0**	**4.7**	**7.7**	**9.0**	**8.6**	**4.7**	**5.6**	**6.4**

Table 4-5: Ratio of delays between SigPT solver and actuated signal control.

Demand Pattern Description	Demand Pattern 1 Different headways for main/side roads			Demand Pattern 2 Same headway for all approaches			Demand Pattern 3 Vehicle headway is lower than pedest.'s			Demand Pattern 4 Vehicle headway is higher than pedest.'s		
Scenario ID	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
Total sim. vehicles	90	68	54	80	60	48	120	90	72	120	90	72
Total sim. pedestr.	90	68	54	80	60	48	60	45	36	240	180	144
CAV Pen. Rate on RIO	**Ratio of vehicle delay [1- (RIO Delay/Actuated Delay)]**											
0%	**48%**	**45%**	**37%**	**46%**	**46%**	**33%**	**46%**	**37%**	**37%**	**57%**	**54%**	**53%**
25%	**52%**	**48%**	**44%**	**48%**	**45%**	**39%**	**46%**	**43%**	**42%**	**58%**	**59%**	**55%**
50%	**57%**	**49%**	**42%**	**51%**	**48%**	**43%**	**50%**	**48%**	**43%**	**59%**	**65%**	**57%**
75%	**59%**	**53%**	**46%**	**56%**	**49%**	**49%**	**50%**	**47%**	**44%**	**60%**	**63%**	**63%**
100%	**61%**	**53%**	**49%**	**46%**	**54%**	**45%**	**42%**	**50%**	**49%**	**61%**	**65%**	**70%**
CAV Pen. Rate on RIO	**Ratio of pedestrian delay [1- (RIO Delay/Actuated Delay)]**											
0%	**-31%**	**-24%**	**-45%**	**-27%**	**-29%**	**-49%**	**-26%**	**-41%**	**-51%**	**-27%**	**-21%**	**-22%**
25%	**-30%**	**-25%**	**-39%**	**-30%**	**-32%**	**-43%**	**-28%**	**-45%**	**-45%**	**-24%**	**-18%**	**-25%**
50%	**-26%**	**-22%**	**-37%**	**-26%**	**-27%**	**-30%**	**-25%**	**-35%**	**-42%**	**-25%**	**-17%**	**-18%**
75%	**-24%**	**-19%**	**-46%**	**-24%**	**-36%**	**-39%**	**-19%**	**-32%**	**-49%**	**-26%**	**-17%**	**-15%**
100%	**-18%**	**-16%**	**-35%**	**-21%**	**-30%**	**-42%**	**-22%**	**-32%**	**-43%**	**-26%**	**-15%**	**-11%**
CAV Pen. Rate on RIO	**Ratio of user delay [1- (RIO Delay/Actuated Delay)] (1 ped = 1 user; 1 veh = 1.2 users)**											
0%	**32%**	**30%**	**21%**	**31%**	**31%**	**15%**	**37%**	**28%**	**27%**	**28%**	**28%**	**28%**
25%	**35%**	**32%**	**27%**	**33%**	**29%**	**21%**	**37%**	**33%**	**32%**	**29%**	**32%**	**28%**
50%	**40%**	**34%**	**26%**	**36%**	**33%**	**27%**	**41%**	**39%**	**33%**	**29%**	**36%**	**32%**
75%	**42%**	**37%**	**27%**	**40%**	**32%**	**30%**	**42%**	**37%**	**34%**	**30%**	**35%**	**37%**
100%	**45%**	**38%**	**32%**	**32%**	**37%**	**26%**	**34%**	**41%**	**39%**	**31%**	**37%**	**43%**

The delay results for a penetration rate of zero indicate that RIO can provide substantial benefits for drivers in conventional vehicles that are connected to the signalization infrastructure. RIO's awareness of the vehicles' position and status in the network allows it to fine-tune the SigPT to prevent lost time. This benefit is possible under the assumption that the driver will adopt the average speed determined by RIO and sent to the driver through a phone app. Although the pedestrians suffer an increase of average delay ranging between 1.6 – 3.9 seconds, the benefits provided for the vehicles result in user delay reductions between 7.7 – 14.1 seconds, with the best results occurring within the demand patter 4 (scenarios 10,11,12) where the vehicle demand is half of the pedestrian demand.

As the penetration level increases, the delay for vehicles in RIO-SS decreases slightly. Meanwhile, pedestrian delay is influenced by the differences in the vehicle and pedestrian demands. Overall, the pedestrians seem to incur less delay in scenarios with simultaneously high pedestrian demand and low vehicle demand (scenarios 11 and 12). These scenarios, along with scenario 1, also show a tendency for pedestrian performance to be less affected by the RIO-SS optimization as the penetration rate of CAV increases.

The user delay is reduced in every scenario. Since vehicles can have more than one user, the scenarios with higher vehicle demand (demand pattern 3) show higher user delay reduction.

Six scenarios of 100% penetration rate and two scenarios of 75% penetration rate provided results that are not statistically significant. This occurs mostly for scenarios where the number of CAV is relatively higher than the total of CNV and pedestrians (CAV/(CNV + *pedestrians*)) as shown in Table 4.3, where the non-significant scenarios previously tested are once more marked with double underline. Most scenarios where this ratio is equal to 1 or higher are also the scenarios with delay results that are non-statistically significant. Since the signal optimization logic of RIO-SS provides good results for low penetration rates, the logic of trajectory optimization of RIO-TS is currently being studied to address this issue.

Table 4-6: Ratio of CAV over the total of CNV and pedestrians.

Demand Pattern Description	**Demand Pattern 1** Different headways for main/side roads			**Demand Pattern 2** Same headway for all approaches			**Demand Pattern 3** Vehicle headway is lower than pedest.'s			**Demand Pattern 4** Vehicle headway is higher than pedest.'s		
Scenario ID	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
Total sim. vehicles	90	68	54	80	60	48	120	90	72	120	90	72
Total sim. pedestr.	90	68	54	80	60	48	60	45	36	240	180	144
CAV Pen. Rate on RIO	**CAV/(CNV + pedestrians)**											
0%	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
25%	0.14	0.14	0.14	0.14	0.14	0.14	0.20	0.20	0.20	0.09	0.09	0.09
50%	0.33	0.33	0.33	0.33	0.33	0.33	0.50	0.50	0.50	0.20	0.20	0.20
75%	0.60	0.60	0.60	0.60	0.60	0.60	1.00	1.00	1.00	0.33	0.33	0.33
100%	1.00	1.00	1.00	1.00	1.00	1.00	2.00	2.00	2.00	0.50	0.50	0.50

4.8 Conclusions

Several changes were made to RIO's framework that relate to the inclusion of pedestrians within the SigPT solver. These changes include inputting and managing pedestrian entries in both simulation and real-time modes and allowing RIO's integration with both present-time common signalized intersections as well as with novel video, radio and LiDAR-based systems that track and count vehicle and pedestrian demand.

A case study was developed using different traffic demand patterns that were managed by RIO's upgraded SigPT solver. These demand patterns were also simulated assuming actuated control with conventional vehicles, and the delay results were compared. The results show that the SigPT solver can decrease vehicle delay with minimal impact to pedestrian delay, while achieving significant overall network delay reduction even when the traffic stream consists of conventional vehicles only.

5 FIELD DEPLOYMENT AT THE TERL AND RESULTS

5.1 Overview

Throughout this project we have further developed a system that is capable of detecting traffic through different sensors and use this information to jointly optimize vehicle's trajectories and signal phase and time. The main components of this system are described below.

Sensible is a real-time multi-sensor fusion software framework that was developed originally as part of the NSF-sponsored UFTI AVIAN project (NSF Award 1446813). It assembles information from several sensors and fuses it to provide vehicle arrival information to our intersection controller algorithm, which is described later in this section. Sensible supports inputs from commercial traffic sensors such as the Econolite Autoscope, Smartmicro radar, and Cohda DSRC for V2X. It contains sensor drivers and tracking algorithms for each sensor type, as well as fusion algorithms for combining all inputs into a single information stream. This information is provided to an intelligent intersection control algorithm which optimizes signal control based on vehicle arrival information. In this project, we developed sensor drivers and detection algorithms for LiDAR-based pedestrian detection near the intersection area (Chapter 2). In this part of the research we study the overall performance of the entire sensor suite, both qualitatively and quantitatively.

The Object Tracking Message (OTM) is a proposed dedicated short-range communication (DSRC) message structure developed in this project (Chapter 3) to allow source-independent communication of detected dynamic objects. The header of the message describes the origin of the source's Easting-Northing measurement frame, the number of objects being tracked, and the number of objects. The object structure of the message reports the detection time, object classification, pose, bounding box, velocity, and the uncertainty of those measurements for a single object, and is repeated in the body of the message for each detected object up to a machine-specified maximum.

The Real-time Intersection Optimizer (RIO) is an algorithm developed by UFTI (NSF Award 1446813) to optimize Signal Phase and Timing (SigPT) and vehicle trajectories. The original version of RIO receives real-time positioning of CNVs or CAVs, determines SigPT in order to better serve the current demand in the intersection, and sends back optimized trajectories such that vehicles depart from the stop bar at the saturation rate and at the maximum possible speed. In this project, RIO was upgraded by adding the presence of pedestrians within the signal optimization framework, with the pedestrian detection being provided either through the LiDAR method discussed in Chapter 2 or through a virtual pedestrian push button provided through a Graphical User Interface (GUI).Once RIO optimizes the trajectory of a CNV, it sends this trajectory information to the driver through a phone app. This phone app was developed by UFTI (NSF Award 1446813) and it provides drivers with a recommended speed in order to match the optimized trajectory. It also shows a time countdown of when the driver is expected to cross the stop-bar according to the optimized trajectory.

The research team conducted a series of field tests to verify and evaluate the modifications made to our system's components which are described in previous chapters. Thus, the objectives of the testing are as follows:

1. Execute a series of test scenarios where a varied number of vehicles and pedestrians must be detected and served as they approach the signalized intersection under different circumstances.
2. Collect quantitative data related to each of the system's components during the execution of the scenarios.
3. Collect qualitative data and drivers' feedback based on their usage of the intersection, both with and without access to the phone app developed previously by the UFTI and refined during this project.
4. Determine whether the system components discussed in Chapters 2-4 successfully perform their specific roles during the test deployment.
5. Evaluate the system's ability to serve vehicle and pedestrian demand.

The next section describes the test site, the system and its functionality, followed by a description of each of the scenarios tested along with the respective quantitative results. The fourth section discusses qualitative observations, the fifth section provides an overview of the hardware and software functionalities, and the last section summarizes the research team's conclusions and recommendations.

5.2 Description of site and system components

This subsection provides an overview of the test site (TERL) and describes the signal control optimization software and hardware components. Figure 5-1 shows a top-down view of the TERL intersection used for testing, with an overlay containing the specifications used in the signal control optimization algorithm. The lane lengths presented closely match the DSRC detection range. The red polygons represent the regions used as pedestrian waiting areas; pedestrians waiting within these two polygons were detected by the LiDAR and sent to RIO. The virtual stop bar for the northbound (NB) approach was moved 15 meters upstream from the real stop bar in order to have the stop bar within the video-camera range. The westbound approach was not used for this test as it is not instrumented with video and other sensors.

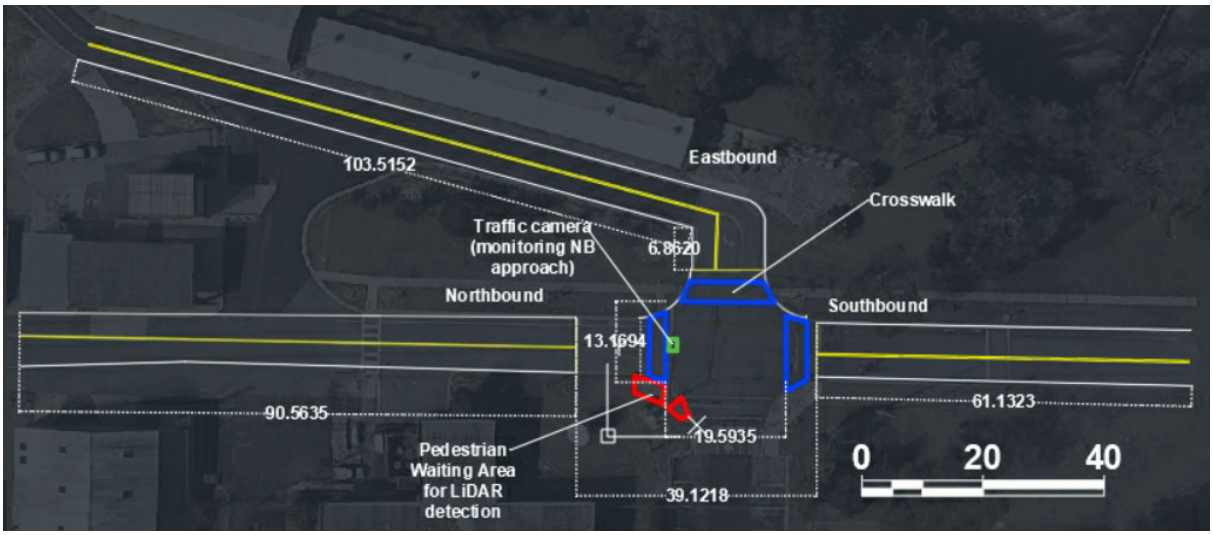


Figure 5-1: Top-down view of the TERL intersection layout configuration.

5.2.1 Vehicle and Pedestrian Traffic Used for Testing

The research team used two connected vehicles and one conventional vehicle for the tested scenarios. Drivers were instructed to maintain a speed of 15 mph whenever possible. Conventional vehicle drivers were instructed to drive as they normally would, respecting the signal indications. Drivers of connected vehicles were in possession of the phone app and therefore received a recommended speed and departure time as they were approaching the traffic signal, in order to minimize their delay through the stop bar. They were instructed to observe the phone app's recommendation but still make their decisions based on their own judgement.

The pedestrian traffic consisted of two pedestrians, with one person requesting Walk time by standing on the waiting area and getting detected by the LiDAR, while the other pedestrian was added to the system using the push-button GUI.

5.2.2 Sensible and Autoscope Camera

We used the Sensible multi-sensor tracking framework to obtain vehicle arrivals which were sent as input to the RIO intersection control algorithm. Sensible is developed in parallel to this FDOT project under an NSF grant. Sensible can process and fuse DSRC, traffic camera, and traffic radar inputs in real-time. At the TERL we had access to one Autoscope camera mounted on the mast arm facing the NB approach. We used two on-board units (OBUs) as sources of DSRC data and fused the video and DSRC tracking information. We did not have a radar (which can be supported by Sensible) present at this intersection (we have conducted field experiments that include radar at the intersection of Gale Lemerand and Stadium Rd. in Gainesville, FL). Additional sensors generally increase the reliability and accuracy of the vehicle arrival information sent to RIO.

5.2.3 LiDAR for Pedestrian Identification

During the TERL testing we combined the outputs of LiDAR detection (Chapter 2) with the outputs of Sensible to produce the inputs to the RIO intersection control algorithm. We mounted the LiDAR on a tripod and placed it near the mast arm base (Figure 5-2) so that it has a clear view of areas where pedestrians wait to cross. Note that in this study, the LiDAR *only* detects pedestrians and therefore does not need to be "fused" with the DSRC and video information obtained by Sensible.

Figure 5-2: LiDAR setup.

5.2.4 RIO

The Real-time Intersection Optimizer (RIO) is an algorithm developed by the UFTI (under NSF Award 1446813) to optimize Signal Phase and Timing (SigPT) and vehicle trajectories. RIO receives real-time positioning of Connected Vehicles (CNV) or Connected Autonomous Vehicles (CAV), determines SigPT to better serve the current demand in the intersection, and sends back optimized trajectories such that vehicles depart from the stop bar at the saturation rate and at the maximum possible speed. The research team modified RIO (Chapter 4) to serve pedestrian demand using either a push-button (currently provided through a Graphical User Interface – GUI) or counts provided by a LiDAR installed near the crosswalks.

In order to determine the best SigPT, a set of feasible phases have to be pre-configured in RIO. For this test, three phases were created, one for each of the three vehicular approaches. In addition, each phase provides the right-of-way to the crosswalk located to the right of the

approach. For example, the eastbound approach is served in the same phase as the crosswalk along the NB approach.

Table 5-1 summarizes all configurable parameters used in RIO during the tests.

Table 5-1: Values adopted for configurable parameters in RIO.

Parameter		Value
Signal Parameters	Minimum vehicle green	5 seconds
	Maximum green	25 seconds
	Pedestrian minimum walk	7 seconds
	Yellow	3 seconds
	All Red	1 second
Trajectory planning parameters	Maximum speed	6.7 m/s (15 mph)
	Minimum CNV headway	2 seconds
	Pedestrian ghost time threshold	3 seconds
	Lag of green start (LoG)	3 seconds
	Minimum distance to stop bar	10 meters

RIO communicates with the signal cabinet through an Econolite Cobalt signal controller using Simple Network Management Protocol (SNMP). The structure developed for this communication uses the National Electrical Manufacturers Association (NEMA) TS2 standard. Thus, any NEMA TS2-compliant signal controller could be potentially connected to RIO with little to no changes needed. The parameters of each phase (signal time and allowable movements) are manually set in the signal controller to match RIO specifications. RIO keeps track of the phase plan. When the current phase needs to go into yellow, RIO sends vehicle and/or pedestrian calls for the next phase to the signal controller. When the signal controller receives these calls, it proceeds to change the signal status based on its own SigPT configuration and internal logic. Therefore, RIO does not communicate the SigPT to the signal controller directly, but it uses detection inputs to adjust the signalization accordingly. Also, RIO and the signal controller may become slightly out of sync because RIO may only send a message to the signal controller at the end of each iteration of a RIO loop. This loop frequency is an adjustable parameter and has been set to 1 Hz for this field deployment. Thus, the communication method results in slight differences between the optimized RIO signal timings and the resulting signal timings provided by the signal controller.

5.3 Connectivity and data flow among system components

Figure 5-3 shows how the different system components are connected.

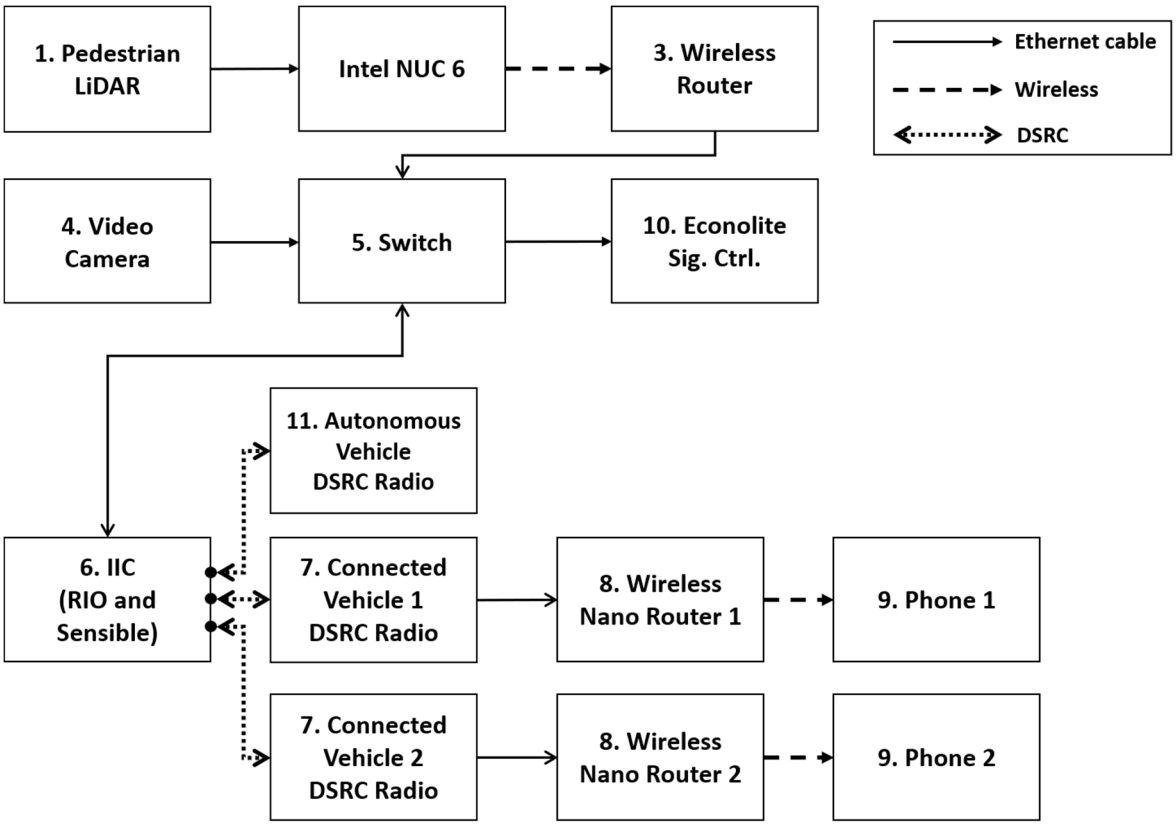


Figure 5-3: Connection setup of system components.

Box 1 consists of the pedestrian LiDAR deployed at the intersection sidewalk. The pedestrian LiDAR data is managed in Box 2 through an algorithm running on an Intel NUC 6 (NUC stands for Next Unit of Computing). In the NUC, an algorithm runs on an infinite loop. This algorithm does the following, in order: detect pedestrians through cluster analysis, determine whether a pedestrian is within one of the waiting areas, count the total number of pedestrians inside each waiting area, and send counts of pedestrians to the Intelligent Intersection Controller (IIC), which consists of a Linux computer running both RIO and Sensible. The pedestrian counts are sent through user datagram protocol (UDP) messages, which are strings of text in the form: Waiting Area ID: total pedestrians in this area. The Intel NUC 6 connects to the main network of our system wirelessly, allowing the pedestrian LiDAR to be deployed across the road from where the signal cabinet is located. Thus, the UDP messages go through a wireless router (Box 3) and a switch (Box 5) before arriving at the IIC.

Box 4 consists of the video camera which is connected by wire to the IIC through an intermediate switch. The video camera is aimed at one of the intersection approaches.

Boxes 7 consist of the connected infrastructure (DSRC radio) onboard of each vehicle. Each radio sends the vehicle speed, direction, and GPS position to the roadside unit (RSU) DSRC radio in the IIC.

Thus, the IIC (Box 6) receives input data for the signal optimization that consists of pedestrian detection information through the LiDAR (Boxes 1-2), vehicle detection through video-camera processing (Box 4) and connected vehicle-related information (Box 7). Two algorithms run at the IIC: RIO and Sensible. Sensible collects video-camera feed and connected vehicle information and generates a fused dataset of the vehicle traffic in the intersection. The counts of pedestrians are already processed when they arrive at the IIC, so they receive no further treatment. The fused vehicle dataset and counts of pedestrians are used by the IIC to jointly optimize vehicle trajectories and signal plan.

The IIC sends the optimal trajectory of each connected vehicle to the respective onboard DSRC radio (Box 7). The radio forwards the optimal trajectory to an onboard smartphone (Box 9) through a wireless nano router (Box 8). The smartphone has an app that displays recommendations of speed and departure time to the driver.

The optimal signal plan is sent from the IIC (Box 6) to the Econolite signal controller (Box 10) passing through the switch (Box 5).

The position and orientation of objects detected by Sensible are sent to the IIC and are then transmitted to the autonomous vehicle's onboard radio (Box 11). The on-board radio forwards the message to the autonomous vehicle's computer where it is used to expand its detection range.

5.4 Test scenarios and their results

The scenarios are divided in four groups. Group 1 consists of scenarios where vehicles utilize one approach and there is no pedestrian demand. In Group 2, vehicles arrive from two approaches in each scenario. Group 3 also uses two approaches for vehicle demand, but the vehicle arrivals to the intersection from the two approaches are staggered. In Group 4 vehicles arrive from one approach and there are different pedestrian demands at the intersection crosswalks. The following subsections provide detailed descriptions of each group of scenarios, the objectives of testing for each of these groups, and the results related to traffic behavior. Each scenario was replicated three times. Only the best of the runs for each scenario is discussed (i.e., we conducted multiple experiments and had to debug the system components to achieve the results described here). We only discuss specific failures or other relevant behavior that are worth mentioning. Additional data and observations obtained from these scenarios are further discussed in later sections of this report.

5.4.1 Group 1 – One-approach scenarios

For scenarios 1 and 2 only the NB approach is utilized (Figure 5-1). There is a video feed of arriving vehicles available on this approach, therefore it is possible to test how our system

behaves when managing conventional vehicles (video-feed only) as well as connected vehicles (DSRC and, optionally, video feed) arriving from a single approach. The objective of Group 1 scenarios is to determine how the order of arrivals of different vehicle types (conventional, connected) influences the performance of Sensible and RIO.

5.4.1.1 Scenario 1 – Platoon leader is a connected vehicle

For Scenario 1, two connected vehicles (with an on-board DSRC unit) are followed by a conventional vehicle at the NB approach (Figure 5-4).

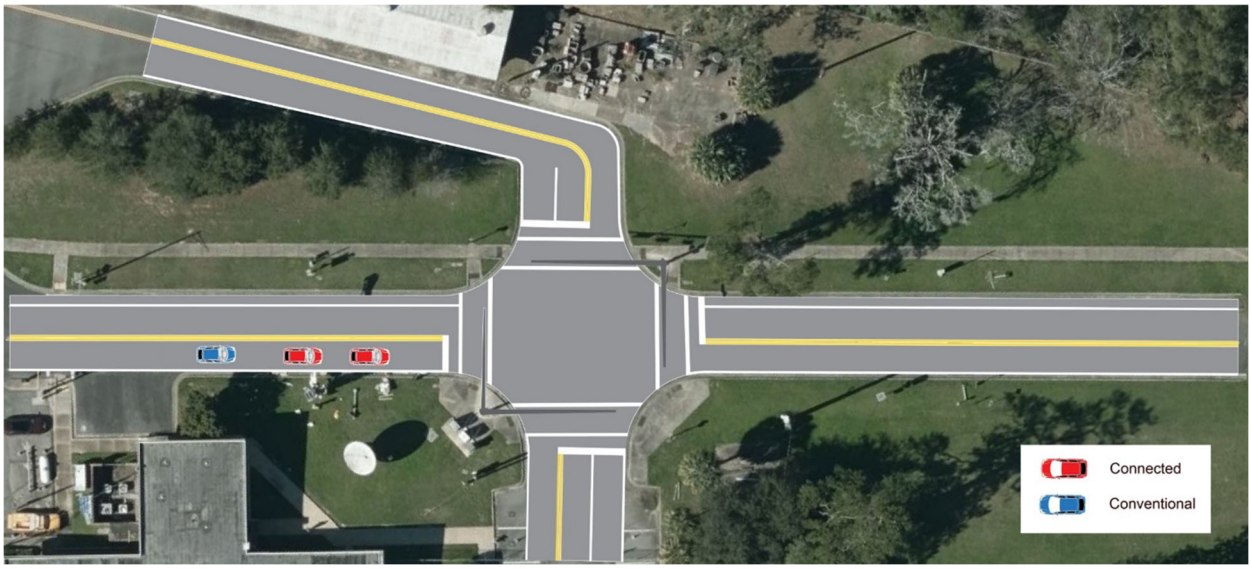


Figure 5-4: Scenario 1 illustration.

Figure 5-4 provides a top-down schematic view of this scenario's layout, while Figure 5-5 shows the time-space diagram for one of the replications of this scenario. This, and all subsequent time-space diagrams, have the same structure, as follows. The continuous lines represent a vehicle trajectory, while the dashed lines with a lighter color represent the trajectory recommended by RIO when the vehicle was at a specific distance from the stop bar. As shown, the recommended trajectories adjust as the vehicle continues its path and deviates from the originally anticipated trajectory. A black dashed horizontal line represents the minimum distance to the stop bar, beyond which RIO does not optimize the vehicle's trajectory. This is a user-defined threshold used to determine how RIO manages a vehicle: if the vehicle is upstream of that threshold RIO updates the vehicle's recommended trajectory and may readjust SigPT according to the overall network status; if the vehicle is downstream from this threshold the vehicle no longer receives a recommended trajectory, and RIO protects SigPT to ensure that this vehicle will reach the stop bar during the green interval if it continues following the recommended trajectory.

In Scenario 1, since there is no demand in the other approaches, RIO begins to change phases to serve the NB approach as soon as a vehicle is detected. This can be observed in the time-space diagram of Figure 5-5. RIO receives the Connected Vehicle 1 DSRC message at

20:17:21 UTC time, and immediately calls for the current phase to end. The yellow time is set as 3 seconds, counting from the next time-step (20:17:ff22). An all-red time of 1 second concludes the previous phase, and the NB phase begins.

As shown in Figure 5-5 the trajectory appears to move slightly backwards when the fusion of DSRC and video occurs at a distance of 30 to 50 meters. This can be observed at 20:17:30 for vehicle 1 and at 20:17:32 for vehicle 2. This is a correction likely caused by a small difference between the DSRC lat/lon coordinates and the Sensible video coordinates when translated to lat/lon.

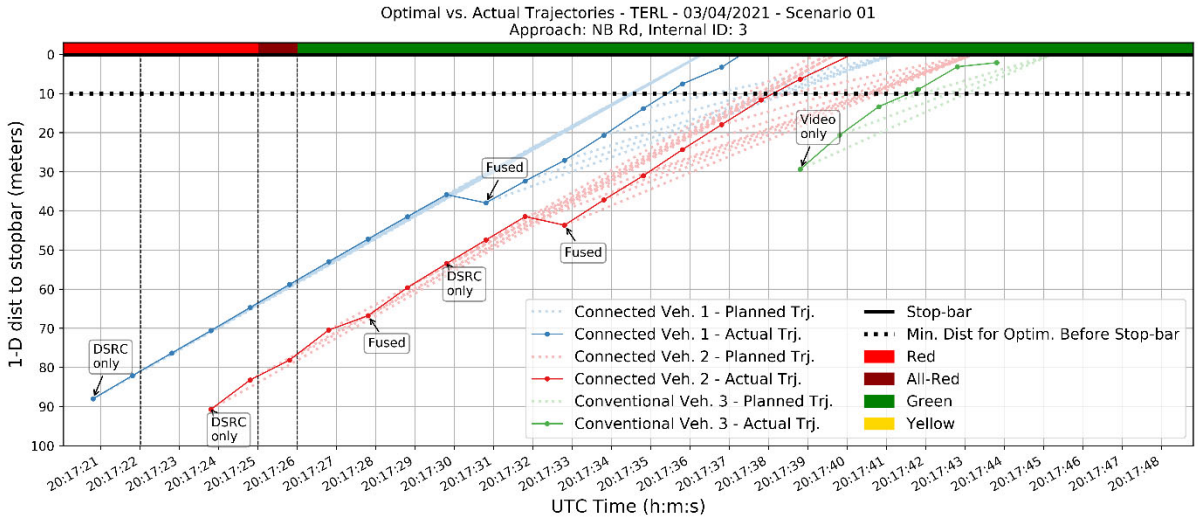


Figure 5-5: Time-space diagram for Scenario 1.

We found that RIO successfully manages to provide suggested trajectories for all vehicles in the network despite false positive vehicle tracks due to sensing errors. Figure 5-6 shows a trajectory of a false positive conventional vehicle (6) intersecting the trajectory of vehicle 2, which was already fused at the intersecting points.

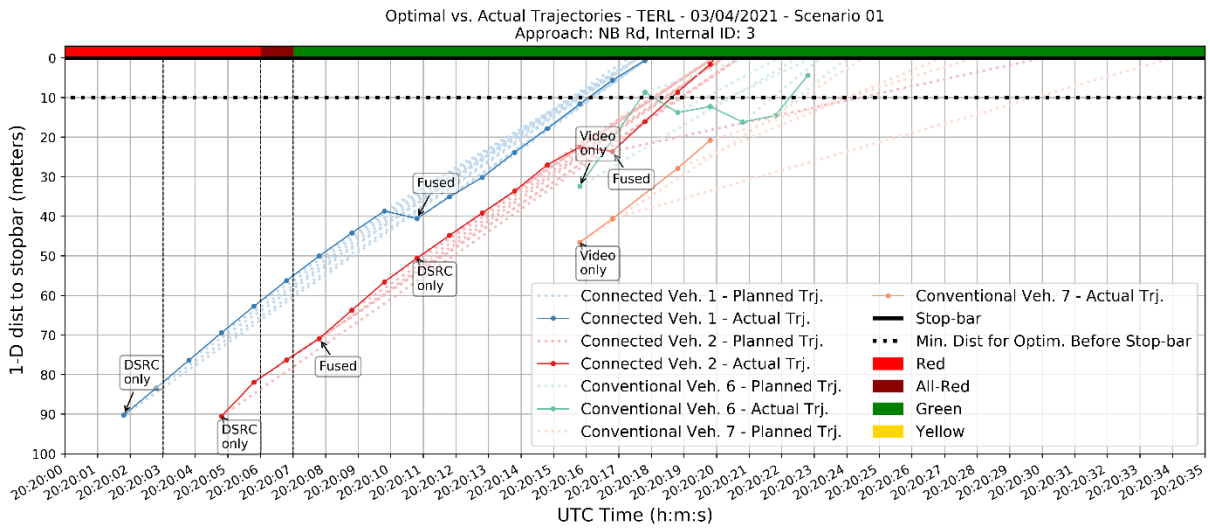


Figure 5-6: Time-space diagram for Scenario 1.

For this scenario, the vehicles are already at a similar speed as what is recommended by RIO. Sometimes, near the end of a vehicle's approach, RIO suggests that the driver slows down, as seen in the first replication of this scenario. This seems to be an issue with RIO's trajectory logic and is currently under investigation.

5.4.1.2 Scenario 2 – Platoon leader is conventional vehicle

In Scenario 2, a conventional vehicle is followed by two connected vehicles on the NB approach. Figure 5-7 provides a top-down schematic view of this scenario's layout, while Figure 5-8 shows the time-space diagram for one of the replications of this scenario.

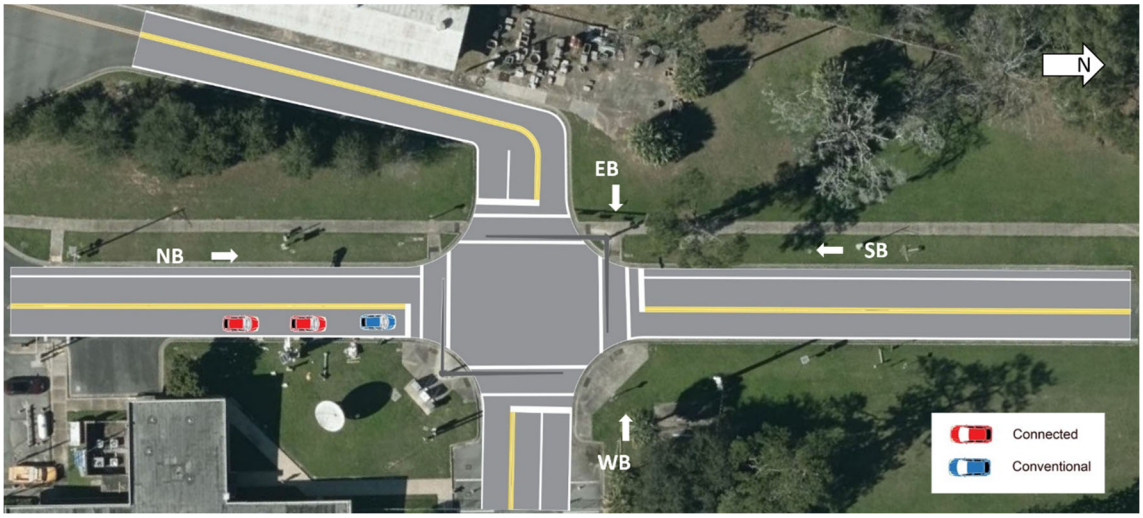


Figure 5-7: Top-down schematic for Scenario 2.

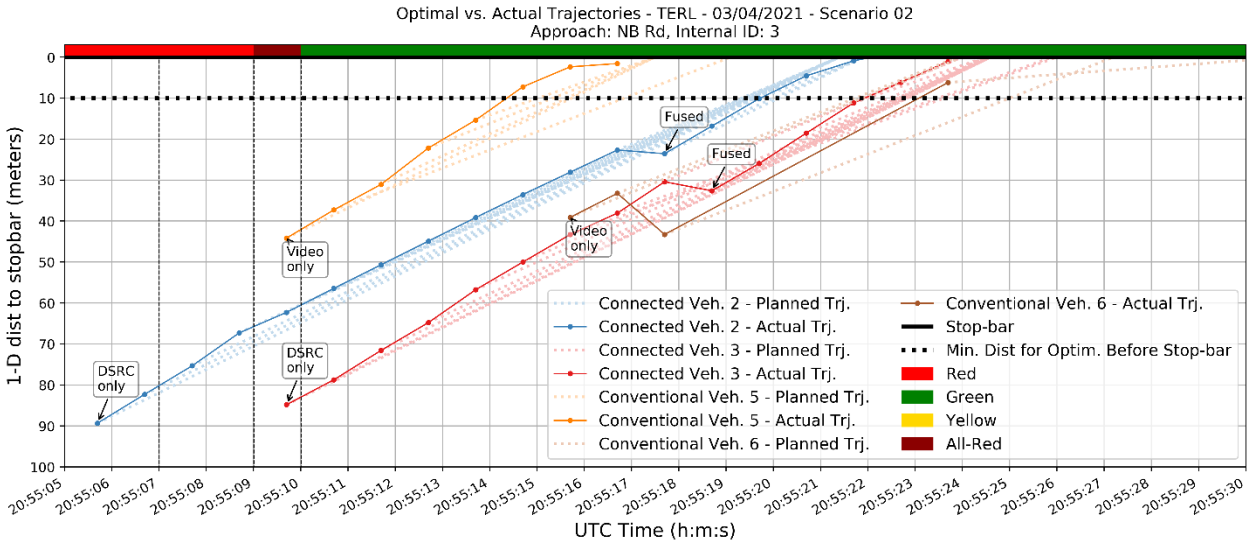


Figure 5-8: Time-space diagram for Scenario 2, second replication.

The vehicles' identifier (ID) are given in the order of detection time. In this scenario, the conventional vehicle 5 is the platoon leader. However, Connected Vehicle 2 is detected first, because the DSRC range is longer than the video range. This can be observed in the time-space diagram by looking at the timestamp of the first detection of each vehicle.

We can also observe how RIO reacted to Connected Vehicle 2 before Conventional Vehicle 5, by looking at the signal plan. The system is already operating during an all-red interval by the time that Conventional Vehicle 5 is first detected, which means that at this point RIO had already given yellow light to the previous phase. This behavior highlights the impact of the detection distance for each type of input: in the current system specifications, the connected vehicles can be detected much further away than conventional vehicles, giving more time for RIO to adjust SigPT to serve the connected vehicle demand. Once the conventional vehicles are closer, they do get incorporated into the optimization, and thus adequate time is provided for all vehicles to clear the approach.

5.4.2 Group 2 – Two-approach scenarios

In this group of scenarios two approaches are used: a conventional vehicle uses the NB approach while a connected vehicle utilizes one of the other approaches. The southbound and the eastbound approaches have different lengths (approximately 40 and 70 meters, respectively). As observed in Scenario 2, the detection distance may affect how RIO assigns SigPT. The objective of Group 2 is to further understand the implications of the detection distance on the system's performance.

5.4.2.1 Scenario 3 – Connected vehicle arriving from short approach

Scenario 3 consists of a conventional vehicle arriving from the NB while a connected vehicle arrives from the southbound approach (Figure 5-9). The southbound approach is relatively short, and its length is similar to the detection range of the video camera at the NB approach (roughly 40 meters).

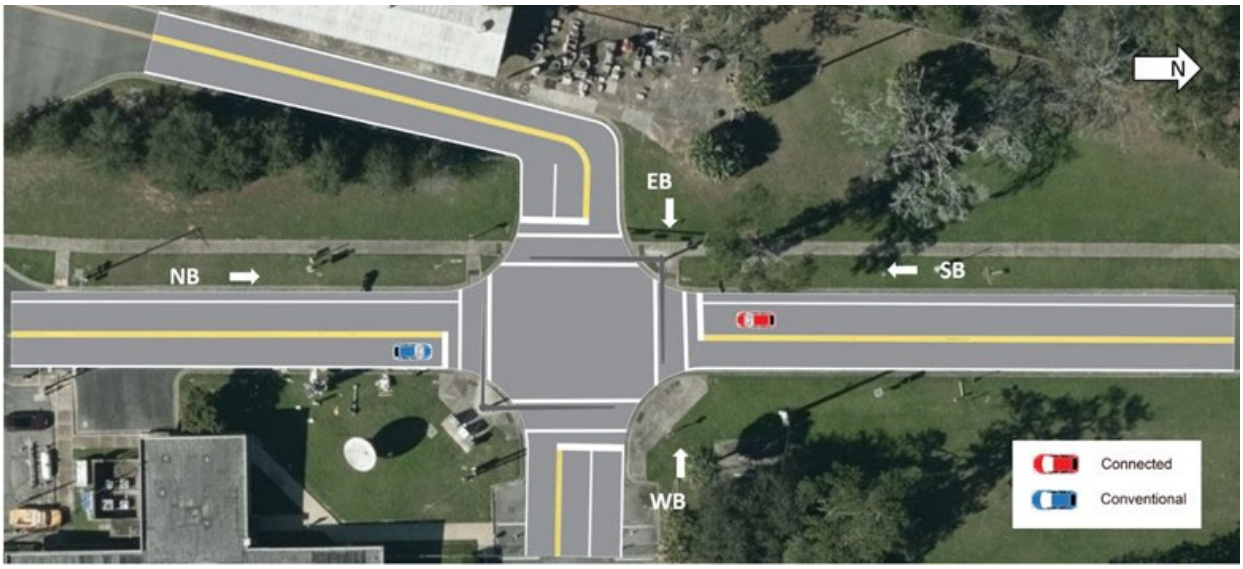


Figure 5-9: Top-down schematic for Scenario 3.

In this scenario the drivers depart at approximately the same time from the beginning of each of the approaches (refer to the introduction of Section 5.2 for the layout of the intersection, which includes the length of each approach). Since the southbound approach is shorter, and the detection range of the radio is greater than the video detection range, the Connected Vehicle 1 (SB) is detected approximately 5 seconds before Conventional Vehicle 2 (NB). Consequently, RIO assigns SigPT for the SB approach earlier.

In this scenario, the parameter Lag of Green start (LoG) causes a significant impact in the system's performance. LoG is an amount of time, given in seconds, that prevents planned trajectories from having a departure time too close to the beginning of the green interval. This allows drivers to visually confirm the green for a reasonable amount of time before traveling through the intersection. The LoG was set to 3 seconds after discussion with the test drivers (more details on this are provided in Section 5.5.3). Its effect can be clearly seen in the time-space diagrams illustrating the paths of both vehicles (Figure 5-10), which show the points where the planned trajectory intersects the stop bar. Vehicle 1 operates below the speed limit (it is traveling at 4.3 m/s, or 9.6 mph, while the maximum speed was set as 6.7 m/s or 15 mph). Although RIO could assign a planned trajectory with maximum speed, because of LoG it plans for an average speed of 5.7 m/s or 12.7 mph, ensuring that the green light would be visible 3 seconds before the driver would make it to the stop bar. Presumably, for fully autonomous vehicles the LoG can be eliminated.

Conventional Vehicle 2 is unaware when it will receive a green light, so it was forced to approach the stop bar and to stop outside of the video detection range. Similarly, the assigned trajectories respect the LoG and set a departure time 3 seconds after the initial green. If this were a connected vehicle, the driver would have been informed of RIO`s planned trajectory and would be able to slow down, reaching the signal at the beginning of the green light.

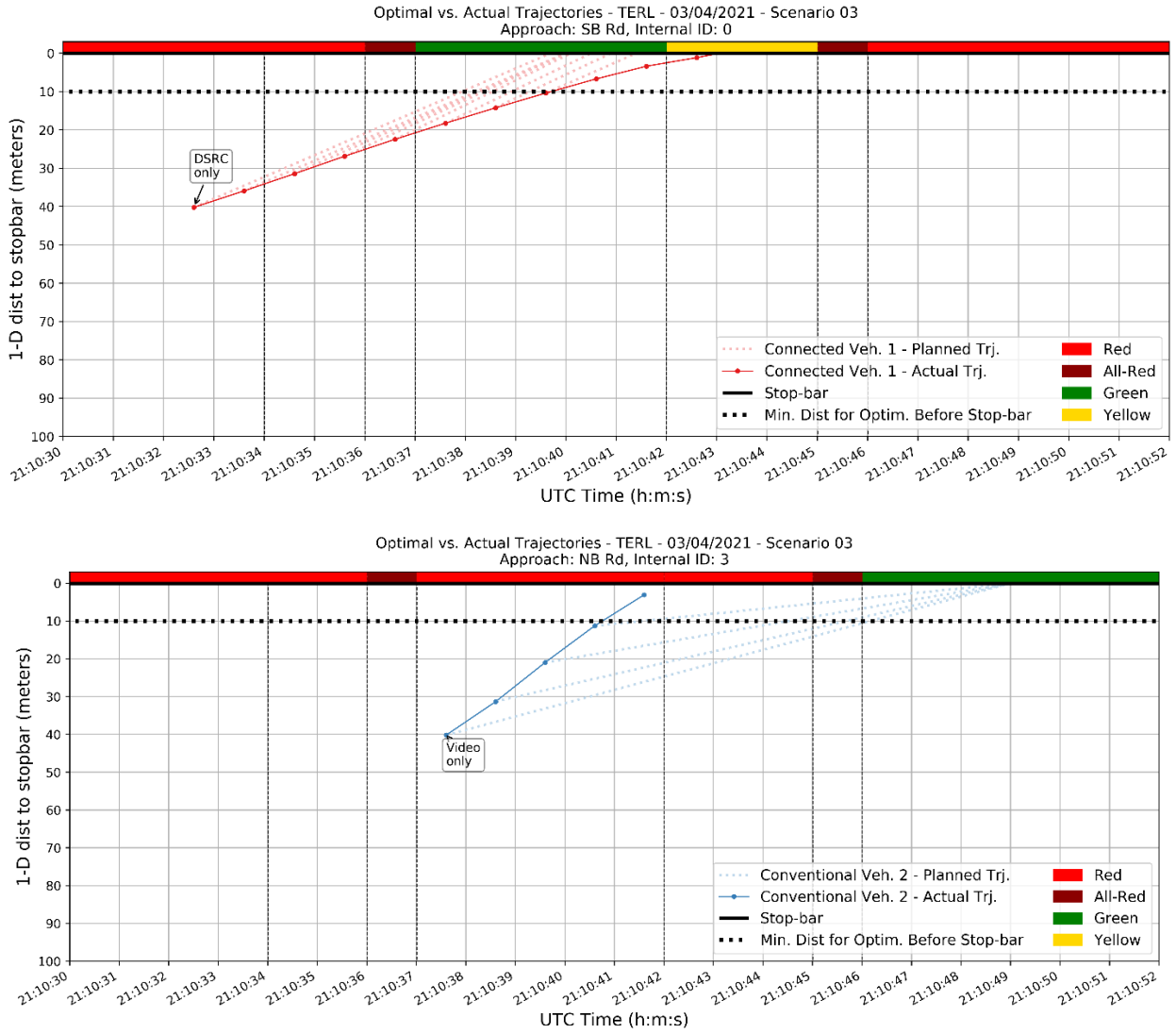


Figure 5-10: Time-space diagram for Scenario 3.

5.4.2.2 Scenario 4 –Connected vehicle arriving from long approach

In this scenario, the detection range of the radio provides a much greater advantage to the connected vehicle traveling EB (Figure 5-11), with the connected vehicle being detected roughly

9 seconds before the conventional vehicle (Figure 5-12), as opposed to 5 seconds in Scenario 3. Since RIO was able to start the EB phase much earlier, the LoG does not affect the start of green. However, the LoG is used again for the conventional vehicle, although this time the planned departure occurs only 2 seconds (rather than 3) after the start of the NB phase. This is an unintended behavior and is under investigation.

Figure 5-11: Top-down schematic for Scenario 4.

For scenarios 3 and 4 the detection range has proven to be the decisive factor in RIO's responsiveness and how SigPT priority was given to each vehicle. When Conventional Vehicle 4 approaches the stop bar, it is forced to stop due to the red light. If this vehicle had access to the phone app, it would be aware of its planned departure time, allowing it to travel slowly rather than stopping. Also, if it had a DSRC radio, it would have been detected earlier and RIO would likely give it preference over Connected Vehicle 1.

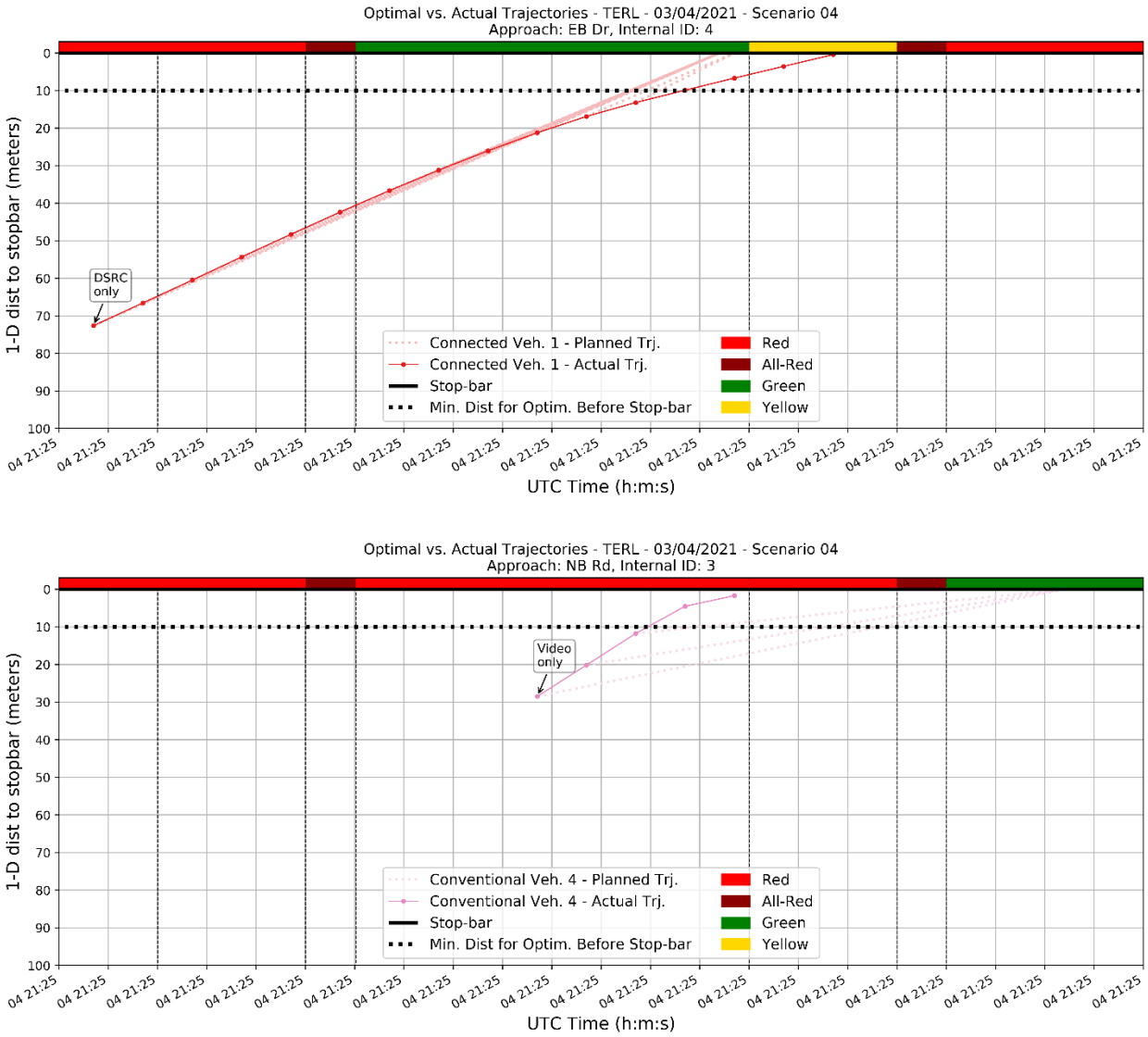


Figure 5-12: Time-space diagram for Scenario 4.

5.4.3 Group 3 – Two-approaches with staggered arrivals at the intersection

As observed in Group 2, the detection range and, consequently, the detection time is the main factor when determining the priority for assigning the right-of-way to each approach when traffic volume is very low. Thus, the connected vehicles possess a significant advantage over the conventional vehicles, especially for longer approaches when the connected vehicles can be detected earlier.

In Group 3, we further explore the implications of the detection time. While in Group 2 the drivers were told to depart at the same time from the upstream end of each approach, in Group 3 the drivers depart at different times. In Scenario 5, the conventional vehicle in the NB approach gets delayed by several seconds, while in Scenario 6 the connected vehicles in the EB

approach get a delayed start. A second objective of these scenarios is to evaluate the impact of having a slightly higher demand by adding another connected vehicle.

5.4.3.1 Scenario 5 – Delayed start for conventional vehicle

In Scenario 5 the conventional vehicle starts traveling from the upstream end of the NB approach seconds after the connected vehicle has already begun traveling along the EB approach (Figure 5-13). This delayed start varies between 3 to 7 seconds on each run for this scenario. As observed in Scenario 4, the EB vehicles will receive a green light early due to the longer detection range of the radios. The EB phase will be running for several seconds by the time the conventional vehicle is detected arriving from the NB approach. Thus, its arrival time is longer than the minimum green (GMin) for the conflicting approach. This will force RIO to decide whether to extend the green time of the EB approach to serve both connected vehicles, or to gap out in order to serve the NB approach. The decision depends on the positioning and speed of each vehicle currently in the network; RIO accounts for these two variables in order to find the feasible solutions of SigPT that can serve both vehicles, and then it chooses the solution that results in the highest throughput. It should be noted that the solver's current objective is to maximize throughput. It is possible to modify the objective such that delay or another parameter are minimized; however, such a modification is not trivial.

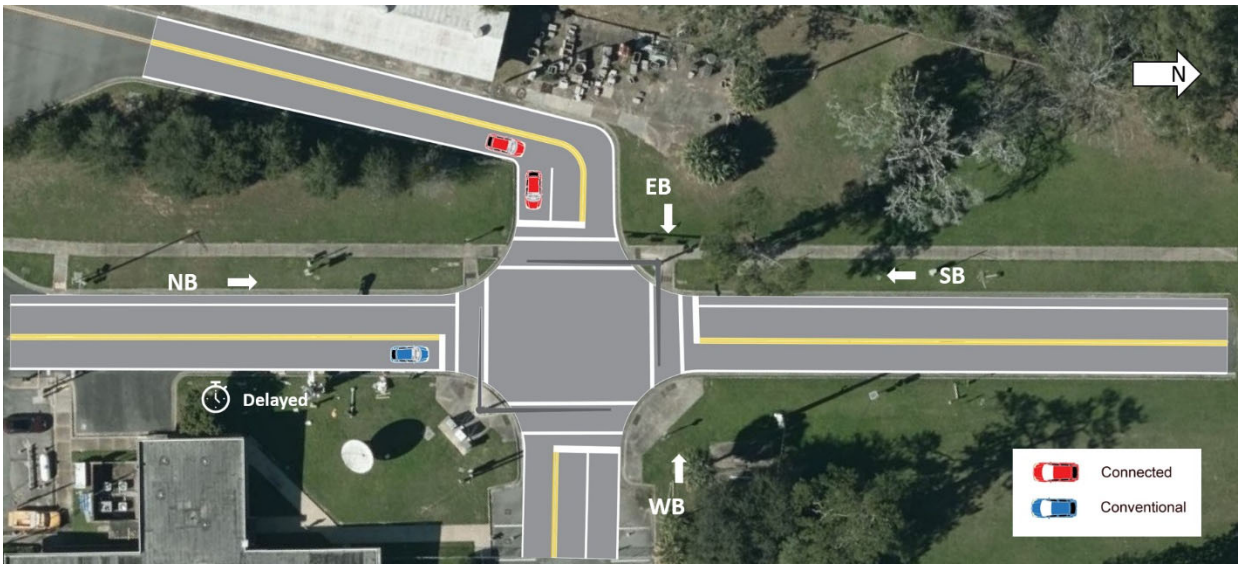


Figure 5-13: Top-down schematic for Scenario 5.

First, it is important to comment on the format of the trajectory of Connected Vehicle 2 (Figure 5-14). Two observations were recorded for each position of the vehicle, creating a staircase-like shape for the trajectory. This pattern repeats itself in Scenario 6, and it seemed to occur to connected vehicles that were in a car-following position, while the trajectory of the lead connected vehicle remains normal. This could be either a failure at the DSRC decoder or a failure of Sensible and is being investigated further. However, this did not seem to impact RIO's performance. Also, although both Scenario 5 and 6 show two conventional vehicles traveling in

the NB approach, there was only one vehicle present, so the second detection is either a false positive (Scenario 5) or a case of interrupted detection (Scenario 6).

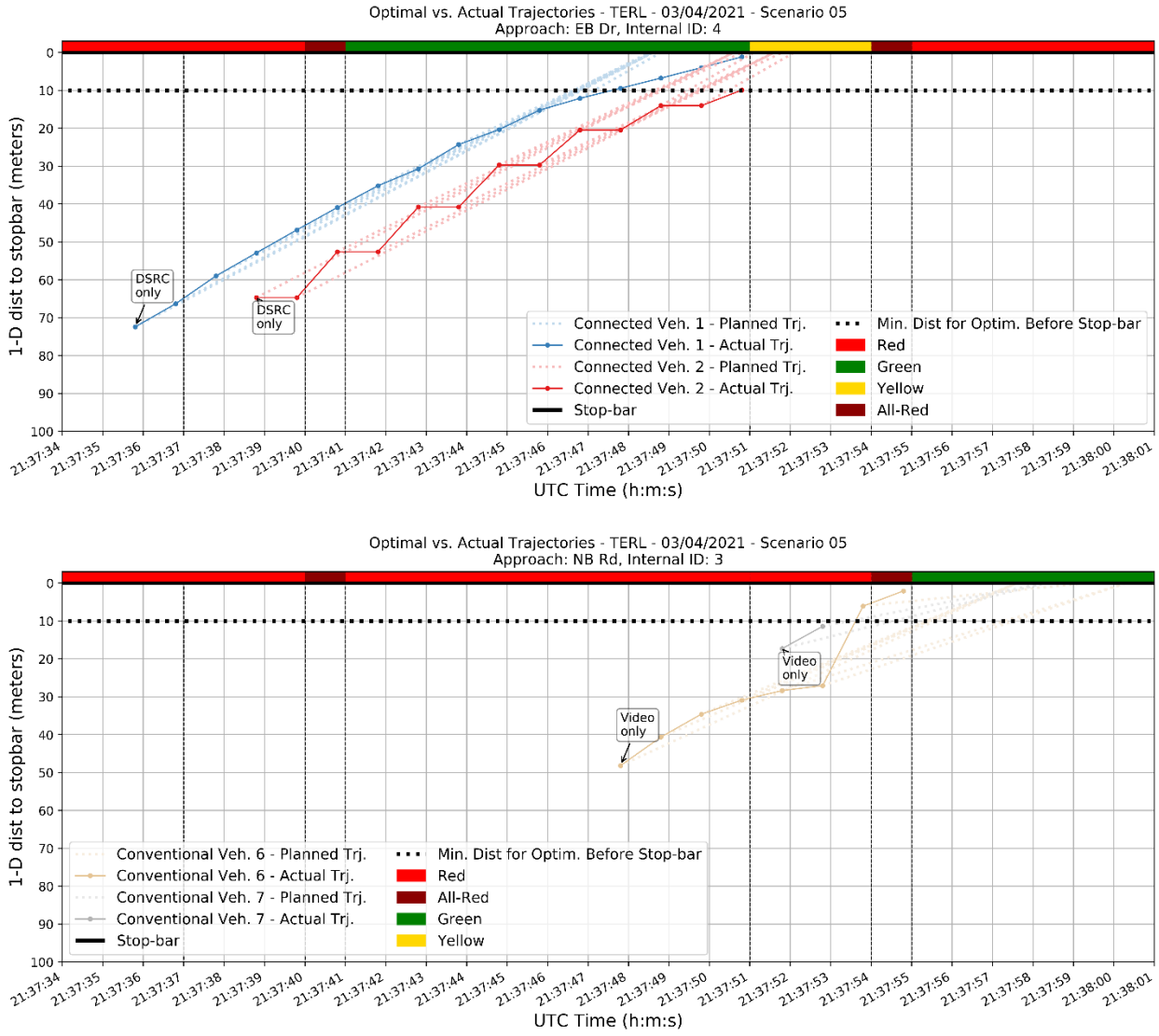


Figure 5-14: Time-space diagram for Scenario 5.

In this replication of the scenario, RIO extended the green light of the EB phase to serve both connected vehicles. This is illustrated by observing the conventional vehicle traveling in the NB direction (Figure 5-14). The conventional vehicle was detected at 21:37:48. If RIO decided to serve the NB approach immediately, the EB phase would immediately turn yellow. However, the yellow light took another 3 seconds to begin, indicating that this is the amount of extended green assigned to the EB phase. The first connected vehicle travelled slower than expected, and thus the second connected vehicle travelled through the intersection during the yellow. RIO accepts planned departures during the yellow, consistent with current practice for conventional vehicles.

Although RIO avoids serving vehicles during the yellow stage, it does not refrain from doing so in order to optimize the intersection throughput. This causes the planned trajectories of Connected Vehicle 2 to have departures during both the green and yellow intervals, depending on the vehicle's current position. Thus, RIO managed to serve all vehicles with little delay for each driver. The conventional vehicle's planned trajectory is not very different from the vehicle's actual speed for most of its trajectory points, hence this vehicle did not have to slow down in order to depart the stop bar during the green.

5.4.3.2 Scenario 6 – Delayed start for connected vehicles

In this scenario, the connected vehicles get a delayed start, allowing the NB conventional vehicle to be detected first (Figure 5-15). Similar to Scenario 5, the objective here is to observe how much time RIO will assign to the phase that is served first.

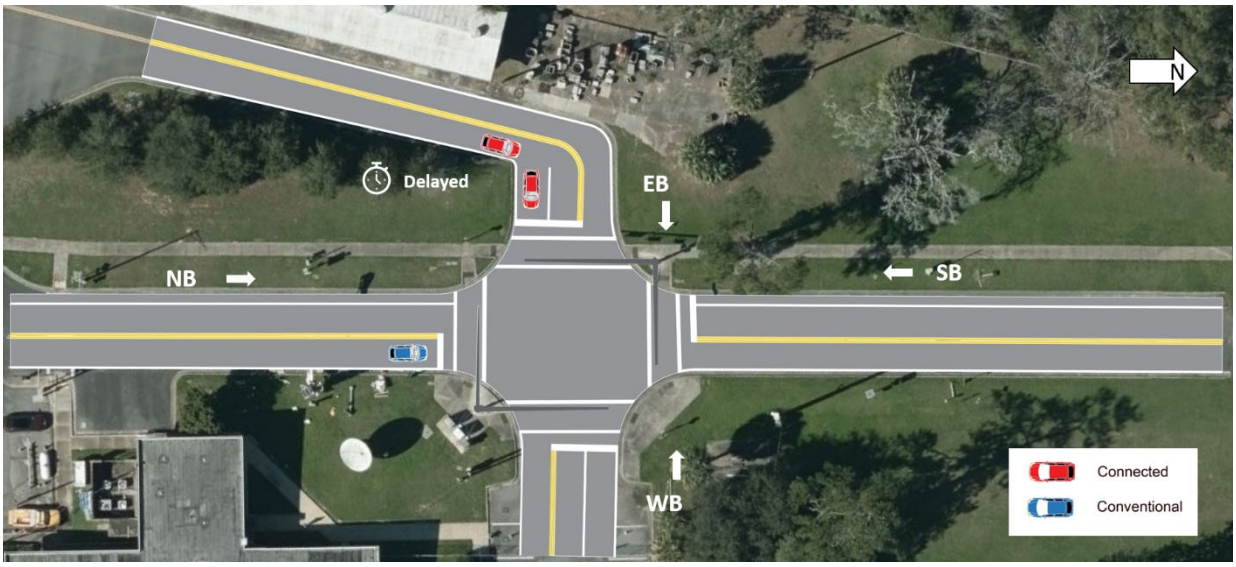


Figure 5-15: Top-down schematic for Scenario 6.

In this scenario, the first detection in each approach occurs with only 2 seconds of difference (Figure 5-16), with the NB phase beginning first. Since there is only one vehicle arriving from the NB, RIO assigned it a green length equal to the GMin of 5 seconds, quickly transitioning to the EB phase. The LoG can be seen for both approaches. As shown, it caused RIO to suggest a lower average speed to the leader conventional vehicle than its current speed.

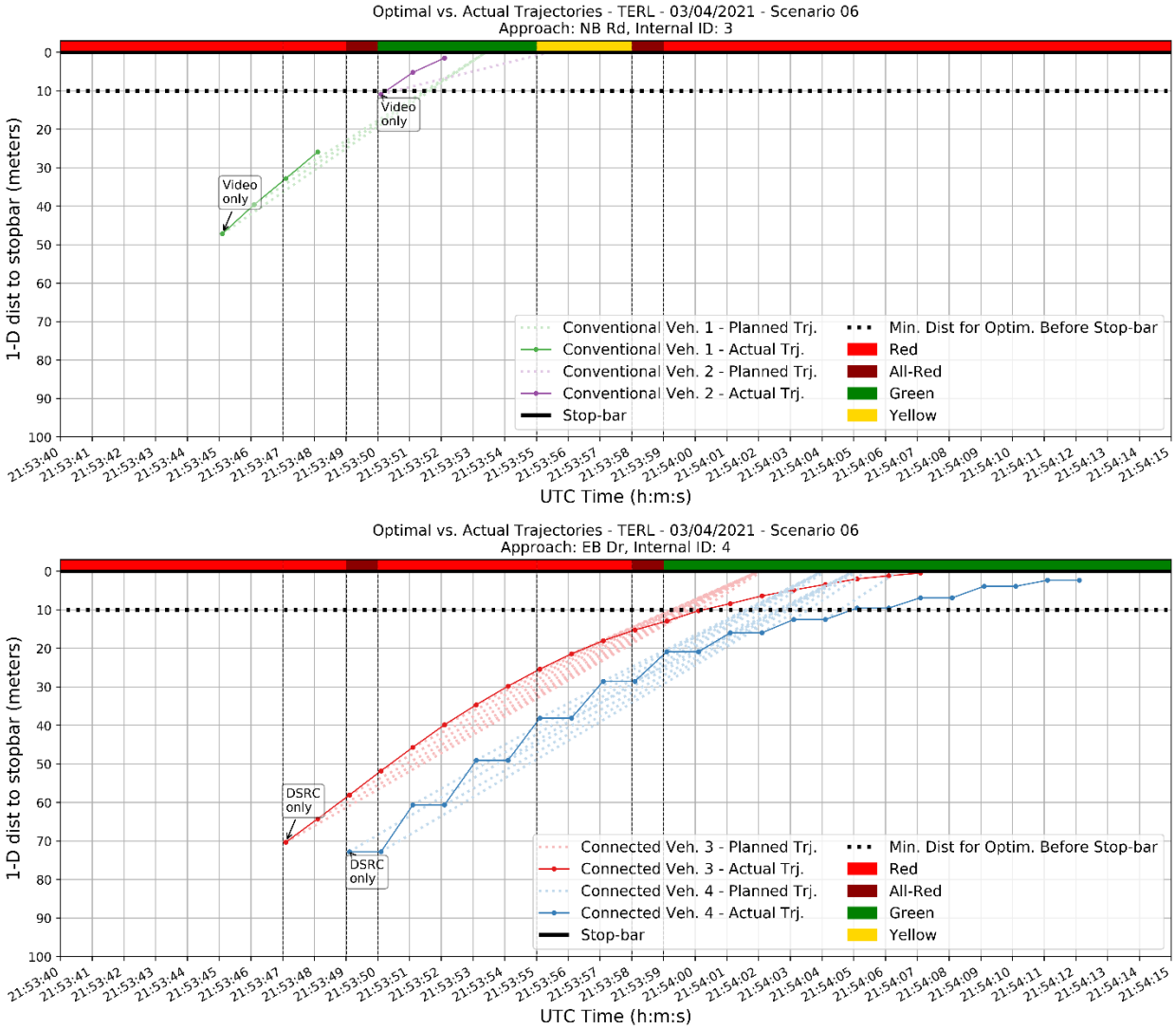


Figure 5-16: Time-space diagram for Scenario 6.

With the conclusion of Group 3 scenarios, we have tested and analyzed the behavior of the system's components and its relationship with the behavior of both conventional and connected vehicles. The next group evaluates the relationship between pedestrians and RIO's framework.

5.4.4 Group 4 – Scenarios involving pedestrians

This group consists of scenarios with pedestrian demand and serves specifically to prove the concepts described in Chapter 4. In these scenarios, a single connected vehicle utilizes the NB approach, while different pedestrian movements are called either through the LiDAR detection or through the Pedestrian Push-Button GUI resembling an actual activation of the real-world push button. The objective of this group of scenarios is to ensure that the implementation of pedestrian logic in RIO works correctly, serving pedestrian demand without conflicting

movements and assigning SigPT that correctly accounts for the amount of time required for pedestrians to cross safely.

5.4.4.1 Scenario 7 – An approaching vehicle conflicting with an approaching pedestrian

Scenario 7 consists of a single pedestrian that aims to cross the NB approach (Figure 5-17). A connected vehicle is traveling NB, creating a conflict with the pedestrian. In this scenario, the pedestrian detection occurs using the LiDAR. The time-space diagram for this scenario is shown in Figure 5-18.

Figure 5-17: Top-down schematic for Scenario 7.

A new layer of information is added to the time-space diagrams for Scenarios 7 and 8. Red, dashed-dotted vertical lines indicate that a pedestrian has approached the crosswalk and is waiting for a walk signal. Each red vertical line represents a pedestrian arrival. Green, dashed vertical lines indicate that a pedestrian has departed. It should be emphasized that pedestrians are not tracked individually in our system. Rather, each pedestrian entry is used to obtain pedestrian counts processed using either the LiDAR or the pedestrian push-button GUI. RIO assumes that a pedestrian registered in the system will depart as soon as they are provided the walk signal. When this occurs, the pedestrian entry status changes to “departed”, and the green vertical line is plotted in the time-space diagram. It is assumed that all pedestrians present at the sidewalk are able to cross during the green walk signal.

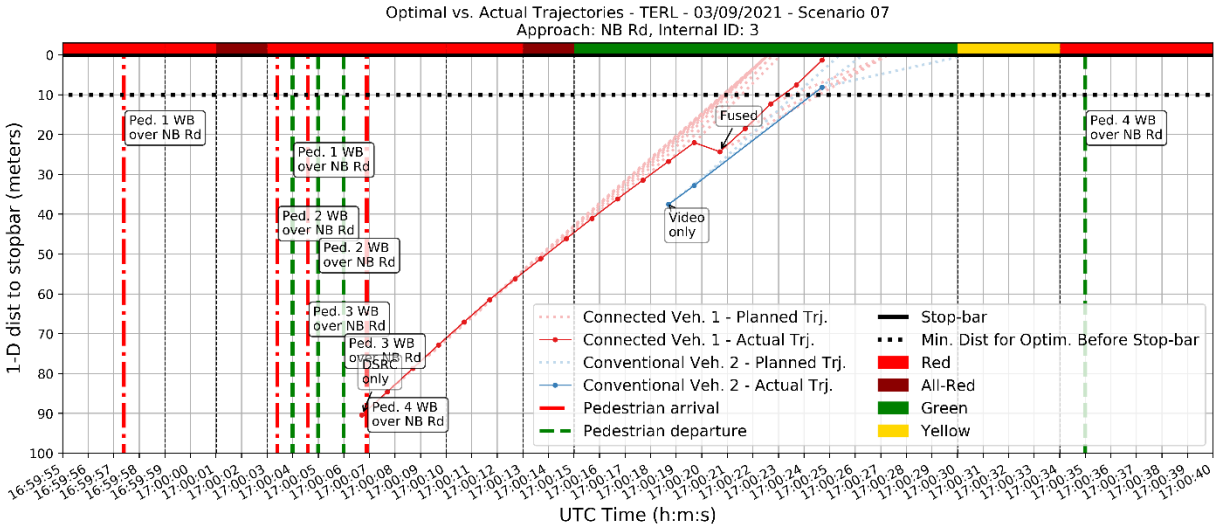


Figure 5-18: Time-space diagram for Scenario 7.

In Scenario 7, pedestrians 1, 2, and 3 are served before the connected vehicle. Also, their arrival and departure times are very close to one another. This is because, as previously mentioned, RIO considers a pedestrian to be served as soon as the walk signal is provided. Also, as long as RIO receives information that a pedestrian is standing at the waiting area, it will continue to ensure that a pedestrian entry exists in RIO. This creates a loop where each pedestrian entry remains for a short amount of time in RIO memory. This loop ends when the walk time is over. At this point, RIO keeps a single pedestrian registered (which is pedestrian 4 in this scenario). This is because, for the sake of testing, the single pedestrian standing at the waiting area stayed within the detection zone. In other words, during testing, we only have a single pedestrian that arrived at 16:59:58 who stayed inside the waiting area despite the walk light. The effect of this was that the pedestrian provided a continuous call for the walk signal. This pedestrian departed at 17:00:35. At this time, sensing technology cannot track the movement of each pedestrian with high enough accuracy. However, RIO is able to handle each pedestrian separately during the optimization, and this feature will be used once sensing accuracy is adequately high.

In this scenario, the vehicle receives a green light that is longer than GMin. This is because the vehicle cannot depart the intersection within the GMin period without traveling above the speed limit. The vehicular phase can last longer and up to the maximum green (GMax). Eventually, the vehicle phase is completed, and the pedestrian is given the right-of-way.

5.4.4.2 Scenario 8 – Two pedestrians and a vehicle in conflicting movements

This is an expansion of Scenario 7 with the addition of another pedestrian crossing at the parallel crosswalk (across the SB vehicular approach), as represented in Figure 5-19. During testing, this second pedestrian was added to the network through the push-button GUI from RIO, which is a temporary substitute for the physical push button typically present at an intersection.

As indicated earlier, each crosswalk is served in a separate phase, hence each of the crosswalks in this scenario must be served separately. Thus, at least three phase calls are required to serve the demand of two pedestrians and the connected vehicle. This scenario was designed to evaluate whether RIO would assign SigPT accordingly when more than two phases need to be planned in advance.

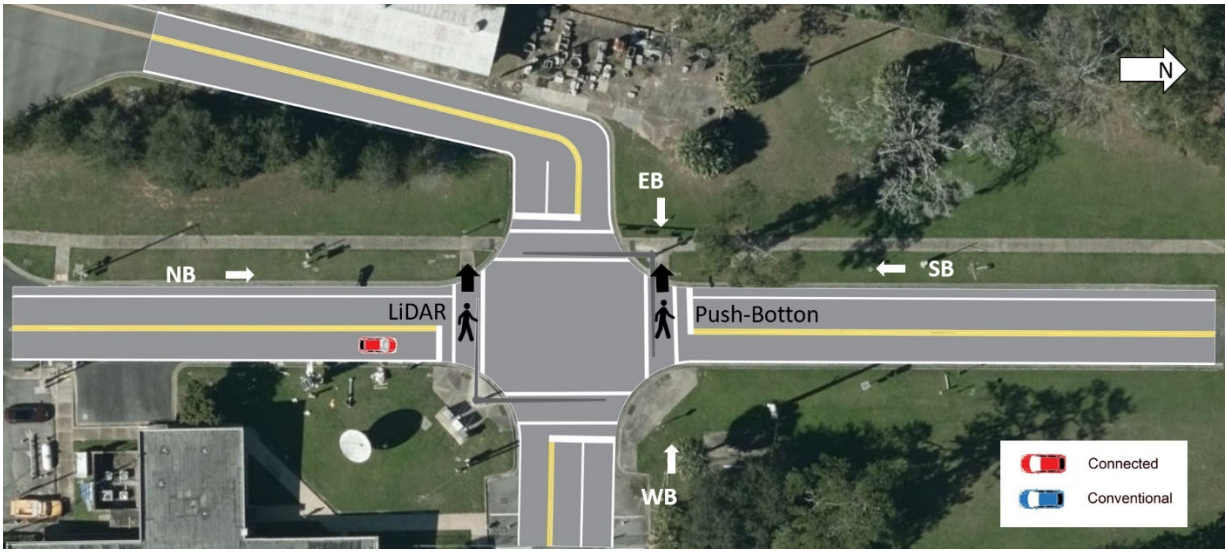


Figure 5-19: Top-down schematic for Scenario 8.

In this scenario, the crosswalk across the NB approach is served first, followed by the NB vehicular approach, and then with the crosswalk across the SB approach (Figure 5-20). The phase order occurred based on the arrival time of each pedestrian or vehicle (first-come, first-served). The first pedestrian phase used the minimum required time (7 seconds of minimum pedestrian walk signal time, 3 seconds of yellow, 1 second of all-red, for a total of 11 seconds) while the NB vehicular phase secured priority for the connected vehicle using 9 seconds of green due to the vehicle's position and approach speed limit.

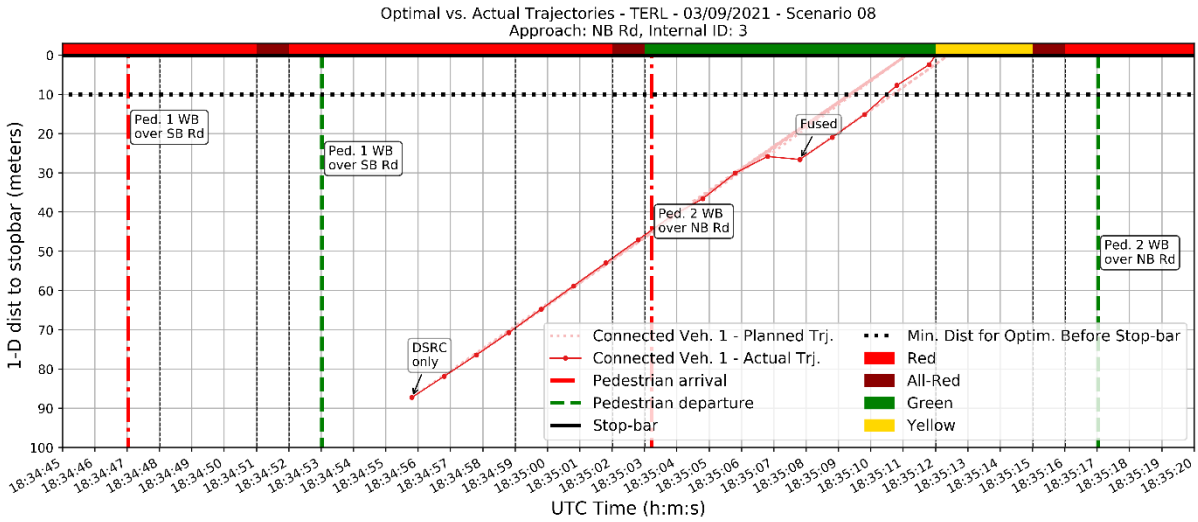


Figure 5-20: Time-space diagram for Scenario 8.

5.4.5 Summary of quantitative results related to traffic operational quality

Based on the test scenarios described above, we conclude the following:

- Sensible correctly identified all vehicles and pedestrians during the testing, and RIO managed to secure the minimum green for both vehicles and pedestrians in all scenarios.
- The signal timings that RIO produces are highly sensitive to the detection range and detection time. This occurs because RIO is programmed to provide the right-of-way in a first-come, first-served order.
- Connected vehicles can be detected earlier, and thus are given right-of-way priority, even when a conventional vehicle may be closer to the intersection.
- Results show that the phone app could enhance the driver's experience by preventing stopping and excessive breaking.
- RIO provided reasonable vehicular and pedestrian SigPT during all scenarios.
- Some RIO parameters can be tweaked by the traffic manager to accommodate local customs and as a function of intersection geometry, specific laws, etc.
- A few bugs and limitations that were identified in system components did not compromise the RIO stability or performance.

5.5 Qualitative observations

This section provides a series of qualitative observations related to specific components of the overall system.

5.5.1 Sensible

As discussed earlier, Sensible fuses data from multiple sensors and provides a single set of inputs regarding each detected vehicle's arrival. At the TERL only the NB approach has a video

camera. The research team observed that the tracking is robust to rainy weather (Figure 5-21) as well as sunny weather (Figure 5-22).

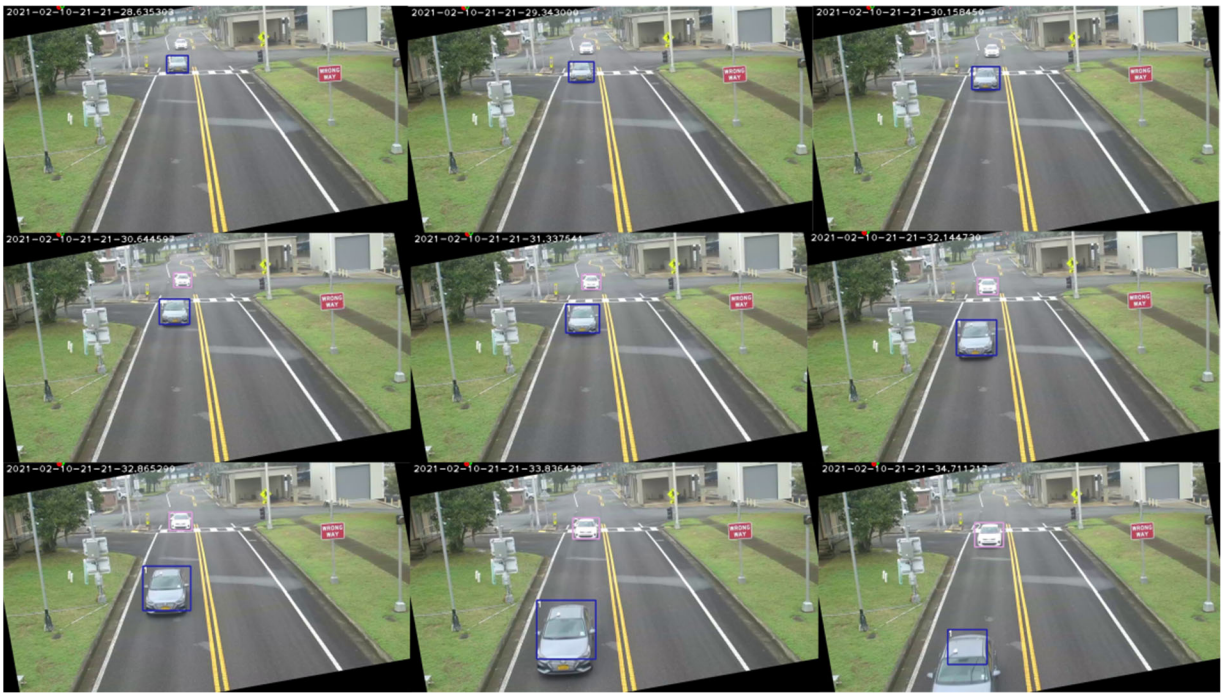


Figure 5-21: Rainy conditions, two vehicles (colored boxes indicate currently tracked objects).

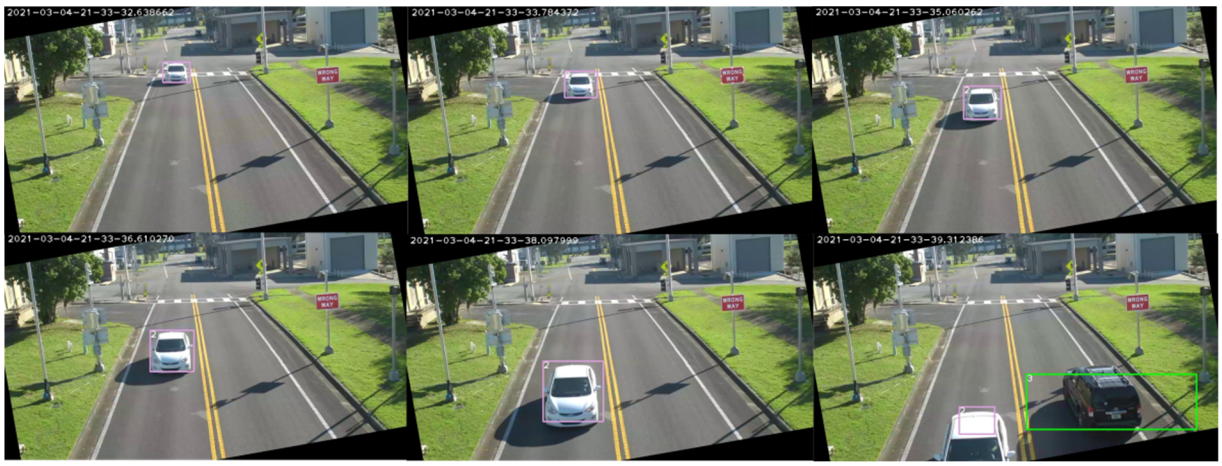


Figure 5-22: Sunny conditions, two vehicles.

5.5.2 LiDAR

The research team established two zones at the TERL (Zone 1 and Zone 2, shown in Figure 5-23) by drawing two polygons near the LiDAR (in sensor-centric coordinates). Any positive detection in a zone triggers the ped call. This effectively allows RIO to ignore any false positive detections from pedestrian-shaped objects outside of these zones. Overall, we found that

establishing these zones helped the LiDAR pedestrian detection work very reliably. Chapter 2 provided a detailed quantitative analysis of the LiDAR performance based on labeled data.

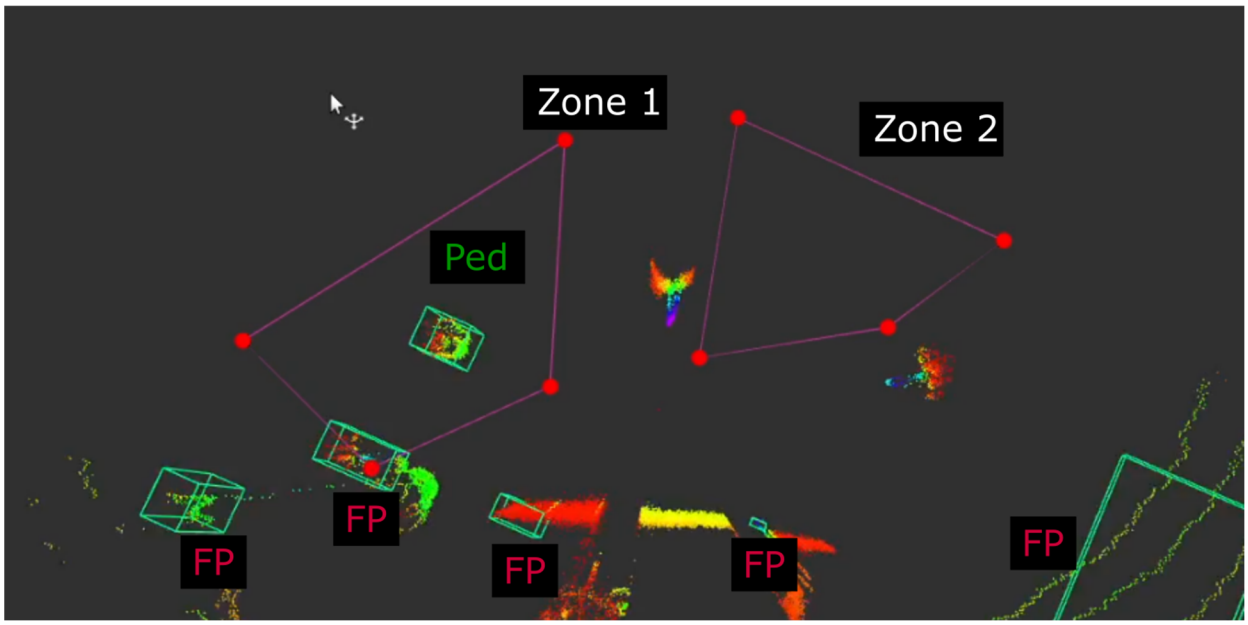


Figure 5-23: A snapshot of the LiDAR detections with two detection zones. Green boxes are detections. FP = False Positive.

In Figure 5-24 (a) a pedestrian waits to cross the street while two vehicles drive through the intersection. The LiDAR detects them and sends a call to the ped phase in RIO. Figure 5-24 (b) shows that the pedestrians received the walk signal soon after.

(a) (b)

Figure 5-24: LiDAR being used to call the ped phase.

5.5.3 Mobile App

Figure 5-25 shows a connected vehicle driver approaching the intersection from the NB. As shown in (a), the driver has the mobile app open on their phone, while (b) shows that once the vehicle is close enough to be identified and processed by RIO (~80 meters), they receive the first

trajectory. The recommended average speed is 12 mph. At that time the signal is still red for the NB approach. In (c) the signal changes to green before the vehicle arrives at the stop bar. At that time the vehicle is ~8.5 seconds away from crossing the stop bar. In (d) the driver reaches the intersection and proceeds through the green. The vehicle is given the right-of-way vimmediately because there is no conflicting demand from other approaches.

(a) (b)

(c) (d)

Figure 5-25: An intersection approach using the mobile app in a lead connected vehicle.

Figure 5-26 shows a driver in a connected vehicle approaching the intersection, except this time there is a conventional lead vehicle. In part (a) the driver receives an initial trajectory as soon as the vehicle is within the system's sensing range. The conventional lead vehicle has already been detected and is being served. In part (b) the two vehicles are shown to cross the stop bar during the green. Occasionally, RIO sent slower trajectories than necessary, and here the app shows additional time to cross and slower speeds (7.7 seconds, 5 mph) near the stop bar.

(a) (b)

Figure 5-26: An intersection approach using the mobile app in follower connected vehicle.

5.5.3.1 Drivers' feedback on the mobile app

Users of the mobile app indicated that they feel insecure when the app tells them to cross at a given time while the signal is still red and they do not know when it will turn green. One driver mentioned the following:

"*When nearing the intersection……low speeds and short distances made it hard to judge whether the speed recommendations made a meaningful impact."*

This seems to cause drivers to slow down. This could be addressed in future work by improving the app's user interface and adding the time when the signal is expected to turn green. Also, with increased use, drivers may trust the system more and feel more comfortable following trajectory recommendations.

Another driver also mentioned that "*Trying to match the speed on the phone, that was mounted to the windshield, and the speedometer is very distracting".* A third driver also commented on the issue: *"Looking at the phone screen to follow many changes in the speed and changing the car's speed based on the recommendations was a huge distraction."* These two comments make it clear that the content and the form of presenting data through the phone app needs to be developed together with the drivers' continuous feedback. One of the drivers also suggests a different form of presenting the speed suggestion: *"I do think it would have been less distracting to match a color change than reading numbers from the screen (this could be troublesome for times when a large change in speed is desired)."*

5.5.4 Object Tracking Message (OTM)

This section discusses the testing of the object tracking message (OTM) and its efficacy. The OTM is a proposed discrete short-range communication (DSRC) message designed to allow source-independent sharing of detected object locations between intelligent intersections and CAVs. In testing, the CAV was parked at the stop bar in the intersection's south bound lane. A conventional vehicle approached the intersection in the north bound lane. As the conventional vehicle passed through the field of detection of the intersection, object tracking messages were populated and transmitted to the CAV.

Figure 5-27 presents the pre-recorded point cloud map of the intersection that the CAV uses to localize itself and to detect obstructions. The blue mark locates the CAV and is centered on the Velodyne LiDAR located on the vehicle and used for detection. The bands surrounding the LiDAR are active detections, and the green enclosed figures are clusters of detections that have been identified as obstructions. The red circular marker in the bottom left of the figure represents the position of the conventional vehicle as measured by the intersection. The red circle in the middle of the figure highlights the position of the conventional vehicle as measured by the CAV. There is a clear disparity in the measured position as viewed by the two sources. This disparity may be explained by the latency in the measurement of the position by Sensible.

However, the position of the conventional vehicle as measured by the intersection is shifted outside of the roadway. This shift may indicate a misalignment of the CAV's coordinate frame to world coordinate system. The average time required to populate and transmit the message was found to be ~0.15 seconds. The object tracking message shows viability due to its low latency. The knowledge base of the CAV was increased as other vehicles became known to the CAV at distances that far exceeded the CAV's onboard detection range. The testing did show the importance of accurately referencing the CAV's pre-mapped point cloud (which the CAV's position is determined from) and the coordinate reference of the IC system. An accuracy of less than 10 cm, which is obtainable, should be sufficient.

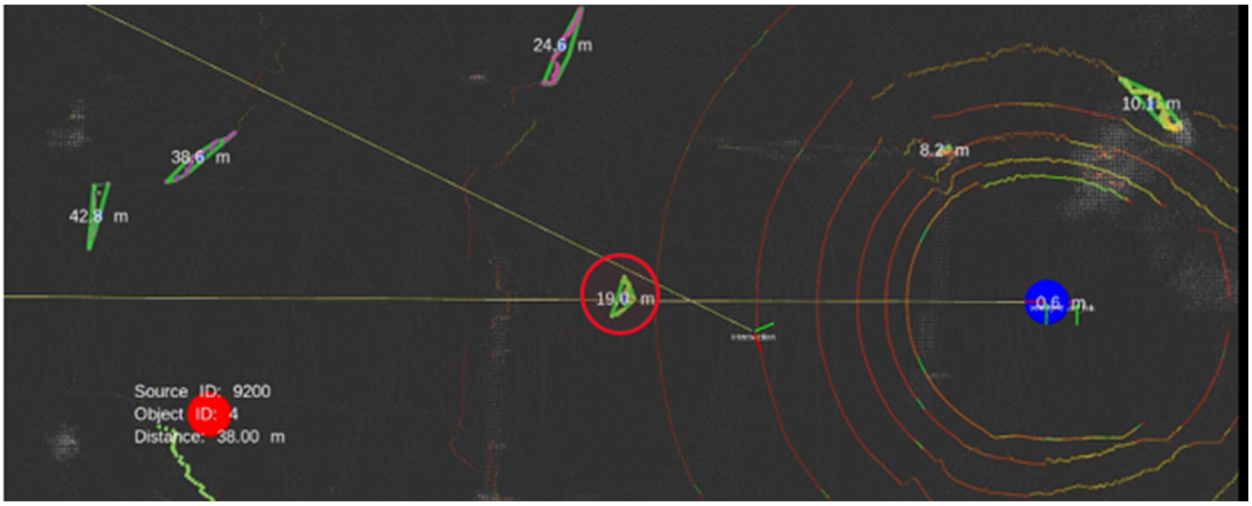


Figure 5-27: Simultaneous detection of the conventional vehicle by the intersection and CAV.

5.6 Additional sensing results

In this section, we analyze additional aspects of the sensing to evaluate the overall performance of the system.

The camera calibration is visualized in Figure 5-28. The research team created a map of each image with the distance of each pixel to the camera center in meters, visualized by a colored heatmap. The distance of various objects to the camera was confirmed using Google Maps to verify the distance measurements. For example, Google Maps shows that the crosswalk is ~47 meters from the camera, which agrees with the calibration visualization. Overall, the calibration was reliable within the range that vehicles were detected at (~80-90 meters). The calibration process was as follows. First, we manually identified 22 key points visible in each video frame as well as from a top-down satellite view taken from Google Maps. Then we computed a 2D-2D homography using OpenCV with the 22 matching pairs of coordinates. Next, a translation and rotation to the camera-centric world coordinate system from the global coordinate system of the satellite view is used to get tracks in meters from the camera's position. After connecting to the RTSP stream we noticed that the camera was tilted such that the road was rotated clockwise by about ~10 degrees. This was throwing off our video tracker, so we applied a ~10-degree rotation counter-clockwise using image processing techniques before detecting vehicles (Figure 5-29).

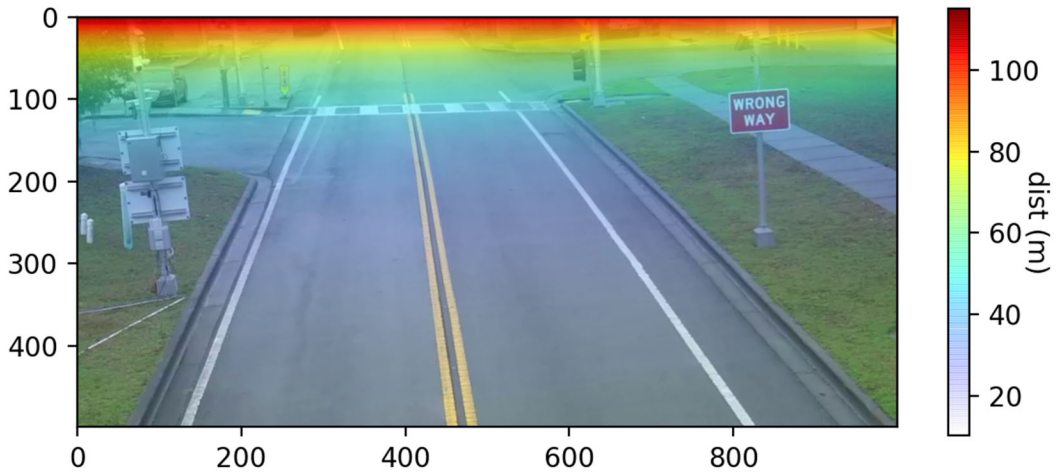


Figure 5-28: Camera calibration accuracy: The heatmap shows the estimated distance of each pixel to the camera, in meters.

(a) (b)

Figure 5-29 (a) Before rotation (b) after rotation.

We show an overall summary of the tracker statistics across all nine tested scenarios in Table 5-2. Each scenario had at most two vehicles with OBUs, so the number of DSRC tracks per scenario never exceeds two. Most scenarios were repeated multiple times, and a new video track was created for each vehicle as it approached on the northbound through lane or drove away from the intersection. We used no more than three vehicles at a time and did not do more than three repetitions for any single scenario. Although Sensible tracks vehicles driving away from the intersection, we filter them out later since they do not need to be included in the optimization. Notice that in Scenario 3, there is a low number of fused tracks (just one). This is because this scenario had the connected vehicle approaching from the SB, which is out of sight of the camera. Therefore, video tracks are created for the conventional vehicle driving NB, but not for connected vehicles that are not seen by the camera. We analyze one run from Scenario 1 later in the report to investigate the ability of Sensible to fuse video and DSRC tracks. We note that in most academic studies, ground truth track labels are available that can be used to compute various accuracy metrics. In our case, we did not possess the extensive resources required to record and label each data point for this field experiment.

Table 5-2: Tracking summary.

Scenario ID	1	2	3	4	5	6	7	8	9
Number of DSRC tracks	2	2	1	1	2	2	1	1	1
Number of video tracks	11	12	13	16	16	17	3	4	2
Number of fused tracks	16	22	1	4	6	13	3	5	2

The set of DSRC tracks generated by each OBU are visualized as a spatial-temporal heatmap in Figure 5-30. Cool colors represent the (temporal) start of a vehicle track and warmer colors the (temporal) end of the track; since tracks are overlaid for a holistic view, individual tracks are not able to be inspected. The OBUs are able to provide GPS-based tracks consistently and within 2-3 meters of accuracy. We also aggregated and visualized all video tracks from the NB approach (Lane 3) as a spatial-temporal heatmap in the left panel of Figure 5-31. The cooler (bluer) portions of the track show where the video tracker initially can detect vehicles along the northbound through lane. Video tracks terminated near the stop bar when they left the view of the camera. We computed a histogram of the distance to the camera when the vehicles are first detected, shown in the right panel of the figure. We find that most video tracks start around ~82 meters from the stop bar. The (A) cluster of short distance tracks represent both vehicles driving away from the intersection as well as spurious tracks. This (B) group represents the typical scenario, with vehicles first being tracked about ~82 meters from the stop bar.

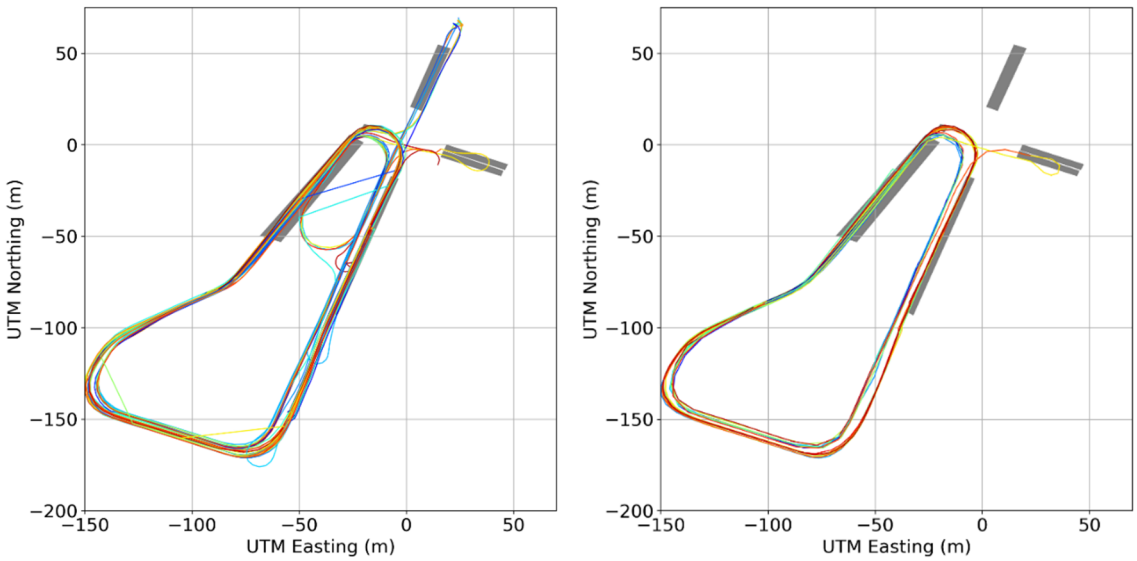


Figure 5-30: All DSRC-based tracks from the 9 scenarios (on the left is the OBU with ID 2000001 and on the right is the OBU with ID 1100000).

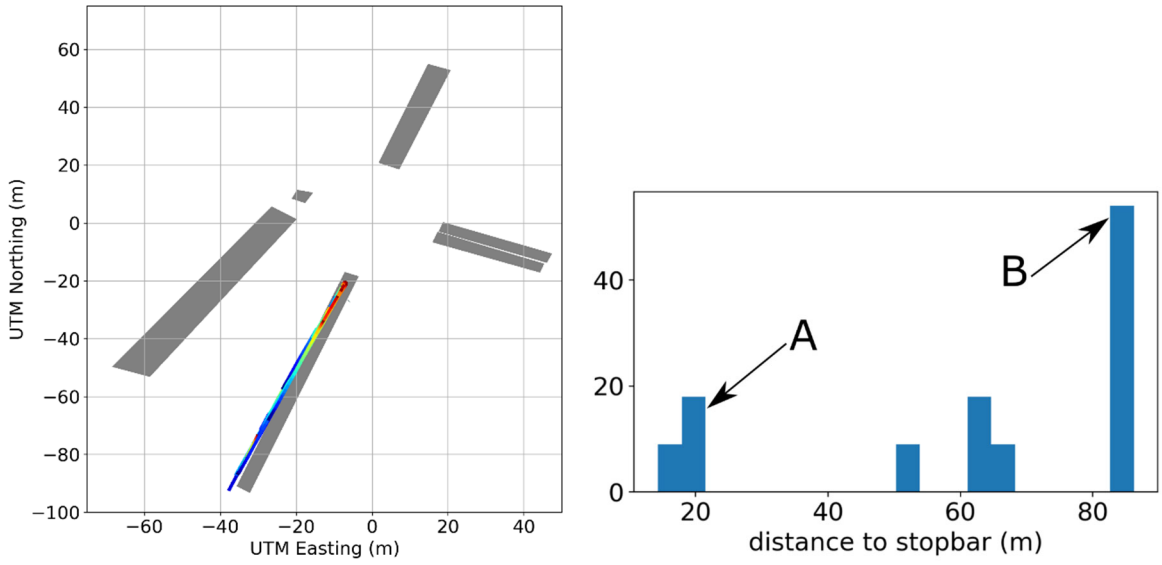


Figure 5-31: (a) Heatmaps for all conventional and connected vehicle video tracks approaching the intersection in Lane 3 across all scenarios; (b) Histogram of initial distance to stopbar for vehicles tracked by video with two clusters (A, B) annotated.

Figure 5-32 depicts eight snapshots of different times during Scenario 1, which we use to highlight the ability to fuse video and DSRC tracks. In the legend for each sub-figure, the single digit numbers (0,1,2) represent video track IDs. The large numbers (2000001 and 1100000) are the OBU IDs. The hyphenated numbers are fused tracks; e.g., 0-2000001 means that video track 0 and DSRC track 2000001 have been associated together.

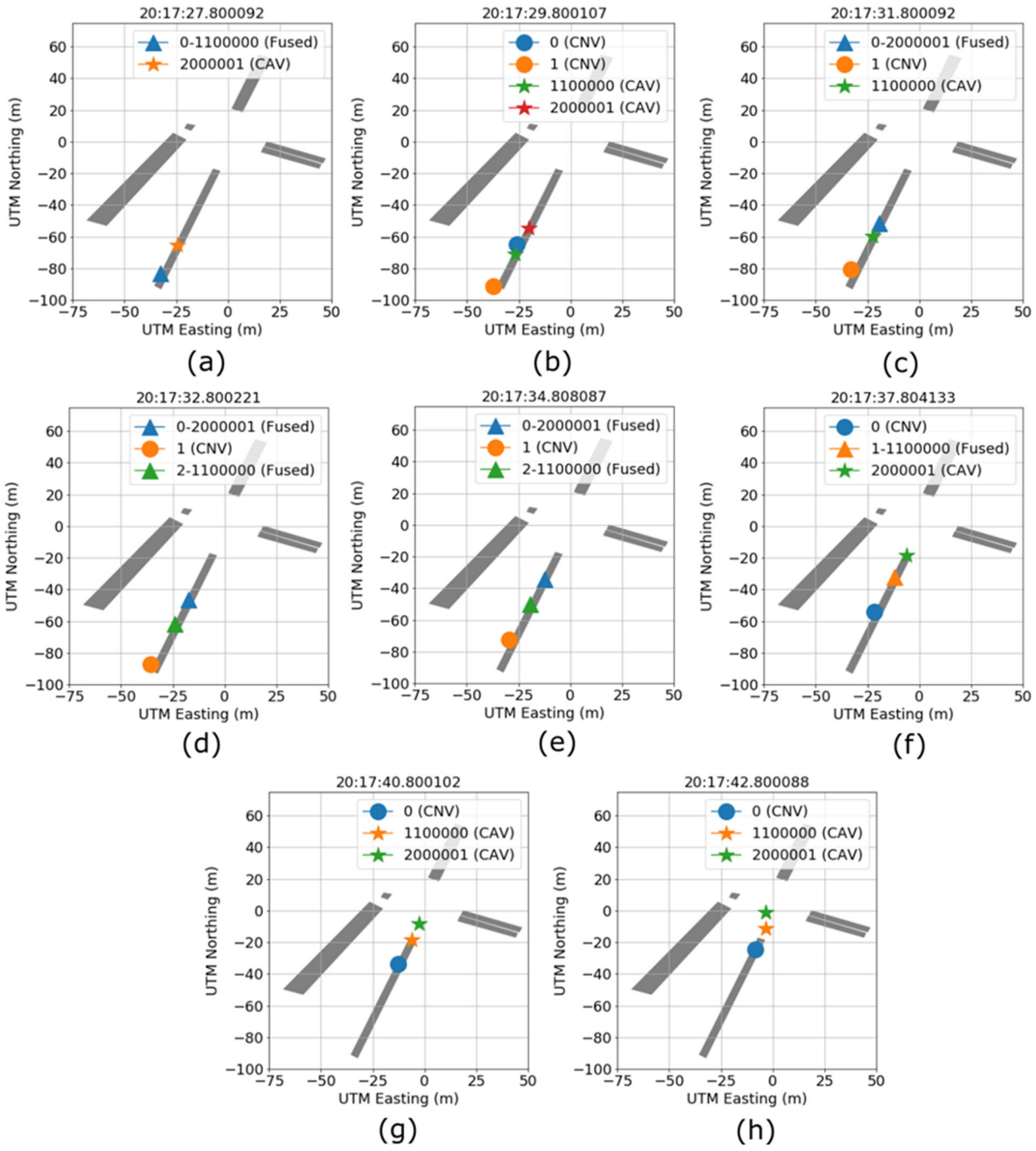


Figure 5-32: Sensible tracking and fusion for three approaching vehicles in Scenario 1 (stars indicate positions of the vehicles at the timestamp indicated by the plot title). Vehicle 0 is a conventional vehicle and vehicles 1100000 and 2000001 are connected vehicles.

In (a) there are two vehicles visible to the intersection but only one video track, which is associated with the follower connected vehicle with ID 1100000 (0-1100000). Based on the subsequent analysis, this is likely a false association, where we show that vehicle track with ID 0 will become associated with DSRC track 2000001. In (b) the tracker begins to correct itself, generating a second video track with ID 1 and un-associating video track 0 with DSRC track 2000001. Subsequently in (c) the tracker has associated video track ID 0 with 2000001. The gap between video track ID 1 and DSRC track 1100000 indicates that these are generated by different vehicles; video track ID 1 is the conventional vehicle following behind the two connected vehicles. In (d), the video tracker has created the track for the second connected vehicle (video track with ID 2) and fused it appropriately. The tracker maintains the assignments of vehicles to IDs in (e). Then in (f), fused track 0-2000001 has reached the stop bar and the video tracker has lost sight of ID 0. We observed an ID swap caused by the video tracker swapping ID 0 and 1 erroneously, now showing that ID 1 is fused with 1100000 and using ID 0 for the conventional vehicle. ID 2 has been dropped. Shortly after, in (g-h) the two connected vehicles have reached the stop bar and the only remaining video track is ID 0 for the conventional vehicle. To summarize, we find that the tracker has high recall (it tracks all vehicles), however, occasionally makes errors maintaining a consistent ID for vehicles. This is heightened in scenarios such as Scenario I where vehicles are in close proximity. While the IDs were not always held consistently, the number of vehicles tracked by the algorithm only increased beyond the true number once (4 vehicles, shown in Figure 5-32b). This is promising for maintaining accurate estimates of queue lengths and traffic volume. We note that the tracker will continue to be improved upon over time as newer and better algorithms emerge.

5.7 Analysis of hardware/software reliability and performance

This section discusses issues related to hardware and software reliability, including latency in communications.

5.7.1 System latency

Given the importance of timeliness in the transmission of information from one system component to another, we assess here for each component any delays encountered in communications and any reliability issues observed during testing.

5.7.1.1 Pedestrian LiDAR

There were roughly 2 seconds of delay for processing of the pedestrian recognition, with ~1.4 seconds needed for the detection algorithm to run for each pedestrian point cluster. The clustering step takes about ~0.6 second. There was nearly instant transmission of this information to the RIO computer.

RIO checks for incoming messages at 1 Hz, the same frequency used for the optimization loop. Thus, the maximum delay for RIO to read a new pedestrian count message is, theoretically,

1 second. Since the pedestrian recognition does not operate in a fixed frequency, RIO receives messages in a non-fixed frequency, hence the delay for utilizing the newest message may vary. In summary, the total delay for the system to react to pedestrian demand is ~3 seconds.

5.7.1.2 Signal controller

During the TERL testing we observed that the signal controller may take between 1 and 2 seconds to properly update signal timings. The reasons for this delay include:

- RIO runs on a IIC at a configurable frequency which was set to 1 Hz during field testing. Messages to the signal controller may only be sent at the end of each loop iteration. Therefore, any changes made by RIO may take at most 1 second to be sent from the IIC.

- SNMP messages are rather slow and may take 1 second to be sent. SNMP messages may also queue up, causing the signal controller to show SigPT that is no longer consistent with RIO. This is noticeable between AR to green transitions.

- The way RIO handles SigPT is different from the way the signal controller does so. Whenever RIO needs to start or extend a phase, it sends a vehicle and/or pedestrian call to the signal controller. The signal controller then assigns SigPT based on its own logic. The signal controller needs to comply with its own boundaries for each signal phase duration. Also, the signal controller runs on an independent frequency from RIO. These factors may cause small timing differences between RIO's intended SigPT and what is generated by the signal controller.

5.7.1.3 Sensible

Across all scenarios, we aggregated and analyzed three different aspects of the timing for Sensible. Histograms with density estimations of the recorded timings are provided in Figure 6-33. First, we note that we estimated a ~0.8 lag in the Real Time Streaming Protocol (RTSP) video feed due to the design of the RTSP protocol. This lag is unavoidable. In the top left panel, we show the time taken from when a single RTSP video frame is consumed by Sensible and when a message is sent out to RIO based on this video frame. This accounts for processing the video frame as well as additional internal overhead due to internal message passing, etc. The median time is 0.39 seconds. The bottom panel shows the median time in seconds for the video tracker to process a single video frame, which was ~0.155 seconds. This translates to between 5-10 video frames per second. The top right panel shows we maintain a small communication time lag of ~0.5 seconds between the OBU outgoing messages and the message sent to RIO containing information from this OBU message.

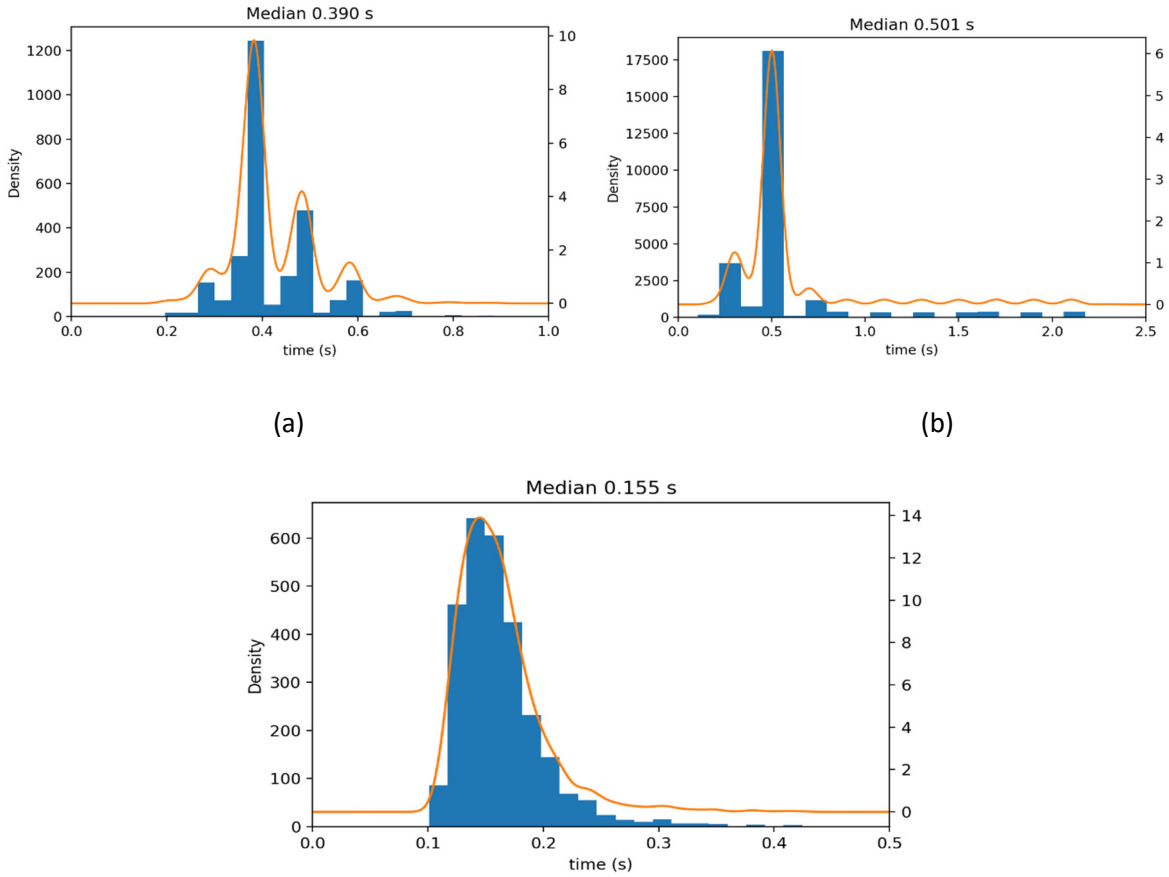


Figure 5-33: (a) Time from when an RTSP frame first arrives to being processed by the multi-sensor fusion and converted into a message for RIO; (b) Time between when a DSRC message is generated and when it gets processed by the multi-sensor fusion and converted into a message for RIO; (c) Time to process a single video frame by the video tracker.

5.7.1.4 RIO

The server used during the field tests was able to execute RIO at 1 Hz without noticeable delays. However, it has been noticed that the Pedestrian Push-Button GUI caused delay in other components and therefore we avoided using this during the TERL testing.

5.7.2 Other observations

This section reports and discusses observations specific to each system hardware or software component that did not pertain to the previous sections.

5.7.2.1 RIO

A few issues have been observed in RIO's algorithm throughout the tests. First, mobile app users reported that the departure time would increase as they got closer to the stop bar. This can also be observed in the scenarios' time-space diagrams. For example, see Figure 5-5 when vehicle 1 gets fused, and Figure 5-16, where this occurs multiple times for vehicle 4 and a final departure time extension occurs when the vehicle is about to enter the minimum distance to stop bar. Both cases share a common error where the vehicle timestamp and/or position gets offset right when the departure time gets changed, while DSRC-only vehicles do not seem to show this issue. Thus, the change in departure time seems to be a reaction from RIO when a vehicle's timestamp/position gets offset. While the ideal solution would be to prevent these offsets from occurring, this is unlikely to be achieved, given that these errors in the sensor fusion and video detection methods are almost impossible to be completely avoided with current technology.

It has also been observed that the all-red and yellow times can have an unintended duration, such as the all-red duration of 2 seconds in Figure 5-18. Even though this impacts the optimization framework within RIO, it is worth mentioning that the signalization seen by the drivers are not affected. This is because the signal controller follows its own configuration based on the vehicle/pedestrian calls sent by RIO, so the signal controller will maintain the correct yellow and all-red times.

It has been noticed that the push-button GUI can cause a significant reduction in both RIO and sensible while it is running. This seems to be a coding issue with the GUI's refresh rate which causes it to demand an unnecessary processing power to keep it updated.

5.7.2.2 Cohda radios

The communication between OBU and RSU works as intended when the whole system starts and a connected vehicle approaches an intersection for the first time. However, as the vehicle loops through the TERL facility to return to the intersection, the communication between the Cohda radios becomes less stable, and within two or three iterations the messages sent from the RSU are no longer shown in the mobile app. Thus, RIO continues to assign SigPT and recommend a speed and departure time, but the driver is unable to observe these values.

6 SUMMARY, CONCLUSIONS, AND RECOMMENDATIONS

6.1 Summary and conclusions

Previous work by the University of Florida and FDOT focused on the development of algorithms, software, and hardware solutions to enhance traffic signal control operations simultaneously with vehicle trajectories. RIO is an algorithm developed by the UFTI to optimize SigPT and vehicle trajectories. The original version of RIO receives real-time positioning of CNVs and CAVs, determines SigPT in order to better serve the current demand in the intersection, and sends back optimized trajectories such that vehicles depart from the stop bar at the saturation rate and at the maximum possible speed. However, the original version of the algorithm does not serve pedestrian demand, nor does it optimize vehicular throughput considering pedestrian presence. A natural extension of this work is to consider pedestrian movements and the fusion of additional existing sensors to refine the algorithm performance and further prepare it for field implementation.

The research team designed, implemented, and integrated a LiDAR-based pedestrian detection system into RIO. The system was not tested with bicycles and scooters due to an inability to collect extra data because of the COVID-19 pandemic. Nevertheless, the developed method is general enough to include those road users in the future. Overall, the performance of the pedestrian detection component was found to be satisfactory for our test scenarios. Pedestrians standing near the LiDAR at the intersection corner were reliably detected automatically, and entered into RIO as desired.

A message structure was designed to allow for source-independent communication of detected dynamic objects. Testing of this message structure showed low latency for a single detected object. Further testing is required to determine the scalability of the message in this respect. In cases where the dynamic object is being observed by both the intersection and the CAV, the large error between the two position measurements suggest incoming messages may not be useable for course planning by the CAV.

The research team developed pedestrian detection for the system together with pedestrian phasing capabilities. The LiDAR algorithm can detect and send the counts of pedestrians to RIO with a delay of roughly 2 seconds. RIO then uses these counts to assign signal phase and time to pedestrians while still serving vehicle demand.

The test scenarios provided additional information on how RIO responds to different traffic conditions, and it was shown that RIO can serve both vehicle and pedestrian demand in all cases presented. RIO can provide reasonable signal phasing and timing even when detection errors occur, such as when the video feed provides false positives or duplicates of vehicles. It was observed that the detection method affects the timing of when a vehicle is detected: DSRC can detect vehicles at a longer distance. Therefore, connected vehicles may be able to place a call to the controller and get the right-of-way sooner.

The research team observed that the detection method affects the timing of when a vehicle is detected: DSRC can detect vehicles at a longer distance. Therefore, connected vehicles may be able to place a call to the controller and get the right-of-way sooner.

Based on the results of the test scenarios and the feedback provided by the drivers, it was shown that the mobile app previously developed can enhance the driver's response by providing a suggested speed and departure time, potentially reducing acceleration/breaking times, and avoiding stopped time altogether.

6.2 Recommendations

It is recommended that the system is implemented in the field on a trial basis for further testing and refinement under real-world traffic conditions. The test intersection(s) should have adequate sensing equipment to ensure detection of vehicles and pedestrians from all approaches. Specific hardware and software improvement recommendations are provided below.

6.2.1 Hardware

6.2.1.1 Road-side unit

The RSU computer used during the tests was pushed to its limit. Sensible and RIO algorithms are highly demanding to the CPU, causing it to barely run the system at 1 Hz. While we have obtained good results at this frequency, it was only possible to do so by disabling Sensible recording of the video feed, and also disabling RIO pedestrian push-button GUI. Our total traffic demand was very small, however, and it is likely that the system will not be able to detect and optimize higher traffic demands. Moreover, the optimization algorithms would benefit greatly from running at higher frequencies, allowing RIO to react more quickly to changing traffic conditions rather than providing a single update in signal phase and time every one second. The drivers in connected vehicles would also receive faster updates for recommended speed and departure time, giving them more time to adjust their trajectories. Thus, it is highly desirable to obtain a new computer that can run RIO and its dependencies at higher frequencies.

6.2.1.2 LiDAR

The range of the LiDAR was insufficient to reach across the road to see the opposite corners of the intersection. As such, additional LiDAR devices are required to detect pedestrians within the entire intersection. Alternatively, a stronger LiDAR may be able to observe the entire intersection, but this would likely require it to be installed at a higher position than the current implementation.

6.2.2 Software

6.2.2.1 RIO

RIO does not have a lane-changing logic. Intersections with more complex geometry and higher vehicle demand would not get an accurate optimization of the trajectories. Adding this feature would allow us to serve higher vehicle demand.

The current system cannot read pedestrian calls coming from a real push button. While we managed to prove the concept of receiving pedestrian inputs from both the LiDAR and a GUI push button, it is necessary to implement this interface between RIO, the signal controller and the signal cabinet in order to fully service pedestrian demand at an intersection, particularly if there is not enough LiDAR coverage for the pedestrian waiting areas.

RIO currently assumes that conventional vehicles will be properly detected, receive signal phase and time (SigPT), and take the opportunity to depart. Should any of these three steps fail, a conventional vehicle could get stuck at the stop bar and never get served. This is because RIO ignores vehicles within the minimum distance to the stop bar in order to avoid abrupt changes to the SigPT. In order to deploy this system in the field, detection and suitable serving methods should be added for vehicles that have arrived at the stop bar undetected. An occupancy message can be added based on the video feed and suitable logic can be added in RIO to serve these stopped vehicles.

6.3 Implementation requirements

In order to implement this system, the following detection components should ideally be installed:

- Video-cameras with line of sight of each approach.
- LiDARs for pedestrian detection (pedestrian push buttons can provide pedestrian input for waiting areas outside of LiDAR coverage).
- Road Side Unit (RSU)/DSRC for providing signalization information to connected vehicles.

Figure 1 shows a schematic of the system's components and data flow. All data processing occurs at the intersection computer, which is located near or in the cabinet. The LiDAR readings for pedestrians are processed on a Intel NUC 6 that connects wirelessly to the Intelligent Intersection Controller (IIC) which allows the LiDAR to be deployed at a distance from the signal cabinet. The IIC processes the vehicle inputs and generates recommended vehicle trajectories and the signalization plan.

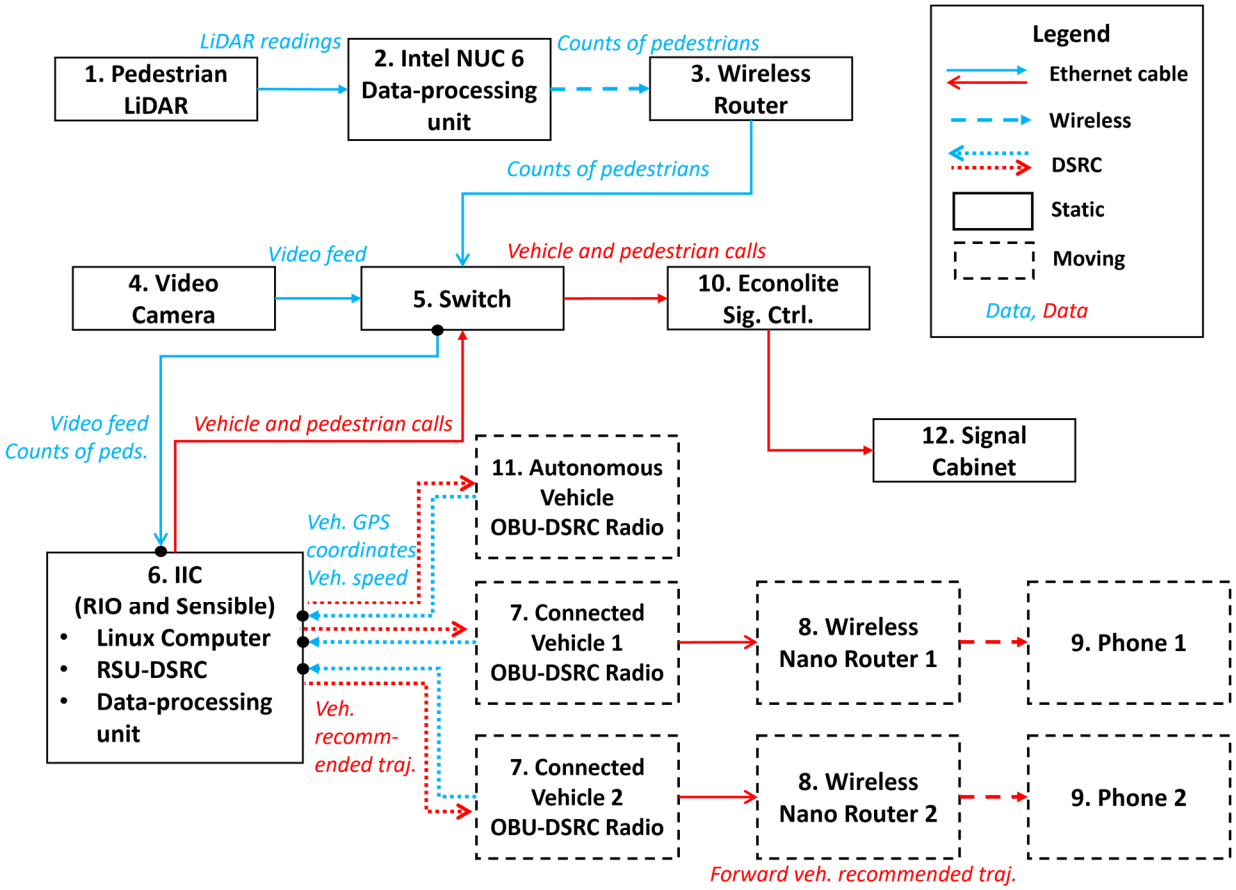


Figure 6-1: System components and data flow.

6.4 Safety and mobility effects

This project developed a system that uses CNV and CAV technologies to reduce congestion for vehicles and increase safety for pedestrians. Providing optimal vehicle trajectories results in better utilization of the intersection, which increases its throughput. At the same time, the system provides phasing and timing to match these trajectories, such that delays are further reduced. These two functions result in shorter queues, and lower overall delay. The detection of pedestrians and the use of that detection in the signalization allows for shorter delays for pedestrians, and therefore lower probability of jaywalking. It also results in better utilization of the intersection as green is not wasted when not needed.

6.5 Lessons learned and best practices

The project expanded the capabilities of an innovative system that optimizes signal control along with CNV and CAV trajectories, in order to accommodate pedestrians. The following summarize the lessons learned from this project:

- The SigPT solver developed can decrease vehicle delay with minimal impact to pedestrian delay, even when the traffic stream consists of conventional vehicles only. Performance improves with increasing CNV and CAV presence.
- Existing vehicle and detection systems are not yet sufficiently reliable, and therefore they require the use of sensor fusion techniques (which use combinations of different sensors) to maximize their effectiveness.
- Existing communication technology and products used in CNV applications are not yet adequately reliable for field deployment, as they require significant testing and calibration to operate.
- The project demonstrated that vehicles can receive a recommended trajectory through a phone app (our app was developed for an Android phone); however, the development of an effective user interface for communicating recommended trajectories to a driver requires extensive research to design it such that the driver can easily follow the recommendation.
- It is important to continue to test this system under a variety of field conditions and to ensure that all possible scenarios have been adequately addressed before full field deployment.

REFERENCES

Akyol, G., Erdagi, I. G., Silgu, M. A., & Celikoglu, H. B. (2020). Adaptive signal control to enhance effective green times for pedestrians: A case study. *Transportation Research Procedia*, *47*, 704-711.

Chang, C. C., & Lin, C. J. (2011). LIBSVM: a library for support vector machines. *ACM transactions on intelligent systems and technology (TIST)*, *2*(3), 1-27.

Comaniciu, D., & Meer, P. (2002). Mean shift: A robust approach toward feature space analysis. IEEE Transactions on *pattern analysis and machine intelligence*, *24*(5), 603-619.

Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine learning*, *20*(3), 273-297.

Ester, M., Kriegel, H. P., Sander, J., & Xu, X. (1996, August). A density-based algorithm for discovering clusters in large spatial databases with noise. In *kdd* (Vol. 96, No. 34, pp. 226-231).

Kidono, K., Miyasaka, T., Watanabe, A., Naito, T., & Miura, J. (2011, June). Pedestrian recognition using high-definition LIDAR. In *2011 IEEE* Intelligent *Vehicles Symposium (IV)* (pp. 405-410). IEEE.

Ma, W., Liao, D., Liu, Y., & Lo, H. K. (2015). Optimization of pedestrian phase patterns and signal timings for isolated intersection. Transportation Research Part C: Emerging Technologies, *58*, 502-514.

Mahadevan, K., Somanath, S., & Sharlin, E. (2018, April). Communicating awareness and intent in autonomous vehicle-pedestrian interaction. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing* Systems (pp. 1-12).

Matthews, M., Chowdhary, G., & Kieson, E. (2017). Intent communication between autonomous vehicles and pedestrians. *arXiv preprint arXiv:1708.07123*.

Muley, D., Alhajyaseen, W., & Kharbeche, M. (2017). An overview of pedestrian signal setting and implementation in the State of Qatar. Procedia Computer Science, *109*, 545-552.

Navarro-Serment, L. E., C. Mertz, and M. Hebert. "Pedestrian detection and tracking using three-dimensional ladar data." The International Journal of Robotics Research 29.12 (2010): 1516-1528.

Omidvar, A., Pourmehrab, M., Emami, P., Kiriazes, R., Esposito, J. C., Letter, C., ... & Ranka, S. (2018). Deployment and testing of optimized autonomous and connected vehicle trajectories at a closed-course signalized intersection. *Transportation Research Record*, *2672*(19), 45-54.

Pourmehrab, M., Optimizing Signalized Intersections Performance under Conventional and Automated Vehicles Traffic. Ph.D. thesis, University of Florida, Gainesville, FL, 2019.

Pourmehrab, M., Elefteriadou, L., Ranka, S., & Martin-Gasulla, M. (2019). Optimizing signalized intersections performance under conventional and automated vehicles traffic. *IEEE Transactions on intelligent transportation systems*, *21*(7), 2864-2873.

Qi, C. R., Su, H., Mo, K., & Guibas, L. J. (2017). Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 652-660).

Qi, C. R., Liu, W., Wu, C., Su, H., & Guibas, L. J. (2018). Frustum pointnets for 3d object detection from rgb-d data. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 918-927).

Rusu, R. B. (2010). Semantic 3d object maps for everyday manipulation in human living environments. *KI-Künstliche Intelligenz*, *24*(4), 345-348.

Thrun, S., Montemerlo, M., Dahlkamp, H., Stavens, D., Aron, A., Diebel, J., ... & Mahoney, P. (2006). Stanley: The robot that won the DARPA Grand Challenge. *Journal of field Robotics*, *23*(9), 661-692.

Wang, L., & Zhang, Y. (2016). LiDAR ground filtering algorithm for urban areas using scan line based segmentation. *arXiv preprint arXiv:1603.00912*.

Wang, Z., & Jia, K. (2019, November). Frustum convnet: Sliding frustums to aggregate local point-wise features for amodal 3d object detection. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (pp. 1742-1749). IEEE.

Yan, Z., Duckett, T., & Bellotto, N. (2017, September). Online learning for human classification in 3D LiDAR-based tracking. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (pp. 864-871). IEEE.

Yu, C., Ma, W., Han, K., & Yang, X. (2017). Optimization of vehicle and pedestrian signals at isolated intersections. Transportation Research Part B: Methodological, *98*, 135-153.

Zermas, D., Izzat, I., & Papanikolopoulos, N. (2017, May). Fast segmentation of 3d point clouds: A paradigm on LiDAR data for autonomous vehicle applications. In *2017 IEEE International Conference on Robotics and Automation (ICRA)* (pp. 5067-5073). IEEE.

Zhang, Y., Su, R., Gao, K., & Zhang, Y. (2017, August). Traffic light scheduling for pedestrians and vehicles. In 2017 IEEE Conference on Control Technology and Applications (CCTA*)* (pp. 1593-1598). IEEE.

Zhao, J., Xu, H., Liu, H., Wu, J., Zheng, Y., & Wu, D. (2019). Detection and tracking of pedestrians and vehicles using roadside LiDAR sensors. *Transportation research part C: emerging technologies*, *100*, 68-87.