Report No. UT-25.22

IDENTIFYING NEAR-MISSES (INCLUDING RED-LIGHT RUNNING) AND REDUCING CONFLICT THROUGH D-FYA AT SIGNALIZED INTERSECTIONS USING LIDAR SENSORS

Prepared For:

Utah Department of Transportation
Research & Innovation Division

Final Report
August 2025

RESEARCH
UDOT Keeping Utah Moving
Utah Department of Transportation - Research Division
4501 South 2700 West - P.O. Box 148410 - SLC,UT 84114-8410

DISCLAIMER

The authors alone are responsible for the preparation and accuracy of the information, data, analysis, discussions, recommendations, and conclusions presented herein. The contents do not necessarily reflect the views, opinions, endorsements, or policies of the Utah Department of Transportation or the U.S. Department of Transportation. The Utah Department of Transportation makes no representation or warranty of any kind, and assumes no liability therefore.

ACKNOWLEDGMENTS

The authors acknowledge the Utah Department of Transportation (UDOT) for funding this research, and the following individuals from UDOT on the Technical Advisory Committee for helping to guide the research:

- Mark Taylor
- Adam Lough
- Blaine Leonard
- Travis Evans
- Andrea Guevara
- Rudy Zamora

TECHNICAL REPORT ABSTRACT

1. Report No. UT-25.22	2. Government Accession No. N/A	3. Recipient's Catalog No. N/A

4. Title and Subtitle Identifying Near-Misses (Including Red-Light Running) and Reducing Conflict Through D-FYA at Signalized Intersections Using LiDAR Sensors	5. Report Date August 2025 6. Performing Organization Code N/A
7. Author(s) Pengfei (Taylor) Li (University of Texas at Arlington) Xianfeng (Terry) Yang (University of Maryland) Peter Zhu (University of Utah) Peirong (Slade) Wang (University of Texas at Arlington) Shrestha Sijan (University of Texas at Arlington)	8. Performing Organization Report No. N/A
9. Performing Organization Name and Address University of Utah Department of Civil & Environmental Engineering 110 Central Campus Drive, Suite 2000 Salt Lake City, UT 84112	10. Work Unit No. 5H09244H 11. Contract or Grant No. 238285
12. Sponsoring Agency Name and Address Utah Department of Transportation 4501 South 2700 West P.O. Box 148410 Salt Lake City, UT 84114-8410	13. Type of Report & Period Covered Final Oct 2022 to December 2025 14. Sponsoring Agency Code UT22.309

15. Supplementary Notes Prepared in cooperation with the Utah Department of Transportation and the U.S. Department of Transportation, Federal Highway Administration

16. Abstract

The primary research objective of this project is to demonstrate the potential capability of emerging LiDAR sensing technologies in identifying and mitigating traffic conflicts (i.e., near-misses) at signalized intersections. The latest LiDAR sensing technologies allow for tracking vehicles and pedestrians (assigning each a unique temporary ID for identification). This capability can be used to mitigate traffic crashes. In this research, the collaborating team between the University of Utah, the University of Texas at Arlington, and the University of Maryland installed the application software developed by UTA and conducted extensive tests in the field, such as the sensors' latency to identify instantaneous traffic conflicts, the algorithm's reliability and accuracy, and a demonstration of safety-centric traffic control algorithms The outcome of this research is expected to provide decision support for UDOT to determine its large-scale deployment plan.

17. Key Words Near-misses, red light running, LiDAR sensor, Pedestrian behaviors, Traffic signal systems.		18. Distribution Statement Not restricted. Available through: UDOT Research Division 4501 South 2700 West P.O. Box 148410 Salt Lake City, UT 84114-8410 www.udot.utah.gov/go/research		23. Registrant's Seal N/A
19. Security Classification (of this report) Unclassified	20. Security Classification (of this page) Unclassified	21. No. of Pages 71	22. Price N/A	

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ACRONYMS

ATSPM	Automated Traffic Signal Performance Measures
B/C	Benefit/Cost Ratio
CMF	Crash Modification Factor
CMFC	Crash Modification Factors Clearinghouse
CRF	Crash Reduction Factor
CUIP	Center of Urban Informatics and Progress
DR	Deceleration Rate
DOT	Department of Transportation
D-FYA	Dynamic Flashing Yellow Arrow
ET	Encroachment Time
FHWA	Federal Highway Administration
FYA	Flashing Yellow Arrow
GT	Gap Time
HCM	Highway Capacity Manual
HSM	Highway Safety Manual
IAC	Inter-Agency Contract
IAPT	Initially Attempted Post-Encroachment Time
ISC	Intersection Safety Challenge
LiDAR	Light Detection and Ranging
MLP	Multilayer Perceptron
MTCC	Minimum Time to Collision
MUTCD	Manual on Uniform Traffic Control Devices
NHTSA	National Highway Transportation Safety Administration
NI	No Injury
NLT	No Left Turn
NRT	No Right Turn
PRM	Permitted Left Turn
PSD	Proportion of Stopping Distance
PET	Post Encroachment Time
RF	Random Forest

SENSR-I	SENSOR for Infrastructure
SI	Severe Injury
SVM	Support Vector Machine
TAC	Technical Advisory Committee
TTC	Time To Collision
UDOT	Utah Department of Transportation
UNR	University of Nevada, Reno
UTA	University of Texas at Arlington
UTRAC	Utah Transportation Research Advisory Council
XGBoost	Extreme Gradient Boosting

EXECUTIVE SUMMARY

This study investigated several critical issues when using a LiDAR sensing solution to detect and analyze near-miss traffic events at signalized intersections and demonstrate a LiDAR-based dynamic flashing yellow arrow. A near-miss refers to a “close call” where two vehicles (or a vehicle and a pedestrian) almost collide but avoid impact through last-second maneuvers. By identifying these events in real time, transportation agencies can proactively address safety concerns before crashes occur.

The research had two main goals: (1) demonstrate how and whether LiDAR can instantly detect near-misses using high-resolution trajectory data without needing complex computations, and (2) demonstrate a LiDAR-based safety-centric traffic signal control strategy called Dynamic Flashing Yellow Arrow (D-FYA), which uses live pedestrian tracking to reduce conflicts with permissive left-turn vehicles.

Despite hardware challenges and failures that impacted the full demonstration of D-FYA, the team successfully verified the LiDAR-based detection system through real-world testing and video validation. The system showed high accuracy and low latency (around 0.13 seconds). Hundreds of near-misses were identified per day at a single intersection, offering strong evidence for its potential.

Based on findings, the researchers recommend UDOT consider wider deployment of LiDAR-based traffic detection systems at intersections to improve traffic safety. They also compare two different D-FYA strategies using actual behavior data and traditional push buttons to drive real-time signal control. Overall, the research highlights the promise of LiDAR in transforming intersection safety by detecting risks before crashes happen.

1 INTRODUCTION

1.1 Problem Statement

This research demonstrates a state-of-the-art LiDAR sensing technology designed to detect traffic near-misses—also known as traffic conflicts—in real time at signalized intersections. A near-miss is defined as a situation in which two or more travelers (vehicles, pedestrians, cyclists) are on a collision course but avoid a crash through last-second evasive maneuvers. Prior studies have established a strong correlation between the frequency of near-misses and the likelihood of future crashes. Therefore, the identification of near-misses can serve as a proactive indicator of crash potential, enabling early intervention before actual collisions occur.

Near-misses have two important dimensions: frequency and severity. Frequency refers to how often near-misses are observed at a given location; high frequency suggests a high risk of imminent crashes, necessitating prompt corrective measures. Severity is defined as the relative kinetic energy, based on the square of the relative velocity and the masses of the conflicting vehicles, integrated into the Crash Potential Index (CPI). This index combines the probability of a collision, derived from the Minimum Time to Collision (MTCC), with a severity factor to assess both the likelihood and potential impact of crashes in traffic simulations. Even if less frequent, severe near-misses warrant immediate attention due to their potential impact. Both aspects are critical for understanding safety conditions and formulating effective responses.

In terms of traffic control, crash mitigation strategies at signalized intersections can be either “responsive” or “adaptive.” Responsive strategies are based on recently observed near-misses (e.g., within the last five minutes), while adaptive strategies rely on short-term predictions (e.g., for the next five minutes). These strategies can further be classified as “collective” or “instantaneous.” Collective measures address general crash risk patterns, whereas instantaneous measures respond to specific near-miss events in real time. For example, a collective response may adjust time-of-day signal plans, while an instantaneous response could involve triggering an all-red interval to prevent a red-light violation by stopping predicted near-miss vehicles from entering the intersection.

This project evaluates and validates a novel method developed to identify instant near-misses and their severity, as well as to predict potential crashes based on vehicle speeds and proximities with the deployed LiDAR system. The method uses high-resolution trajectory data but requires minimal computing power and avoids complex tracking algorithms. Initial results are promising and support the development of safety-centric traffic-signal control strategies designed to prevent crashes in real time.

1.2 LiDAR Hardware Deployment

The LiDAR hardware platform used in this study was inherited from a previous UTRAC research project focused on LiDAR applications at signalized intersections. The commercial hardware components, including LiDAR sensors and an edge computing unit, were manufactured by Koito-Cepton Inc. (formerly Cepton, Inc.). The specific sensor model, P6o, provides a horizontal field of view of 60 degrees. To achieve full intersection coverage, four LiDAR sensors were strategically deployed to provide comprehensive spatial monitoring.

An edge computer installed in the roadside cabinet receives and fuses raw point cloud data from all mounted sensors. It processes the data in real time to extract trajectories of moving objects, including both vehicles and vulnerable road users. The software developed by the research team operates on this platform, synchronizing object detection with real-time traffic signal status. This integrated system enables the identification of relevant traffic events—such as near-misses—and supports both real-time interventions and post-analysis evaluations.

For further details on the hardware configuration and foundational methods, readers are encouraged to consult UTRAC Research Report UT-22.26, titled "*Utilizing LiDAR Sensors to Detect Pedestrian Movements at Signalized Intersections*."(*1*).

1.3 Pilot Projects Using LiDAR for Traffic Signal Systems in North America

As per UDOT's request, the research team conducted a literature review and internet search to summarize the previous pilot projects on applying LiDAR to traffic signal systems in the US. It should be pointed out that the sources of this summary vary, ranging from the most credited news press website to LiDAR manufacturers' own websites and casual user forums.

Therefore, readers are advised to further verify if some pilot projects are considered particularly interesting.

1.3.1 Velodyne/Ouster Sensor/Perception + BlueCity.ai Traffic Analytics Software

- In July 2021, Velodyne began a LiDAR-based traffic monitoring trial at the intersection of East 7th Street & Springdale Road in Austin, Texas — a known high-injury location in the city's Vision Zero network. The integration software was BlueCity.
- In May 2021, Velodyne began piloting its Intelligent Infrastructure Solution (IIS), integrated with BlueCity AI, at several intersections in New Brunswick, New Jersey.
- In August 2022, Michigan's Michigan Mobility Funding Platform (MMFP) funded a pilot deployment for BlueCity and Velodyne, partnered with the University of Michigan and MCity to deploy BlueCity's real-time traffic monitoring solution at five signalized intersections in Ann Arbor and surrounding areas.
- In March 2024, the City of Chattanooga (alongside the University of Tennessee at Chattanooga's Center for Urban Informatics & Progress) received a $2 million SMART "Planning & Prototyping" Stage 1 Grant through the U.S. Department of Transportation's SMART program, funded by the Bipartisan Infrastructure Law, to deploy C-V2X and ITS technologies, including LiDAR-enabled intersections, to improve pedestrian and multimodal safety around downtown and mid-block crossings.
- In March 2025, Utah DOT selected Econolite to deploy the LiDAR system after evaluating multiple vendors. Econolite will integrate Ouster 3D digital LiDAR with the BlueCity AI platform, paired with Econolite's Cobalt® controllers and EOS firmware. The initial phase includes 15 BlueCity-equipped systems, part of a 5-year statewide project featuring dynamic signal actuation, vulnerable road user detection, and V2X messaging. The funding source is from UDOT's operational budget.

1.3.2 Seoul Robotics Perception + BlueBand Traffic Analytics + Various LiDAR Sensors

- In 2023, Seoul Robotics partnered with Chattanooga's CUIP and FHWA for one of the largest U.S. smart intersection networks, then covering ~130 intersections from 2023–2024., creating a city-scale 3D perception network, funded through a $60 million FHWA award.
- In March 2024, Seoul Robotics, in partnership with Gades Sales Company and BlueBand, deployed a LiDAR-controlled traffic signal in the U.S. at the intersection of State Street & 5900 South in Murray, Utah. The system used four corner-mounted LiDAR sensors feeding into SENSR-I, Seoul Robotics' edge-based 3D perception engine. BlueBand software translated the real-time perception data into NTCIP-compliant commands for an existing signal controller—no trenching or rewiring was required. The project was initiated under UDOT's operational budget as a testbed for scalable smart intersection technology.
- In April 2025, 5 additional intersections were deployed with the same solution. The locations are 7800 South and 2200 West, State Street and 6100 South, Redwood Road and 4700 South, Wall and 20th Street, and Wall and 23rd Street.

1.3.3 The University of Nevada, Reno (UNR)+ Velodyne/Ouster LiDAR Sensors

- In February 2017, UNR's Civil & Environmental Engineering team—led by Associate Professor Hao Xu—installed what's believed by many to be the world's first roadside LiDAR sensor at Virginia Street & 15th Street in Reno. This Velodyne LiDAR setup created three-dimensional traffic trajectories, enabling detection of near-crashes, speeding, lane changes, and pedestrian movements. It was part of the Nevada Living Lab in partnership with Washoe County RTC and the City of Henderson, funded by state/regional transportation agencies, including NDOT and RTC Washoe County.
- From 2018–2020, UNR expanded the network to eight sensors along Virginia Street and into Henderson, NV, as part of intelligent mobility projects.

1.3.4 The University of Texas at Arlington (UTA) + Cepton LiDAR Sensors

- From 2020 to 2023, researchers at the University of Texas at Arlington (UTA)—through the UTA's internal funds and a USDOT-funded NITC study—deployed Cepton's Helius® Smart LiDAR System at two high-pedestrian intersections: one in Arlington (between UTA campuses) and another in Irving (near a high school). This research led to many technology-transfer activities like webinars and seminars hosted by NITC UTC and FHWA.
- From 2021–2022, UTA's pilots in Texas led to a UDOT-sponsored collaborative pilot deployment with the University of Utah at 600 North & 300 West in Salt Lake City. It monitors pedestrian walking speeds, average waiting time, and effective perception-reaction to WALK to provide decision support for pedestrian facility design and evaluation.
- The current project was kicked off in 2022 and extended the previous LiDAR project to near-miss evaluations and a D-FYA demonstration at the same intersection (600 North & 300 West in Salt Lake City).
- From March 2025 to July 2025, UDOT supported the University of Texas at Arlington to deploy a separate edge computing device to connect Seoul Robotics' output and collect driving behaviors during yellow and the first 10-second red at State Street and 5900 South in Murray, Utah. This data collection was for the FHWA pooled-fund study: *"Yellow Change and Red Clearance Intervals (CCI)*"(*2*). The collected data formed the foundation of two manuscripts on this topic. The data collection system worked as expected until July 2025, when an unexpected hardware failure on the research edge computer caused UDOT's LiDAR detection system to go offline for an extended period. After this incident, UDOT confirmed that the researchers had collected sufficient data for their studies and subsequently disconnected the research devices from the UDOT traffic network.

1.3.5 Texas DOT's LiDAR Pilot Project

- In 2024, Texas DOT (TxDOT) submitted a concept paper to the USDOT Intersection Safety Challenge (ISC) program. TxDOT was awarded for a prize and later flowed this fund to the University of Texas at Arlington via an Inter-Agency Contract (IAC) to further demonstrate and prototype the proposed full-spectrum LiDAR-based traffic safety solutions. This effort catalyzed the first pilot project of its kind in TxDOT. The adopted LiDAR sensors were Luminar's IRIS sensor (120-degree FOV), Seoul Robotics' SENSR perception software. The university researchers will develop the proposed applications on top of SENSR. The project is expected to finish in the summer of 2026.

1.3.6 SEYOND LiDAR and its own SIMPL Perception/Applications

- In October 2024, Seyond launched its SIMPL platform in Columbia County, Florida, achieving 99% vehicle detection accuracy at a four-leg intersection with three lanes in each direction.
- On October 25th, 2024, UDOT deployed the Seyond LiDAR solution at the intersection of 700 East and 1300 South; it has been fully operational ever since the deployment.
- In May 2025, Peachtree Corners, Georgia (in partnership with Curiosity Lab), deployed SIMPL at Technology Parkway.
- In May 2025, Seyond signed a distributing contract with Twincrest Technologies based in Texas. Twincrest has deployed Seyond's turn-key solution at a few locations for demonstration.

1.4 Objectives

This project pursued two primary research objectives. (I) The first objective was to evaluate the concept of LiDAR-based, instantaneous near-miss identification at signalized intersections. This approach aims to detect potential collisions in real time, enabling proactive safety interventions before crashes occur. (II) The second objective was to assess an augmented traffic control algorithm designed to separate crossing pedestrians from permissive left-turning vehicles. This algorithm modifies the flashing yellow arrow (FYA) operation on a cycle-by-cycle basis by delaying or canceling the FYA indication when necessary. The approach is referred to as Dynamic Flashing Yellow Arrow (D-FYA).

Unlike existing systems, which typically assume pedestrian behavior based on static inputs, the adopted D-FYA algorithm in this project responds directly to observed pedestrian movements tracked by the LiDAR system. For example, pedestrians often change their intentions while waiting, such as crossing a different crosswalk that activates first or choosing not to cross at all. The D-FYA system accounts for these real-world behaviors and adjusts signal timing accordingly. In contrast, current push-button-based systems may be unresponsive to pedestrian presence (e.g., the “FYA Delay”) or unresponsive to pedestrian behavior (e.g., the “minus pedestrian call” logic in FYA configuration), resulting in less effective conflict mitigation between pedestrians and turning vehicles.

1.5 Scope

The scope of this research is divided into four tasks, including preliminary investigation (kick-off meeting, literature review, definition of context-aware conflict zones); LiDAR Sensor Latency Evaluation – assessing the responsiveness and timing accuracy of LiDAR data for real-time applications; LiDAR-Based Near-Miss Identification – validating the detection of near-miss events through cross-referencing LiDAR-reported conflicts with synchronized video recordings; Demonstration of Dynamic Flashing Yellow Arrow (D-FYA) – implementing and evaluating the cycle-by-cycle decision algorithm for pedestrian-vehicle separation in a field setting.

1.5.1 Preliminary Investigation

The research team and members of the Technical Advisory Committee (TAC) discussed and confirmed the proposed project tasks during the initial kick-off meeting, including a review of the necessary resources to ensure successful execution. While most of the technical infrastructure was inherited from a previous UTRAC project, all key components were re-examined and verified for compatibility and readiness. The confirmed resources included:

- VPN access credentials for secure connection to traffic control equipment behind UDOT's firewall.
- A virtual server within UDOT's traffic control network, configured to receive and archive data from deployed LiDAR sensors and host the central application software.
- Verification of LiDAR sensor operational health and calibration status.

Attendees of the kick-off meeting included project investigators from the University of Utah, the University of Maryland, and the University of Texas at Arlington; the UDOT research project manager; and members of the TAC, comprising staff from various UDOT divisions.

1.5.2 Literature Review

A literature review is conducted to understand the state of the art of traffic safety and traffic conflicts studies.

1.5.3 Traffic Conflict Zone Design within Intersections

Traffic conflicts within intersections are not isolated to single points but are better represented as zones derived from LiDAR-captured vehicle trajectories. Based on road geometry and vehicle kinematics, five distinct types of conflict zones are defined, covering both vehicle-to-vehicle and vehicle-to-pedestrian interactions.

1.5.4 Evaluating LiDAR Perception's Latency

Instantaneous near-miss identification requires low-latency LiDAR perception. This task developed a framework to evaluate LiDAR perception latency using synchronized video recordings.

1.5.5 Verification of LiDAR-Reported Near-Miss with Recorded Video

LiDAR-based near-miss identification relies on perceived object trajectories extracted from point clouds. To ensure accuracy and responsiveness, the LiDAR perception system aggressively filters out irrelevant objects and background noise, outputting only the target relevant objects, such as vehicles or pedestrians. However, this filtering process limits the availability of contextual information needed to fully investigate the causes of near-misses. This task conducted a case study to demonstrate how LiDAR-reported near-misses can be verified using corresponding video recordings at intersections.

1.5.6 Demonstration of an Advanced Dynamic Flashing Yellow Arrow (D-FYA)

The dynamic flashing yellow arrow (D-FYA) in this context refers to a novel approach for separating crossing pedestrians from permissive left-turn vehicles, aiming to enhance safety without significantly reducing left-turn capacity. Unlike existing D-FYA systems, this implementation leverages LiDAR sensors to dynamically track pedestrian presence and movement during the WALK phase. The project team initially planned a comprehensive data collection effort and live demonstration of this D-FYA concept. However, due to repeated LiDAR sensor failures beginning in early 2023, this task could not be completed on schedule. The persistent hardware issues caused substantial project delays, and despite several major salvage attempts, the failures could not be resolved. As a result, UDOT approved a simplified scope for Task 4 to facilitate project closure. In light of the revised objectives, the project team has documented the D-FYA mechanism and provided a comparison with a newly developed push-button-based D-FYA system designed by UDOT's traffic signal group. The project team also explains how to implement similar operations on the current traffic signal systems.

1.5.7 Recommendations and Conclusions

The research team summarizes the key research findings and makes recommendations for a left-turn phasing design, based on the research outcomes. The research team also offers some suggestions on large-scale deployment of this technology to improve traffic safety.

1.5 Outline of Report

This project report is organized with the following chapters:

- Introduction
- Literature review
- Traffic conflict zone design
- Latency evaluation of LiDAR perception
- Verification of LiDAR-reported near-miss with recorded video
- Demonstration of dynamic flashing yellow arrow (D-FYA)
- Conclusions

2 LITERATURE REVIEW

Traffic near-misses, or traffic conflicts in other literature, refer to potential crash situations when conflict vehicles are in an imminent situation of collision but take evasive actions to avoid it. The concept of traffic near-misses can be traced back to 1968 when Perkins and Harris from General Motors observed and summarized traffic conflicts at intersections (*3*). Other early studies include Older and Spicer, who categorized the observed traffic near-misses at various road locations into different types (*4*); Baker confirmed the close association between near-misses and crashes based on the data collected at 392 intersections and pointed out the benefits of traffic conflict analysis for traffic safety at rural areas (*5*); Glauz et al. conducted an extensive survey and summary on traffic conflict analysis and practice in the US (*6; 7*). Among those early studies, tabulating the numbers and frequencies of observed traffic near-misses and their association with real crashes is the main finding. To reveal the other element of traffic safety, it is required to understand the severity of each near-miss. Gettman and Head summarized seven indicators of near-miss severity (*8*): gap time (GT), encroachment time (ET), deceleration rate (DR), the proportion of stopping distance (PSD), post-encroachment time (PET), initially attempted post-encroachment time (IAPT) and time to collision (TTC). Among these indicators, TTC (*9; 10*) and PET (*11*) are the most popular because they can be easily measured from the vehicle trajectories. The difference between TTC and PET is whether the late following vehicle's (v2) deceleration is considered.

There are also other variants of TTC, such as the modified time to collision or (Minimum Time to Collision) MTCC due to Ozbay et al. (*12*). The data sources for near-miss identification include loop detectors (for longitudinal near-misses), video detection and tracking, radar, or LiDAR detectors. The algorithms used for near-miss identification and prediction include regression, Bayesian, and artificial intelligence techniques etc. As summarized by Hossain et al. (*13*), the literature on traffic near-miss studies are prolific and still an active research area.

It becomes increasingly appealing for real-time estimation and prediction of traffic near-misses, rather than using historical data. Being “real-time” is a relative concept. It can refer to (I): A short period like a 5-minute time window or (II): instantaneous when the time window is approaching zero (e.g., less than 0.1 s). Achieving instantaneous identification is challenging and restricted by many factors, such as computing resources and algorithms, but it can pave the road for novel safety-centric traffic control strategies.

Cai et al. used a microscopic vehicle detection system on freeways to measure vehicles' longitudinal maneuvers and identify near-misses. They applied a Bayesian multilevel logistic regression to estimate the possibilities of crashes using the Bernoulli distribution (*14*). Wang et al. adopted a similar framework, but they considered a more macroscopic feature, such as trip generation and socio-demographic information (*15*). Zheng and Sayed retrieved vehicle trajectories from video cameras at signalized intersections to identify near-misses and developed a generalized extreme value (GEV) model to predict at intersections. They also derived two new safety indices, the risk of crash (RC), and the return level of a cycle (RLC). The developed method was validated with observed crashes (*16*). Athanasios et al. used the loop detector data containing vehicle headways and speeds and crash records to generate a training data set. They also applied multiple machine learning and deep learning models to examine various models' performance and concluded that the deep learning models perform better than the traditional machine learning models in crash prediction (*17*). Basso et al. used two deep-learning models to capture the nuanced difference among vehicles within video detections and developed a training data set to estimate crash potentials. They also adopted oversampling techniques to increase the importance of rare crash data to make the framework more effective (*18*). Yuan et al. applied the deep learning model not only to estimate the crash risk but also to predict the crash risk in the near-term future (*19*). They adopted a long short-term memory recurrent neural network (LSTM-RNN) model and generated the training data set with oversampling techniques for the rare crash data.

A feature in preparing the training data is that the input variables include the Bluetooth-based travel time and automated traffic signal performance measure (ATSPM) data. The prediction accuracy is reported as 60%. Li et al. further enhanced this framework and increased the prediction accuracy up to 88% (*20*). Arash and Ahmed used the connected vehicle (CV) data and crash record to prepare the data set for training various logistic regression models to predict crashes from a rural CV testbed in Wyoming. The input variables include continuous and categorical ones, most of which were associated with speeds and volumes (*21*). To overcome the rare-event nature of crash records, Peng et al. adopted the Youden Index method to adjust the classification threshold in the crash prediction models and the experiments show better performance and accuracy than the original data set (*22*). Li and Abdel-Aty expanded the crash prediction to secondary crashes. Using the spatio-temporal thresholds which are 15 minutes after a crash occurred and up to 1,600 meters from the crash site, secondary crashes are first identified. In the meantime, the corresponding traffic speed, volume, and lane occupancy were collected with roadside detectors on freeways. After the training data were generated for the XGBoost model, the reporting accuracy reached 80% (*23*). Thapa et al. divided road links into cell segments over time. The crash samples are aggregated into cell segments with small time windows. This is a new sampling technique and the results show they can reduce the samples by 25% to achieve a similar performance, interpreted into a reduced computational load (*24*).

Yu et al. developed a new tensor structure to construct the inputs of training data sets and adopted refined-focal loss functions for the imbalanced data issue. Using the collected data, the proposed model obtained 67% accuracy and four false alarm rate (*25*). Using 28,000 investigation results of frontal vehicle collisions, Wang et al. adopted a deep neural network model to extract kinematic features and then predict crash risk with a support vector machine or SVM model accordingly (*26*). The prediction accuracy was reportedly 85.4% with a latency of less than 1.2 milliseconds. Shuangguan et al. predicted crash risks by observing and extracting drivers' behaviors. Using the naturalistic data set and four machine learning models (XGBoost, SVM, RF, and MLP), drivers' crash potentials are predicted (*27*).

There is additional research literature on real-time crash prediction. The review in this paper is recent and selective. Nonetheless, a common pattern that is found is that most literature is driven by big data sets and various regression, machine learning, and deep learning models. The output will be the predicted overall/statistical collision risk (e.g., five minutes ahead) for all vehicles. While this information is important to support traffic managers in reducing crashes, it may not necessarily identify and predict individual imminent crashes and their severity. In the meantime, the real-time near-miss events would be a fundamental input for safety-centric traffic signal systems at intersections. This paper distinguishes itself from other literature by presenting a method to identify instantaneous near-misses and predict crash severity based on state-of-the-art LiDAR-tracked trajectories, including both vehicles and vulnerable road users (VRUs).

3 TRAFFIC CONFLICT ZONES WITHIN INTERSECTIONS

3.1 Conflict Points vs. Conflict Zones at Intersections

Collisions at intersections occur more frequently than those on highway segments because there are more traffic conflicts at intersections. Traffic conflict point analysis (TCPA) is the foundation of traffic safety analysis and control design for intersections. While it is commonly accepted by practitioners and academia, the TCPA method, by nature, ignores vehicles' lateral maneuvers and drivers' random decisions to change lanes while moving within intersections.

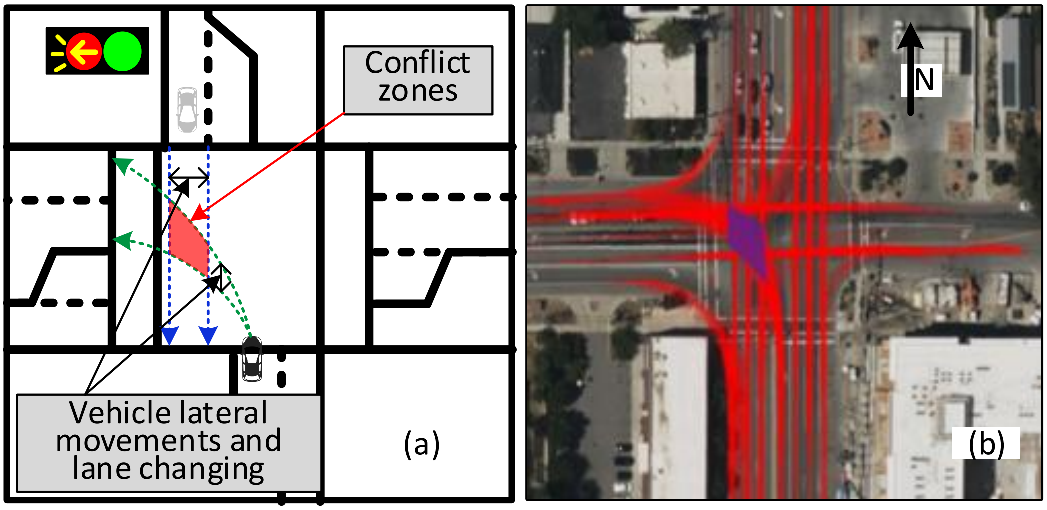


Figure 3-1 Demonstration of conflict zones due to random driving behaviors and lateral maneuvers

Fig. 3-1-a demonstrates one of the conflict zones between permissive left-turn vehicles and opposing through vehicles. In the real world, approaching vehicles make random maneuvers within intersections, creating "conflict zones" instead of "conflict points." Fig. 3-1-b is a plot of 1% of hourly trajectories of all approaching vehicles to an intersection in Salt Lake City (Utah). It clearly shows the random maneuvers of vehicles and the existence of "conflict zones." In light of these thoughts, we derive a new definition of traffic conflicts or traffic near-misses based on the conflict zones in this study, referred to as the traffic conflict zone analysis (TCZA) method. Each traditional conflict point will be stretched into a conflict zone based on random maneuvers of conflicting vehicles.

3.2 Conflict Types at Intersections

A traffic conflict or “a near-miss” occurs when conflicting vehicles are too close to avoid collisions comfortably. There are two causes for near-misses: (I) **Ignorance** and (ii) **Misperception**. Ignorant drivers may fail to see conflicting vehicles or pedestrians in time, and so they must make hard brakes to avoid collisions, such as near-misses between permissive right-turn vehicles and crossing pedestrians or between permissive left-turn vehicles and crossing pedestrians. On the other hand, drivers may also misperceive imminent traffic conflicts, causing small gaps between conflicting vehicles, such as near-misses between permissive left-turn vehicles and opposing through traffic or red-light running (failing to pass the stop line before the yellow’s expiration).

The near-misses due to ignorance can occur regardless of vehicles’ instantaneous speeds, while the near-misses due to misperception often occur at relatively high speeds. Therefore, we distinguish the ignorance-type near-misses from misperception-type near-misses in this study. The near-misses of interest at intersections are grouped as follows:

- Near-misses due to ignorance:
    - Permissive right-turn vehicles vs. crossing pedestrians (Fig.3-2 (a))
    - Permissive left-turn vehicles vs. crossing pedestrians (Fig.3-2 (c))
    - Permissive right-turn on red vs. crossing pedestrians (Fig.3-2 (f))
    - U-turn vs permissive RT vehicles (Fig.3-2 (e))
- Near-misses due to misperception:
    - Permissive left-turn vehicles vs. opposing through vehicles (Fig.3-2 (b))
    - Red-light running (entering intersections after the yellow expires) (Fig.3-2 (d))

Note that, even though frequent near-misses suggest potential collisions, an individual collision may not necessarily have a corresponding near-miss occurrence. Near-misses are the result of drivers trying their best to avoid collisions, whereas the causes of real collisions are much more complicated.

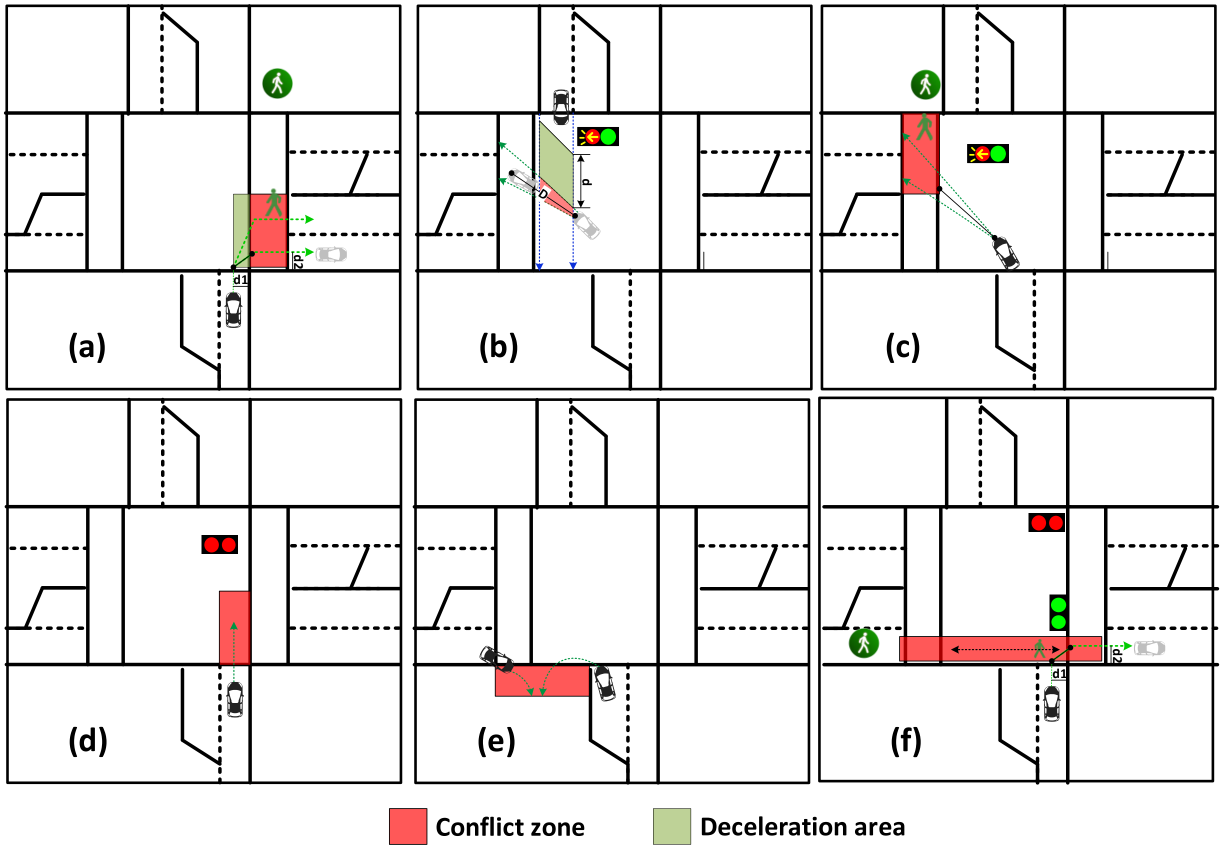


Figure 3-2 Demonstration of near-misses of interest at intersections

3.3 Kinematics Analysis for Near-Misses at Intersections

3.3.1 Near-Misses between Permissive Right-Turn Vehicles and Crossing Pedestrians

Right-turning (RT) vehicles should unconditionally yield to pedestrians and stop when a pedestrian is crossing. Assuming an RT vehicle fails to see a concurrent pedestrian and passes the stop line at speed v_0 (miles per hour), then its (linearized) stopping distance can be calculated as (*28; 29*):

$$d = 1.47 v_0 t_0 + \frac{v_0^2}{30\left(\frac{11.2}{32.2} \mp G\right)} \qquad (3\text{-}1)$$

Where: d is the total stopping distance; t_0 is the perception-reaction time; G is the grade in percentage; 11.2 (ft/s^2) is the maximal deceleration rate recommended by AASHTO and 33.2 (ft/s^2) is the gravitational acceleration rate. As shown in Fig. 3-2-a, the shortest stopping distance in the buffer zone will be

$$d_{min} = \sqrt{\left(\frac{d1}{2}\right)^2 + \left(\frac{d2}{2}\right)^2} \quad (3\text{-}2)$$

Where d_1, d_2 are the lengths from curbs to the centerline of the outermost lanes, if $d > d_{min}$Then the RT vehicle cannot stop out of the conflict area, and it generates a near-miss.

3.3.2 Near-Misses Between Permissive Left-Turn (LT) Vehicles and Crossing Pedestrians

This type of near-miss is illustrated in Fig. 3-2-c, for an imminent conflict with the concurrent crossing pedestrians, the LT vehicle will either brake very late or not brake at all due to concerns of right-angle collisions. In this case, if a crossing pedestrian and an LT vehicle appear in the conflict zone at the same time, a near-miss event occurs.

3.3.3 Near-Misses Between Permissive Left-Turn Vehicles and Opposing Through Vehicles

A permissive left-turn vehicle becomes in conflict with opposing through vehicles when its front end enters the opposing through lanes until its rear end leaves that area. During this hazardous process, the left-turn vehicle's total (linearized) travel distance *D* and travel time *t* can be roughly estimated as:

$$D = \delta(L + N \times w);\ t = d/v_0 \quad (3\text{-}3)$$

Where L is a vehicle's length, N is the number of lanes, w is the lane width, δ is an empirical factor to consider the LT vehicle's curvy movement and curb spaces; and v_0 is the LT vehicle's speed.

Whenever an LT vehicle starts to turn, we assume that the driver has perceived acceptable gaps and decided on a safe crossing speed. It is also reasonable to assume that the LT vehicle has no chance to adjust its decision during its turning maneuvers due to the impact of centrifugal force and limited vision, etc. Therefore, if the LT vehicle's perception is wrong, then collision avoidance will mostly rely on the responses of the opposing through vehicles. When an opposing through (TH) vehicle perceives a potential collision with a permissive LT vehicle, the TH vehicle will slow down to ensure it takes at least t before entering the conflict zone. Assuming the TH vehicle takes the maximal deceleration rate, its total travel distance d during t is at least:

$$d = 1.47 v_0 t_0 + \frac{\left(v_0^2 - \left(v_0 - 11.2\,(t - t_0)\right)^2\right)}{30\left(\frac{11.2}{32.2} \mp G\right)} \quad (3\text{-}4)$$

From Eq. (3-4), an opposing through vehicle approaching at v_0 will need at least d from the conflict zone to avoid a collision. (See Fig. 3-2b).

3.3.4 Red-Light Running (RLR), a Special "Near-Miss" with Yellow

RLR is a special traffic conflict at signalized intersections, and we consider a red-light runner has a "near-miss" with the yellow. As shown in Fig. 3-2-d, if a through vehicle enters the conflict zone at a speed that is too high for turning maneuvers during red, then this vehicle becomes a red-light runner and generates an RLR event.

3.3.5 Near-Misses Between U-Turn Vehicles and Permissive Left-Turn Vehicles

Vehicles may perform U-turns during either the flashing yellow arrow phase or the protected left-turn phase. Simultaneously, vehicles from the opposing approach may execute permissive right turns. In practice, such scenarios often lead to near-miss events, where drivers are forced to make abrupt maneuvers to avoid collisions, even at low travel speeds. As shown in Fig.3-3e, a near-miss can be identified once it enters the conflict zone at the same time.

3.4 Identify the Severity of Near-Misses by Scoping the Conflict Zone with TTC and PET

The severity of near-misses may or may not be associated with conflicting objects' speeds. For near-misses involving slow VRUs or vehicles, the severity of near-misses between vehicles and VRUs can be solely determined by their proximity. If the conflict zone is scoped small, then only those dangerously proximate near-misses will be captured. With a larger conflict zone, more near-misses will be captured, including both severe and less severe near-misses.

For high-speed, vehicle-to-vehicle near-misses (e.g., permissive LT vehicles vs. opposing through vehicles), the near-misses are evaluated by whether the vehicle(s) must take evasive actions to avoid collision. This criterion involves both proximity and speed. The severity of high-speed near-misses is important to estimate the vehicle-to-vehicle crash risks. For instance, a "bumper-to-bumper" near-miss is more dangerous than one with two seconds of PET[1]. Given that

[1] Post Encroachment Time (PET) is a traffic safety indicator that measures the time interval between a leading vehicle leaving a conflict zone and a following vehicle entering that same zone, essentially measuring how close two vehicles came to a collision without actually crashing.

state-of-the-art roadside sensors can only directly measure vehicle positions and speeds but need to derive accelerations, it makes sense to choose the time-to-collision (TTC) and post-encroachment time (PET) to measure the near-miss severity. As shown in Fig. 3-3, TTC is the elapsed time from when the first conflicting vehicle leaves the conflict zone to when the second conflicting vehicle is projected to arrive at the conflict zone without deceleration. PET is the time gap between when the first conflicting vehicle leaves the conflict zone and when the second conflicting vehicle arrives at the conflict zone, with an attempt to avoid a collision by deceleration. Therefore, PET is always equal to or greater than TTC in the same scenario.

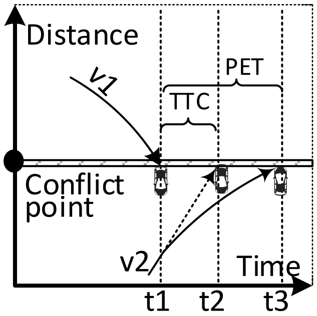


Figure 3-3 Demonstration of time to collision (TTC) and post-encroachment time (PET) (adapted from (*8*))

TTC and PET are, in essence, the safety buffer between conflicting vehicles. The aforementioned kinematic analysis in Section 3 is a special case in which both TTC and PET are set to zero. In other words, if the kinematic analysis is used to scope the conflict zone, then any captured near-misses will be of “bumper-to-bumper” types. We should hold the same interest in less severe near-misses because the highly dangerous near-misses will be similarly rare as real crashes.

Let T_{TTC} and T_{PET} denote the minimally acceptable safety clearance (i.e., minimal allowed proximity between conflict vehicles). Then conflict zones defined in Section 4 can be extended to accommodate the TTC or PET. Once two fast, conflicting vehicles appear in an expanded conflict zone at the same time, it means they are proximate enough to generate a near-miss. For the four conflict zones:

- Conflict Zones Between the RT Vehicles and VRUs: The RT vehicle is supposed to unconditionally yield to VRUs. Therefore, TTC and PET are not considered. Using smaller conflict zones will ensure focusing on the most dangerous near-misses only. Using larger zones will reflect a strict protection rule for the VRUs.

- Conflict Zones Between the Permissive LT Vehicles and Concurrent Pedestrians: Since the permissive LT vehicles are unlikely to decelerate, the conflict zone can be expanded according to the TTC threshold as shown in FIG. 4. Assuming a concurrent VRU enters the intersection at the WALK onset, t_0, and reaches the median at t_1, a near-miss can be identified if a permissive LT vehicle and a pedestrian appear in the expanded conflict zone between t_0 and t_1, meaning they are dangerously proximate. Whenever a permissive LT vehicle enters the extended conflict zone with concurrent VRUs (the blue area) at t, the vehicle's predicted arriving time to the crossing (the red area) will be predicted, t_2. If t_2 is sooner than $t_1 + TTC$, then a near-miss can be identified. Note that VRUs take much longer to clear the conflict zone. Therefore, it is almost certain that a near-miss can be identified whenever permissive LT vehicles and concurrent crossing pedestrians are identified at the same time within the extended conflict zone. The average length of the extended conflict zone can be jointly determined by the prevailing LT speed, intersection layouts, and target predicting time window (i.e., how many seconds in advance?).

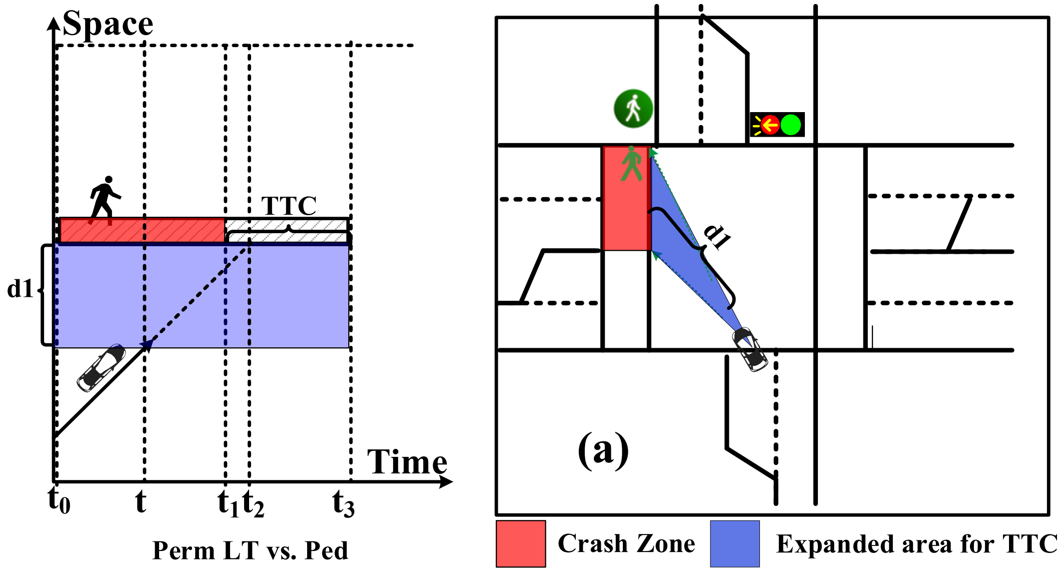


Figure 3-4 Conflict Zone extension to accommodate TTC and PET

- Conflict Zones Between Permissive LT Vehicles and Opposing Through Vehicles: If we assume that only the opposing through vehicles will decelerate, the conflict zone can be expanded according to PET from Eq. (3-4). As shown in FIG. 5, whenever a permissive LT vehicle v_1 enters the conflict zone at $t = t_0$ and leaves at $t = t_1$ (calculated according to measured instantaneous vehicle speed, vehicle length, and projected lateral path). This vehicle blocks the through lanes for $(t_1 + PET - t_0)$ seconds. If an opposing vehicle v_3 is far enough (d_2 or further) at $t_0$then it does not need to respond to the lane blockage. If an opposing vehicle is proximate, then it must decelerate to arrive at the conflict zone no earlier than t_2 to avoid a near-miss. The more proximate they are, the longer the braking action needs to be. When the opposing vehicle must take the maximal deceleration, $11.2\frac{ft}{s^2}$ recommended by AASHTO (*30*), then its instantaneous distance d_1 is the minimal distance closer than which the opposing vehicle must brake evasively. An opposing through vehicle will experience a near-miss with the permissive LT vehicle if it is closer to the conflict zone than d_1 at t.

The shortest distance d_1 can be calculated as Eq. (3-5) where v_0 is the opposing through vehicle's instantaneous speed at t_0T is the perception-reaction or P-R time.

$$d_1 = 1.47v_0T + \frac{v_0^2-(v_0-11.2*\max(0,t_2-T-t_0))^2}{30\left(\frac{11.2}{32.2}\mp G\right)} \qquad (3\text{-}5)$$

Note that the P-R time may not apply because the through vehicle driver may have noticed the likely LT maneuvers even before lane blockage (See Eq. (6)). $d_1 > d_1'$ And so, we should always adopt Eq. (3-5) to scope the conflict zones.

$$d_1' = \frac{v_0^2-(v_0-11.2*\max(0,t_2-t_0))^2}{30\left(\frac{11.2}{32.2}\mp G\right)} \qquad (3\text{-}6)$$

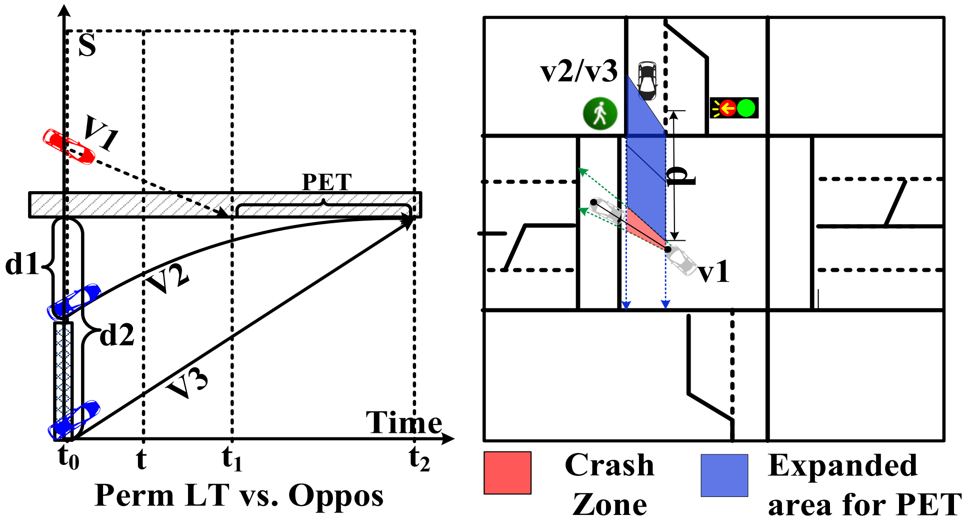


Figure 3-5 Extended conflict zones to accommodate TTC and PET

According to Eq. (5), if an opposing vehicle's P-R time T is longer than $t_2$then it may not be able to respond to the near-miss and keep moving at v_0 until it reaches the conflict zone.

- Red-Light Running: There are no associated TTC or PET for this special near-miss. So, the conflict zone is not expanded.

- Conflict Zones for Permissive LT Vehicles and U-Turn Vehicles: The near-miss will be caused by drivers' negligence, and vehicles are relatively slow. The near-miss can be solely determined by the proximity.

3.5 Evaluating High-Speed Near-Misses at Intersections

There is only one type of high-speed vehicle-to-vehicle near-misses at intersections: permissive LT vehicles vs. opposing through vehicles. The conflict zone will be scoped as illustrated in FIG. 5, covering both the crash zone and the expanded area for PET. Given the target PET and prevailing approaching speed, the shortest distance from the crash zone, d_1, can be calculated with Eq. (3-5). Other conditions for identifying a high-speed near-miss include:

- Permissive LT vehicles should enter the conflict zone before the opposing through vehicle to make the opposing through vehicle apply heavy braking to avoid a collision, even though it has the right of way during the permissive left-turn phase.
- The opposing through vehicle should be faster than the design speed in Eq. (5), which should be decided based on collected approaching speed samples (e.g., 50th percentile time-of-day speed). Slow-approaching vehicles do not necessarily need to brake hard to avoid a crash.

During a permissive left-turn phase (e.g., Flashing Yellow Arrow), whenever an opposing through vehicle enters the conflict zone at *t* $(t_0 < t < t_2), d_1$feet away from the crash zones, it will check the following conditions (illustrated in FIG. 5):

1. Is this vehicle faster (v_1) than the designed approaching speed (v_0) to be considered?
2. Is there a blocking permissive LT vehicle that enters the conflict zone earlier at t_0 and blocks the through lanes until t_2 (considering its lane-occupying time plus PET).

If the above two conditions are met and the through vehicle is willing to keep taking $a_{max} = -11.2\ \frac{ft}{s^2}$ deceleration until it arrives at the crash zone at $\boldsymbol{t_x}$, then $\boldsymbol{t_x}$ can be calculated as:

$$d_1 = v_1 T + \frac{v_1^2 - \left(v_1 - a_{max}(t_x - t - T)\right)^2}{2a_{max}} \rightarrow \boldsymbol{t_x} = \frac{\left(v_1 - \sqrt[2]{v_1^2 - 2a_{max}(d_1 - v_1 T)}\right)}{a_{max}} + t + T \qquad (3\text{-}7)$$

Discussion:

- If the P-R time T is large, then it will be possible that the subject vehicle will keep moving at v_1 until it reaches the crash zone at $\boldsymbol{t_x} = t + \frac{d_1}{v_1}$.

- If $t_x < t_2$ (e.g., the through vehicle is fast), then two vehicles will either crash or generate a near-miss. By selecting various target PET values, we can selectively capture the near-misses according to the level of severity.

In this case,

$$TTC = \max\left(0, t + \frac{d_1}{v_1} - t_1\right)) \text{ and } PET = \max(0, t_2 - t_x) \tag{3-8}$$

4 LATENCY EVALUATION OF LIDAR SENSORS IN THE FIELD

4.1 Introduction

The objective of this task is to evaluate the feasibility of using LiDAR sensors to capture multiple types of near-misses at intersections. As the near-misses occur and then disappear instantaneously, capturing such events accurately requires rather low latency. It is critical to evaluate the latency (i.e., the time lag) between when the near-misses occur and when they are captured and reported by the LiDAR sensors.

The latency of near-miss capturing is composed of two parts: (a) Data processing time, including perceiving, identifying conflict vehicles, identifying the current traffic signal status, and timestamping near-miss records; (b) Near-miss reporting time, such as uploading to servers after an intended delay. The latency in this context refers to the first part and contains three parts:

- First, all LiDAR sensors, including those installed at the test intersection, can continuously scan and perceive new vehicles appearing in the conflict zones at an interval of 0.1 seconds (10 Hz). The latency of this step will be up to 0.1 seconds (e.g., a new vehicle enters the conflict zone right after scanning, and it will be captured during the next round of scanning in 0.1 seconds). The latency at this step is caused by the inherent scanning/sampling frequency of LiDAR sensors and cannot be avoided.
- Second, the list of perceived vehicles in a conflict zone will be sent over to the near-miss capturing algorithm. The algorithm will determine the turning movement of each perceived vehicle and identify conflicting vehicles. If the conflicting vehicles meet the required conditions for near-misses, then the near-misses will be identified and timestamped according to when the perceived vehicles first enter the conflict zones. The latency at this step can be minimized through algorithm optimization.
- Third, for the sake of the evaluation, the captured near-misses need to be displayed on the computer screen. The algorithm's host computer (the LiDAR processor) and the monitoring laptop are connected via the Secure Shell (SSH), and therefore, transmitting near-miss events from the LiDAR processor to the laptop's screen will generate additional latencies. Note that this step exists only for the latency evaluation, and it will be skipped

when the LiDAR solution is fully deployed. Therefore, this experiment may slightly overestimate the latency of near-miss identification.

The challenge in this task is how to identify the “groundtruth” time when a near-miss occurs. A natural approach is to observe and record the groundtruth near-miss occurrences via a pro-quality camera in the field. After investigating a few available options, the researchers decided to use one researcher’s iPhone-13 Max-Pro smartphone to serve this purpose. The Iphone-13 Max-Pro smartphone was wired to a laptop and coupled via a software tool called “Camo”[2]. This configuration can retrofit the smartphone into a high-resolution (4K), high-speed (60 fps) computer camera. Through indoor experiments, the researchers did not observe any noticeable latencies between real activities and what was displayed on the laptop screen.

Another challenge is how to synchronize three clocks to calculate the accurate latency: the smartphone clock, the laptop clock, and the LiDAR processor’s clock. To solve this problem, the researchers installed a millisecond-level clock in the monitoring laptop to display a uniform reference time clock on the screen.

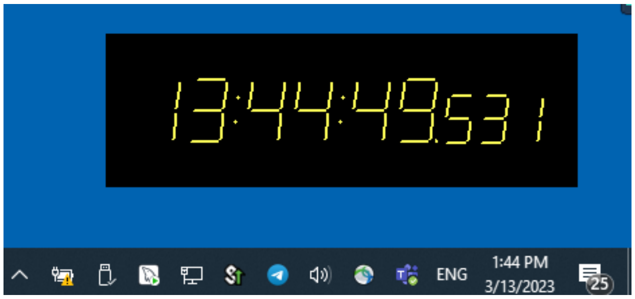


Figure 4-1 The millisecond clock as the time reference

4.2 Experiment Design

The regular near-miss identification by the LiDAR sensing solution includes two steps:

(I) Perceiving vehicles in conflict zones with the LiDAR perception software and then identifying their turning movements according to their origins with the application algorithms.

[2] https://apps.apple.com/us/app/camo-webcam-for-mac-and-pc/id1514199064

(II) Identifying conflicting vehicles and near-misses among conflicting vehicles with the application algorithms.

Step One causes most of the latency for the entire near-miss identification, whereas the computing latency at Step Two (mostly composed of a few "IF-THEN" rules) can be safely ignored. The experiment focused on capturing the latency in Step One.

The researchers customized the application algorithm to duplicate the computing process of Step One and report the computing time (latency) to identify vehicles with turning information. As shown in Fig. 4-2, Zone D at INT 7122 is selected to evaluate the latency. Whenever a vehicle enters Zone D, it is either speeding up or maintaining a relatively high speed (except for the NBR vehicles). Therefore, most vehicles' maneuvers in Zone D are like those in conflict zones.
Whenever a new vehicle in Zone D is identified, the special application algorithm will first check three origin zones (Zone A, B, and C) to determine this vehicle's movement. For example, if a vehicle holding the same temporary ID enters the intersection from Zone A and leaves the intersection through Zone D, then this vehicle's movement is *South-Bound-Left*. The time difference between when a vehicle enters Zone D and when this vehicle is identified and reported by the algorithm is the latency of Step One, approximately equal to the total latency of near-miss identification.

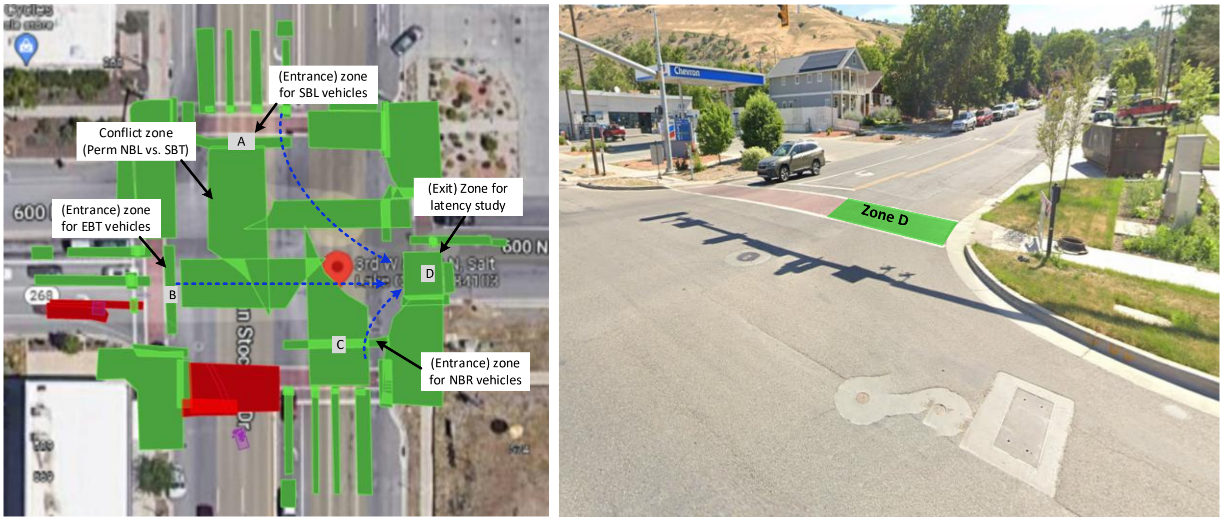


Figure 4-2 Zone Layouts at INT 7122 in Salt Lake City, Utah

4.3 Data Collection in the Field

The raw data collection system was configured as in Fig. 4-3. The pro-quality camera (smartphone) is aimed at Zone D and sends high-speed video(s) (60 frames per second) back to the monitoring laptop. In the meantime, the researcher remotely logged in to the live LiDAR processor in the traffic signal cabinet and ran the customized application algorithm to display any newly captured vehicles in Zone D on a remote console in the monitoring laptop.

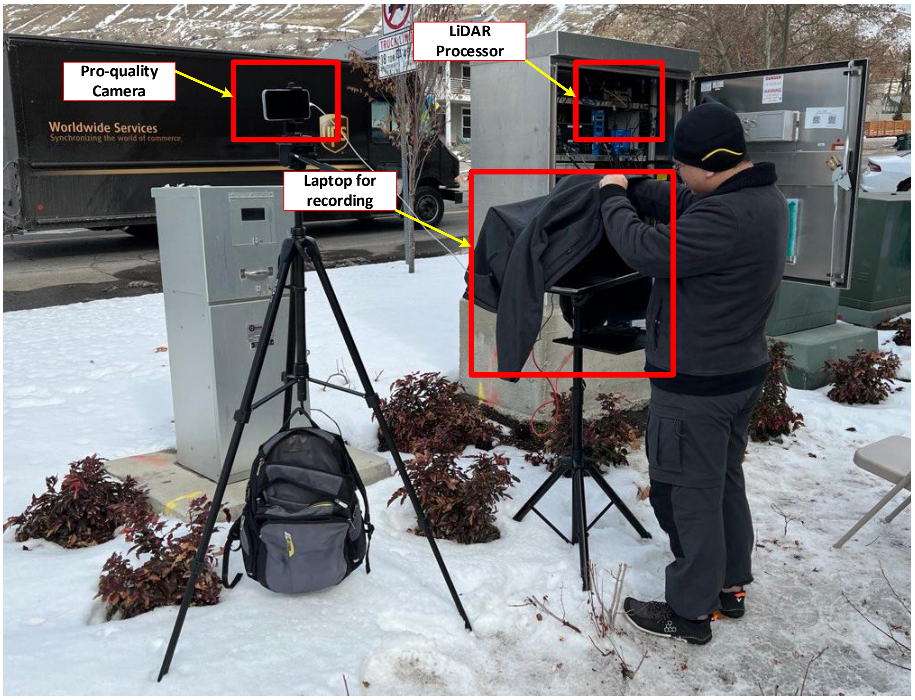
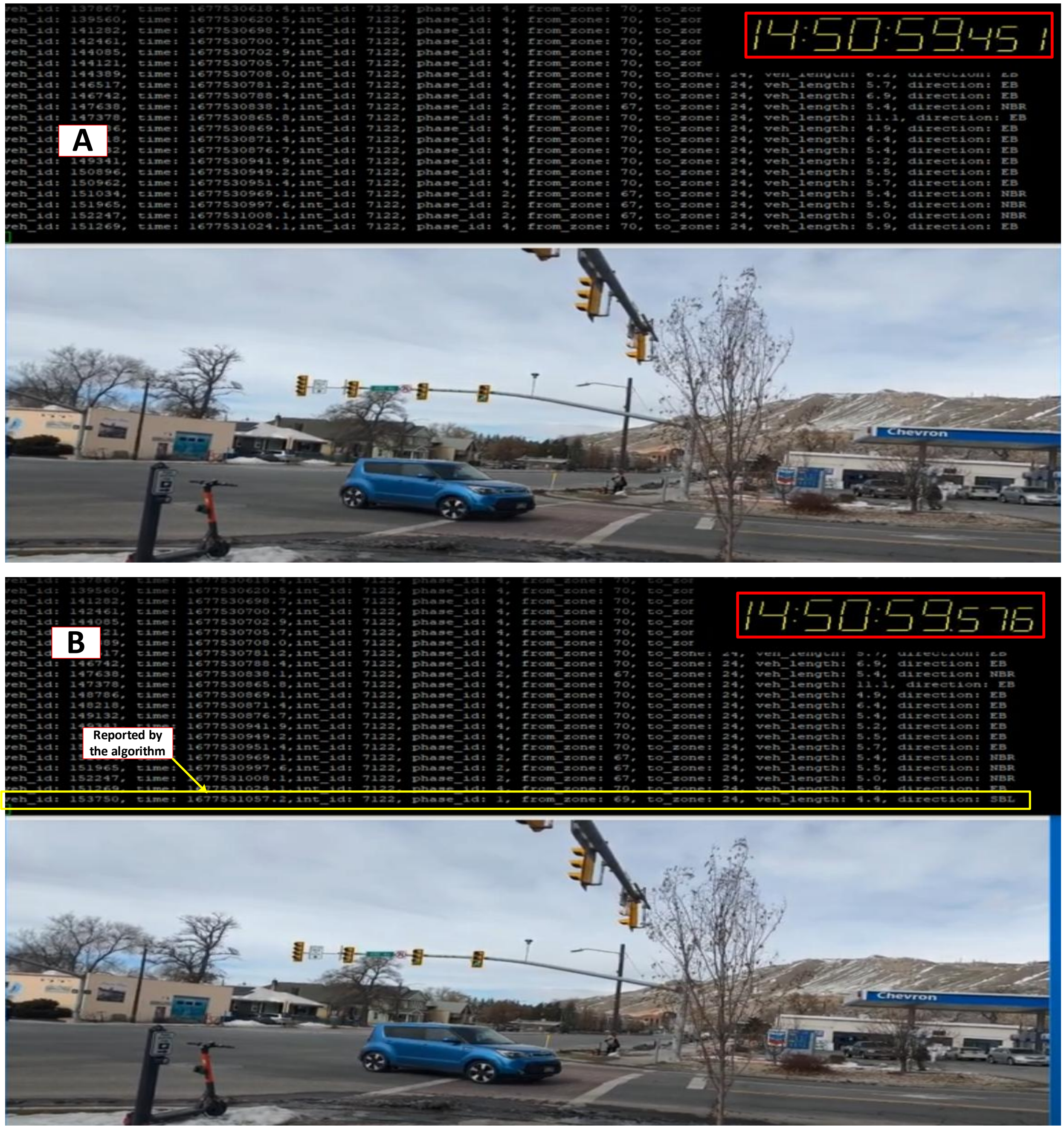


Figure 4-3 Configuration of raw data collection system

The camera's live image and the LiDAR console were aligned on the screen of the monitoring laptop in conjunction with the referenced millisecond clock. During the data collection process, the researchers recorded the laptop screen with three programs aligned. The recorded video clip is the raw data for further latency evaluation in this context. The computing latency was calculated for each identified vehicle entering Zone D. As shown in Fig.4-4, whenever a vehicle's front bumper was observed to enter Zone D, its groundtruth time entering Zone D was stamped according to the millisecond clock at the top right corner (Fig. 4-4A); if this vehicle was reported by the application algorithm later (Fig. 4-4B), then this event was timestamped again. The latency for this vehicle was calculated as the difference between two timestamps.



```
veh_id: 137867, time: 1677530618.4,int_id: 7122, phase_id: 4, from_zone: 70, to_zor
veh_id: 139560, time: 1677530620.5,int_id: 7122, phase_id: 4, from_zone: 70, to_zor
veh_id: 141282, time: 1677530698.7,int_id: 7122, phase_id: 4, from_zone: 70, to_zor
veh_id: 142461, time: 1677530700.7,int_id: 7122, phase_id: 4, from_zone: 70, to_zor
veh_id: 144085, time: 1677530702.9,int_id: 7122, phase_id: 4, from_zone: 70, to_zor
veh_id: 144121, time: 1677530705.7,int_id: 7122, phase_id: 4, from_zone: 70, to_zor
veh_id: 144389, time: 1677530708.0,int_id: 7122, phase_id: 4, from_zone: 70, to_zone: 24, veh_length: 6.2, direction: EB
veh_id: 146517, time: 1677530781.2,int_id: 7122, phase_id: 4, from_zone: 70, to_zone: 24, veh_length: 5.7, direction: EB
veh_id: 146742, time: 1677530788.4,int_id: 7122, phase_id: 4, from_zone: 70, to_zone: 24, veh_length: 6.9, direction: EB
veh_id: 147638, time: 1677530838.1,int_id: 7122, phase_id: 2, from_zone: 67, to_zone: 24, veh_length: 5.4, direction: NBR
veh_id: 147378, time: 1677530865.8,int_id: 7122, phase_id: 4, from_zone: 70, to_zone: 24, veh_length: 11.1, direction: EB
veh_id:   6, time: 1677530869.1,int_id: 7122, phase_id: 4, from_zone: 70, to_zone: 24, veh_length: 4.9, direction: EB
veh_id:   8, time: 1677530871.4,int_id: 7122, phase_id: 4, from_zone: 70, to_zone: 24, veh_length: 6.4, direction: EB
veh_id:   2, time: 1677530876.7,int_id: 7122, phase_id: 4, from_zone: 70, to_zone: 24, veh_length: 5.4, direction: EB
veh_id: 149341, time: 1677530941.9,int_id: 7122, phase_id: 4, from_zone: 70, to_zone: 24, veh_length: 5.2, direction: EB
veh_id: 150896, time: 1677530949.2,int_id: 7122, phase_id: 4, from_zone: 70, to_zone: 24, veh_length: 5.5, direction: EB
veh_id: 150962, time: 1677530951.4,int_id: 7122, phase_id: 4, from_zone: 70, to_zone: 24, veh_length: 5.7, direction: EB
veh_id: 151034, time: 1677530969.1,int_id: 7122, phase_id: 2, from_zone: 67, to_zone: 24, veh_length: 5.4, direction: NBR
veh_id: 151965, time: 1677530997.6,int_id: 7122, phase_id: 2, from_zone: 67, to_zone: 24, veh_length: 5.5, direction: NBR
veh_id: 152247, time: 1677531008.1,int_id: 7122, phase_id: 2, from_zone: 67, to_zone: 24, veh_length: 5.0, direction: NBR
veh_id: 151269, time: 1677531024.1,int_id: 7122, phase_id: 4, from_zone: 70, to_zone: 24, veh_length: 5.9, direction: EB
```

```
veh_id: 137867, time: 1677530618.4,int_id: 7122, phase_id: 4, from_zone: 70, to_zor
veh_id: 139560, time: 1677530620.5,int_id: 7122, phase_id: 4, from_zone: 70, to_zor
veh_id: 141282, time: 1677530698.7,int_id: 7122, phase_id: 4, from_zone: 70, to_zor
veh_id: 142461, time: 1677530700.7,int_id: 7122, phase_id: 4, from_zone: 70, to_zor
veh_id: 144085, time: 1677530702.9,int_id: 7122, phase_id: 4, from_zone: 70, to_zor
veh_id   21, time: 1677530705.7,int_id: 7122, phase_id: 4, from_zone: 70, to_zor
veh_id   89, time: 1677530708.0,int_id: 7122, phase_id: 4, from_zone: 70, to_zor
veh_id   17, time: 1677530781.2,int_id: 7122, phase_id: 4, from_zone: 70, to_zone: 24, veh_length: 5.7, direction: EB
veh_id: 146742, time: 1677530788.4,int_id: 7122, phase_id: 4, from_zone: 70, to_zone: 24, veh_length: 6.9, direction: EB
veh_id: 147638, time: 1677530838.1,int_id: 7122, phase_id: 2, from_zone: 67, to_zone: 24, veh_length: 5.4, direction: NBR
veh_id: 147378, time: 1677530865.8,int_id: 7122, phase_id: 4, from_zone: 70, to_zone: 24, veh_length: 11.1, direction: EB
veh_id: 148786, time: 1677530869.1,int_id: 7122, phase_id: 4, from_zone: 70, to_zone: 24, veh_length: 4.9, direction: EB
veh_id: 148218, time: 1677530871.4,int_id: 7122, phase_id: 4, from_zone: 70, to_zone: 24, veh_length: 6.4, direction: EB
veh_id: 149132, time: 1677530876.7,int_id: 7122, phase_id: 4, from_zone: 70, to_zone: 24, veh_length: 5.4, direction: EB
veh_id: 1      1677530941.9,int_id: 7122, phase_id: 4, from_zone: 70, to_zone: 24, veh_length: 5.2, direction: EB
veh_id: 1      1677530949.2,int_id: 7122, phase_id: 4, from_zone: 70, to_zone: 24, veh_length: 5.5, direction: EB
veh_id: 1      1677530951.4,int_id: 7122, phase_id: 4, from_zone: 70, to_zone: 24, veh_length: 5.7, direction: EB
veh_id: 1      1677530969.1,int_id: 7122, phase_id: 2, from_zone: 67, to_zone: 24, veh_length: 5.4, direction: NBR
veh_id: 151965, time: 1677530997.6,int_id: 7122, phase_id: 2, from_zone: 67, to_zone: 24, veh_length: 5.5, direction: NBR
veh_id: 152247, time: 1677531008.1,int_id: 7122, phase_id: 2, from_zone: 67, to_zone: 24, veh_length: 5.0, direction: NBR
veh_id: 151269, time: 1677531024.1,int_id: 7122, phase_id: 4, from_zone: 70, to_zone: 24, veh_length: 5.9, direction: EB
veh_id: 153750, time: 1677531057.2,int_id: 7122, phase_id: 1, from_zone: 69, to_zone: 24, veh_length: 4.4, direction: SBL
```

Figure 4-4 Demonstration of computing latency calculation for zone-based near-miss identification

4.4 Results Summary and Analysis Performance of Identifying Vehicles

The raw data were collected (recorded) in the afternoon of Feb. 27, 2023, for an hour. There were a total of 145 vehicles from three directions (SBL, NBR, and EBT) that were observed, and

144 were reported by the application algorithm. The identifying accuracy was **99.3%.** Table 4-1 summarizes the performance of vehicle capturing.

Table 4-1 Performance of Vehicle Capturing with the Application Algorithm.

	Actual Observation			The algorithm reports		
	EB	**NBR**	**SBL**	**EB**	**NBR**	**SBL**
2:14-2:29	35	3	1	34	3	1
2:29-2:44	29	7	1	29	7	1
2:44-2:59	26	9	1	28	9	1
2:59-3:14	27	3	3	27	3	3

Latency Analysis

Based on frame-by-frame observation from the recorded raw data, the average latency of vehicle identification is 0.126 seconds among 144 identified vehicles. The latency distribution is shown in Fig. 4-5. From Fig. 4-5, more than 80% of latencies were lower than 0.15 seconds. For those identifications with high latencies, they were mostly the slow NBR vehicles. It should be pointed out that the actual latency may be lower than the calculated values because it took extra time to send information from the LiDAR processor to the remote console screen in the laptop.

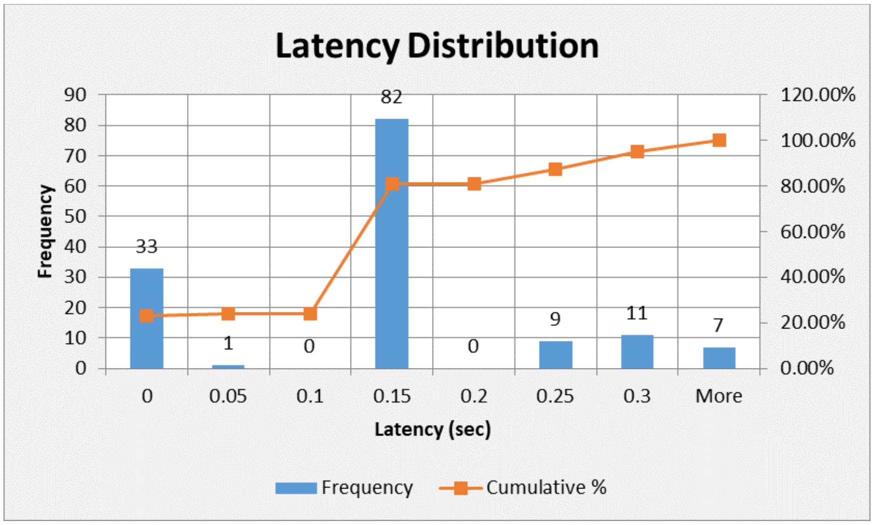


Figure 4-5 Algorithm's latency distribution to identify vehicles

4.5 Remarks and Discussion

Although the researchers consider the above experiment valid and effective, it is not perfect due to the limitations of the equipment. For example, latencies exist when the live images are processed and transmitted from the camera (smartphone) to the laptop, making the perceived "groundtruth" a little behind. Transmitting the captured information from the LiDAR processor to the console screen of the laptop may have taken extra time, which may be relatively nontrivial. The vehicles' arrival timestamps at Zone D were judged not by advanced video analytic tools but arbitrarily by the researchers. Thus, the criteria may vary from person to person and from time to time. Nonetheless, the latencies of LiDAR sensors were low enough to accurately identify near-misses, including red-light running at intersections.

5 VALIDATIONS OF LIDAR-REPORTED NEAR-MISSES WITH THE PTZ CAMERA

5.1 Introduction

The overarching goal of this chapter is to collect and validate the LiDAR-reported near-misses based on the previously described near-miss identification method using recorded video. To avoid bias, multiple types of unfiltered near-misses identified by LiDAR were evaluated. Those reported near-misses were manually evaluated by researchers with the corresponding video clips. In practice, the accuracy and reliability of LiDAR-based near-miss identification are critical to inform the potential safety issues at intersections.

5.2 Workflow

For each type of near-miss, the validating process is shown as in Figure 5-1. First, the conflict zone is designed according to the methods described in Chapter 3. Second, various types of LiDAR-reported near-misses were collected for a few days to estimate the frequency and spatiotemporal distribution of near-misses. This information was used to determine how many hours of synchronous traffic videos should be recorded for validation. At last, the LiDAR-reported near-misses were verified from the synchronous traffic videos according to their synchronous timestamps.

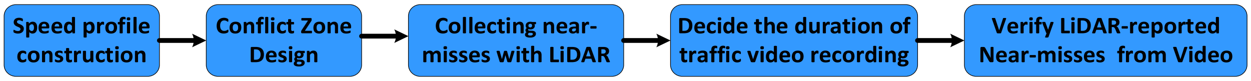

Speed profile construction
Conflict Zone Design
Collecting near-misses with LiDAR
Decide the duration of traffic video recording
Verify LiDAR-reported Near-misses from Video

Figure 5-1 Workflow of capturing and validation of near-misses via the LiDAR solution

5.3 Conflict Zone Layout

Without loss of generality, five conflict zones were designed as illustrated in Fig. 5-2. Zone 1 is the conflict zone between permissive NBL and SBT traffic during the flashing yellow arrows (FYA); Zone 2 is the conflict zone between permissive EBR vehicles and concurrent pedestrians; Zone 3 is the conflict zone to capture NBT red-light runners; Zone 4 is the conflict zone between permissive NBL and concurrent crossing pedestrians; and Zone 5 is the conflict zone between EBR vehicles and NB U-turn vehicles. The exact scope of each conflict zone will be calculated based on the method described in Chapter 3. Whenever two conflicting vehicles or pedestrians appear in the conflict zone at the same time and the speed thresholds (for vehicle-to-vehicle conflicts) are met, a near-miss of the corresponding type will be reported.

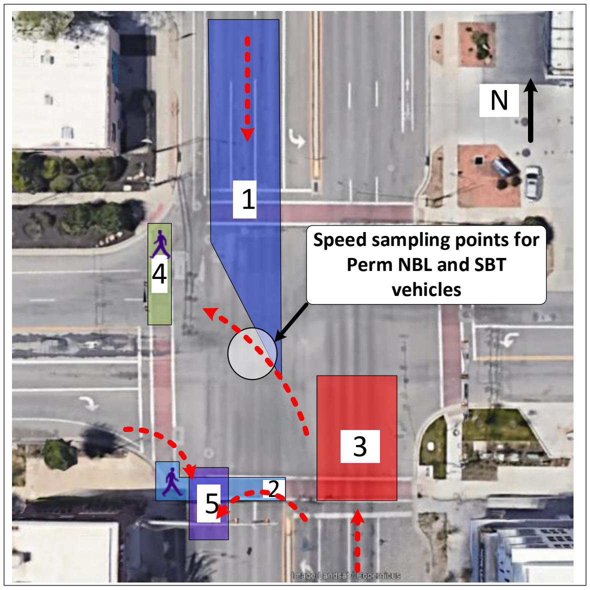


Figure 5-2 Illustrative layouts of five types of conflict zones at intersections

5.4 Vehicle Speed Profiles Within Intersections

Vehicles' prevailing speeds within the intersections are critical to determining the conflict zone layouts, especially for vehicle-to-vehicle conflicts. To address this issue, the project team developed a special LiDAR program to collect approaching vehicles' speeds within intersections. Special attention was paid to the prevailing speeds of permissive NBL vehicles and SBT vehicles as their near-misses mostly occur at high speeds due to misperceiving the gaps (See Zone 1 in Fig. 3-2). For both permissive NBL vehicles and SBT vehicles, 300 samples of speed were collected, respectively, during the off-peak hours when vehicles move faster and are more prone to crashes. Accordingly, the researchers constructed speed profiles for permissive NBL vehicles and SBT vehicles in Fig. 5-3.

The researchers also noticed that four permissive left-turn vehicles (NBL, SBL, EBL, and WBL) and four through vehicles revealed different speed profiles within the same intersection.

Multiple reasons can cause such differences, including but not limited to variable speed limits, the number of opposing traffic lanes, conflict pedestrian volumes, and/or surrounding street layouts. Therefore, it is recommended that constructing comprehensive speed profiles within intersections should be done first to capture near-misses properly.

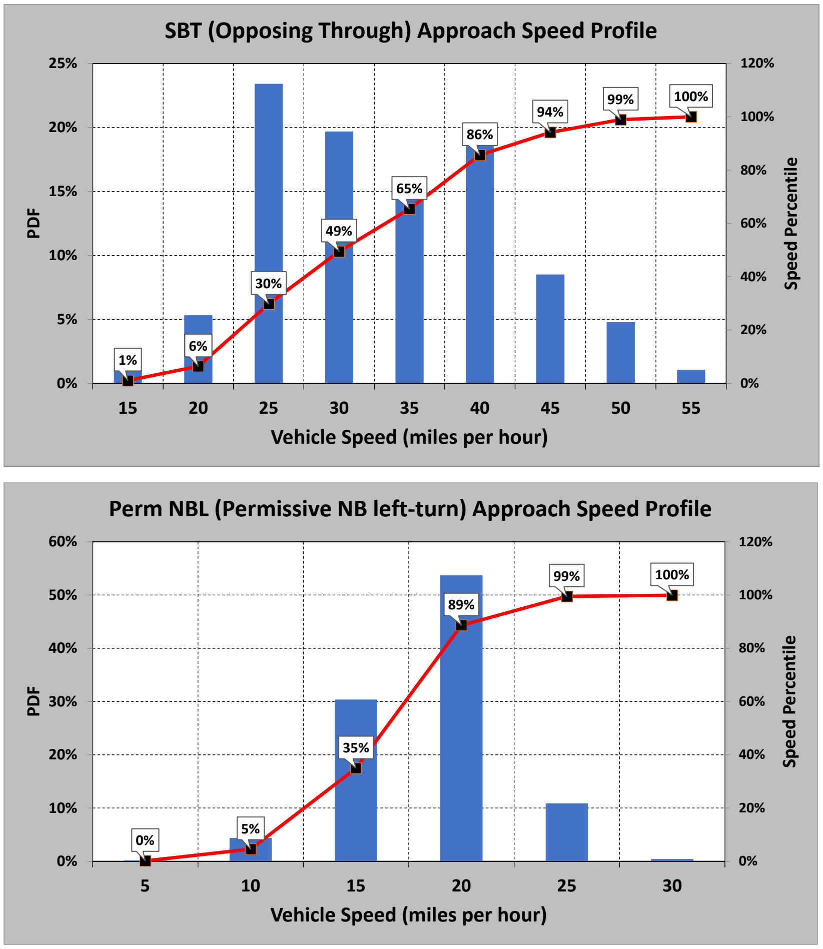


Figure 5-3 Speed profiles of conflicting permissive NBL and SBT vehicles

5.5 Scope of Conflict Zones at Intersection 7122

5.5.1 Conflict Zone Design for Near-Misses Between Permissive NBL and SBT Vehicles Due to Drivers' Misperception

At Intersection 7122, the permissive NLT vehicle's dwelling time in the conflict zone will be: $t = \frac{(\delta(L+N\times w))}{v_O} = \frac{1.2*(14+3*12)}{13*1.47} \approx 3.1$ s. For this calculation, the standard vehicle length is set to 14 feet (4.3 meters), the lane width is 12 feet (3.7 meters), N is equal to 3 (lanes), δ is empirically set to 1.2 to accommodate the vehicle's curvy maneuvers; v_o Is set as 10 MPH (4.47 meters/s), the 5 percentile speed of permissive NLT vehicles, representing the relatively slow vehicles that may dwell in the conflict zone longer. During this 3.1 s, a permissive NLT vehicle may block approaching opposing through vehicles.

For the opposing SBT vehicles, the scope of the expanded conflict zone depends on the adopted approaching speed. The higher the approaching speed we use, the larger the conflict zone will be. While capturing high-speed near-misses means preventing severe crashes, the researchers found two challenges in adopting high prevailing speeds (e.g., 95th percentile) to design the conflict zones.

- If a southbound through vehicle enters the conflict zone at a speed below the predefined threshold, the algorithm assumes the vehicle will maintain its speed or decelerate slightly when approaching the intersection. Under this assumption, the vehicle is expected to reach the stop bar after the northbound left-turning vehicle has left the conflict zone, and thus no near-miss is reported. However, in real-world scenarios, the southbound vehicle may instead accelerate while entering the zone due to the driver's misjudgment, then perceive an imminent collision with the permissive left-turn vehicle. This can trigger sudden evasive braking to avoid a crash, potentially resulting in a near-miss if not a real collision. The likelihood of such incidents increases when the conflict zone is large, providing more spatial and temporal allowance for changes in vehicle behavior and decision-making.
- The conflict zone may become excessively long when the speed of approaching vehicles is high. For instance, if an additional 2 seconds is allocated as the post-encroachment time (PET) for the permissive northbound left-turn vehicle, then the system must begin evaluating the southbound through vehicle's speed at least 5 seconds upstream from the stop bar. At an approaching speed of 45 MPH, this corresponds to approximately 330 feet. However, vehicles at this distance are often not fully aligned with their final travel paths (i.e., not yet channelized), increasing the likelihood of inaccurate trajectory predictions and resulting in a higher rate of false near-miss alarms.

After identifying these challenges, the researchers recognized that the popular concepts of traffic conflicts like time-to-collision or post-encroachment-time originally root from freeway operations. Thus, they must be modified in the context of intersections. To address this issue, the far edge of the conflict zone should be drawn closer to the stop bar to capture only the final maneuvers of the southbound vehicle before they arrive. This ensures the algorithm focuses on the most relevant interactions with permissive left-turn vehicles when both vehicles enter the conflict zone simultaneously.

According to this rationale, the researchers selected the 5th percentile speed of SB opposing vehicles, 19 MPH (28 feet per second), to design the conflict zone. Given the low-speed threshold, more than 95% of approaching vehicles will be evaluated whenever they enter the conflict zone. According to the AASHTO mandate, braking-involved perception-reaction time, t_0, is 2.5 s. If we set $t_{PET} = 1$, the far edge of the conflict zone can be calculated following Eq. (3-5) as:

$$d_1 = 1.47 * 19 * 2.5 + \frac{28^2 - (28 - 11.2 * \max(0,\ \ 1 + 3.1 - 2.5)^2}{30\left(\frac{11.2}{32.2}\right)} = 134\,ft$$

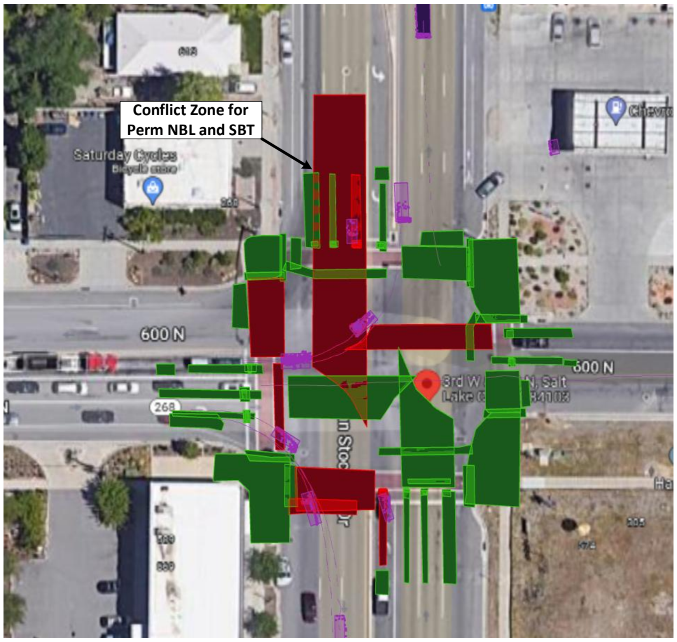


Figure 5-4 Scope of conflict zone for the permissive NBL and SBT vehicles

5.5.2 Other Conflict Zones Due to the Driver's Inattention

All other types of conflicts are caused by drivers' negligence, and so they can occur at any speed. Therefore, conflict zones of other near-misses focus on the areas where conflicting vehicles and pedestrians appear at the same time, regardless of their speeds, specifically, for the conflicts between permissive RT vehicles and concurrent pedestrians. The conflict zone is along the pedestrian crossing with slight expansion (Zone 2 in Fig. 5-2); the conflict zone for red-light running is right after the stop bar to check if the through vehicles appear over there after the yellow expires (Zone 3); the conflict zone between permissive LT vehicles and concurrent crossing pedestrians is composed of pedestrian crossing plus the available space for vehicles to turn left (Zone 4); and the conflict zone between permissive RT vehicles and U-turn vehicles is on the outermost lane right after the intersections and a RT vehicle and a U-turn vehicle will be too close if they appear in Zone 5 at the same time.

5.6 Collecting Near-Misses Using the LiDAR Sensors in the Field

The researchers deployed the near-miss identification algorithm to capture all five near-misses. Without loss of generality, the research team attempted to collect near-misses between permissive NBL vehicles and SBT vehicles (Type I); between permissive EBR vehicles vs. concurrent pedestrians (Type II); NB red-light running vehicles (Type III); permissive NBT vehicles vs. concurrent pedestrians (Type IV); and permissive EBR vehicles vs. NB U-turn vehicles. In other words, this experiment was associated with the NB approach of Intersection 7122, while the other three approaches can be configured similarly.

Two highlights of this experiment are: it will report multiple types of near-misses in real time; it can reveal under what traffic signal operations (instantly retrieved from the traffic signal controller) a near-miss occurred. For instance, if a Type-I near-miss was caught, then this near-miss occurred during the permissive NB left-turn operations (i.e., NB FYA).

Each captured near-miss includes the following information:

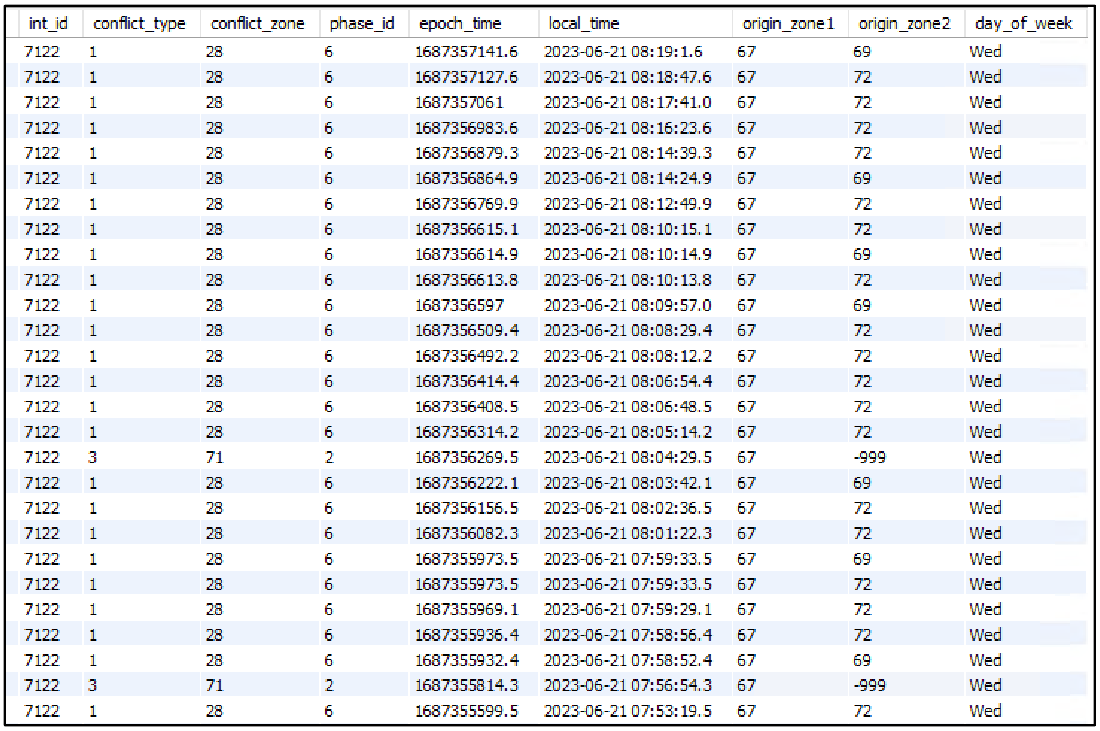
- Intersection ID: UDOT’s internal intersection IDs. In this case, it is 7122.
- Conflict type: There are a total of 5 types of near-misses defined in the algorithm.
- Conflict Zone: the zone ID defined in the LiDAR user interface (refer to Figure 5-4).
- Phase ID: the current green phase when a near-miss occurred.
- Epoch time: the near-miss’ timestamp in the form of Epoch time (total seconds since midnight Jan-01-1970).
- Local time: the near-miss’ timestamp in the form of ASCII time (human-readable);
- Origin Zone 1: The origin of the first vehicle in conflict. The zone ID was defined in the LiDAR perception user interface (refer to Figure 5-4).
- Origin Zone 2: the origin of the second vehicle in conflict. The zone ID was defined in the LiDAR perception user interface (refer to Figure 3-4). “-999” means “not applicable” for the red-

light running (the Type-III near-miss) as the red-light runners do not conflict with other vehicles but miss the yellow end.

- Day-of-Week: the day of the week when the near-miss occurred.

The captured near-misses were stored in the LiDAR process in the field temporarily and then were uploaded to the central MySQL server hosted in a UDOT computer at 1 AM every day. The results could also be reviewed from the dashboard program, UTA-in-Motion. In a nutshell, the near-misses between permissive NBL vehicles and SBT vehicles (Type I) and NB red-light running (Type III) contributed to most near-misses, where other types were relatively rare. After several rounds of fine-tuning to remove the "false alarms," it was estimated that the total near-misses of all types associated with the NB approach would range from 400 to 600 per day. The researchers estimated the total near-misses could be around 2,000 per intersection per day. From Fig. 5-6, the near-misses appear intense and continuous except for the period from midnight to the early morning (before the AM peak hours start).

Note that the above near-misses and conflict zones only reflect the researchers' engineering judgment of conflict zones for near-miss identification. The definition of near-misses is by nature subjective, and so the frequency can significantly change if users perceive the conflict and design the conflict zone differently. In practice, the number of captured near-misses can change depending on the scope and skewness of conflict zones to reflect local regulations and agencies' opinions on near-misses. For instance, at intersections with high pedestrian crashes, it is recommended to draw big conflict zones between vehicles and pedestrians to increase the sensitivity of near-miss identification. It is also possible to draw skewed conflict zones to reflect customized definitions of near-misses. If the conflict zones are drawn very small, then only the most severe near-misses (e.g., bumper-to-bumper) will be captured.



int_id	conflict_type	conflict_zone	phase_id	epoch_time	local_time	origin_zone1	origin_zone2	day_of_week
7122	1	28	6	1687357141.6	2023-06-21 08:19:1.6	67	69	Wed
7122	1	28	6	1687357127.6	2023-06-21 08:18:47.6	67	72	Wed
7122	1	28	6	1687357061	2023-06-21 08:17:41.0	67	72	Wed
7122	1	28	6	1687356983.6	2023-06-21 08:16:23.6	67	72	Wed
7122	1	28	6	1687356879.3	2023-06-21 08:14:39.3	67	72	Wed
7122	1	28	6	1687356864.9	2023-06-21 08:14:24.9	67	69	Wed
7122	1	28	6	1687356769.9	2023-06-21 08:12:49.9	67	72	Wed
7122	1	28	6	1687356615.1	2023-06-21 08:10:15.1	67	72	Wed
7122	1	28	6	1687356614.9	2023-06-21 08:10:14.9	67	69	Wed
7122	1	28	6	1687356613.8	2023-06-21 08:10:13.8	67	72	Wed
7122	1	28	6	1687356597	2023-06-21 08:09:57.0	67	69	Wed
7122	1	28	6	1687356509.4	2023-06-21 08:08:29.4	67	72	Wed
7122	1	28	6	1687356492.2	2023-06-21 08:08:12.2	67	72	Wed
7122	1	28	6	1687356414.4	2023-06-21 08:06:54.4	67	72	Wed
7122	1	28	6	1687356408.5	2023-06-21 08:06:48.5	67	72	Wed
7122	1	28	6	1687356314.2	2023-06-21 08:05:14.2	67	72	Wed
7122	3	71	2	1687356269.5	2023-06-21 08:04:29.5	67	-999	Wed
7122	1	28	6	1687356222.1	2023-06-21 08:03:42.1	67	69	Wed
7122	1	28	6	1687356156.5	2023-06-21 08:02:36.5	67	72	Wed
7122	1	28	6	1687356082.3	2023-06-21 08:01:22.3	67	72	Wed
7122	1	28	6	1687355973.5	2023-06-21 07:59:33.5	67	69	Wed
7122	1	28	6	1687355973.5	2023-06-21 07:59:33.5	67	72	Wed
7122	1	28	6	1687355969.1	2023-06-21 07:59:29.1	67	72	Wed
7122	1	28	6	1687355936.4	2023-06-21 07:58:56.4	67	72	Wed
7122	1	28	6	1687355932.4	2023-06-21 07:58:52.4	67	69	Wed
7122	3	71	2	1687355814.3	2023-06-21 07:56:54.3	67	-999	Wed
7122	1	28	6	1687355599.5	2023-06-21 07:53:19.5	67	72	Wed

Figure 5-5 A snapshot of archived near-misses in the MySQL database

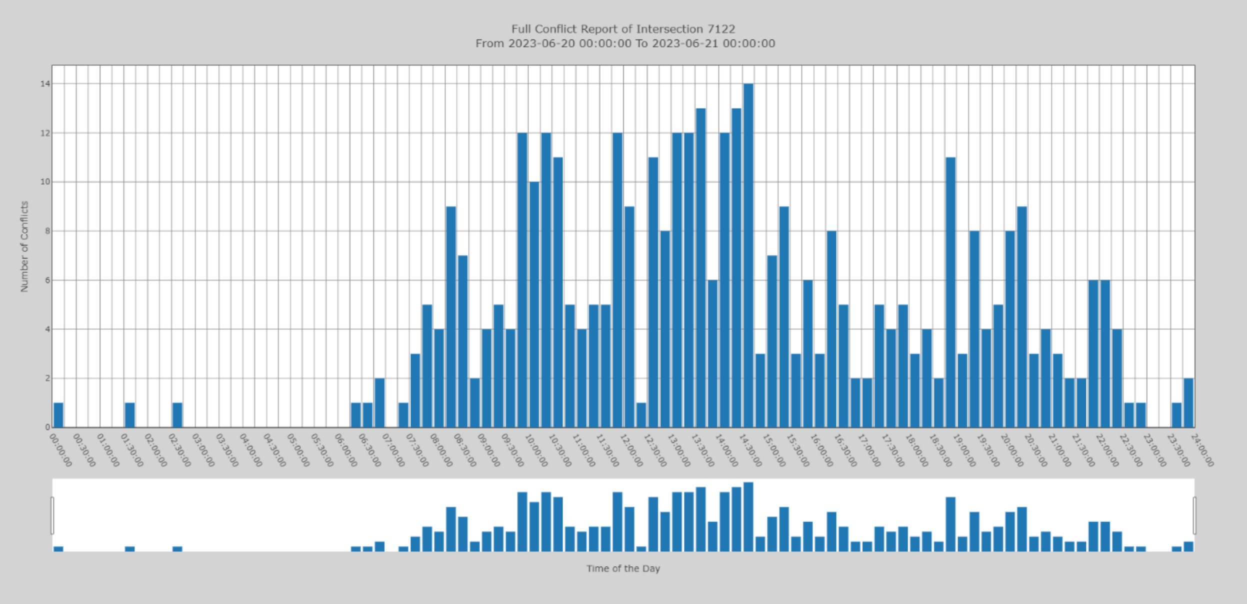


Figure 5-6 The time-of-day summary of all near-misses (5 types) associated with the NB approach

5.7 Near-Miss Capturing and Validation with Videos

It is critical to validate the above results to offer meaningful recommendations for traffic safety improvement. Therefore, the researchers conducted extensive experiments to evaluate the accuracy of the near-miss capturing algorithm.

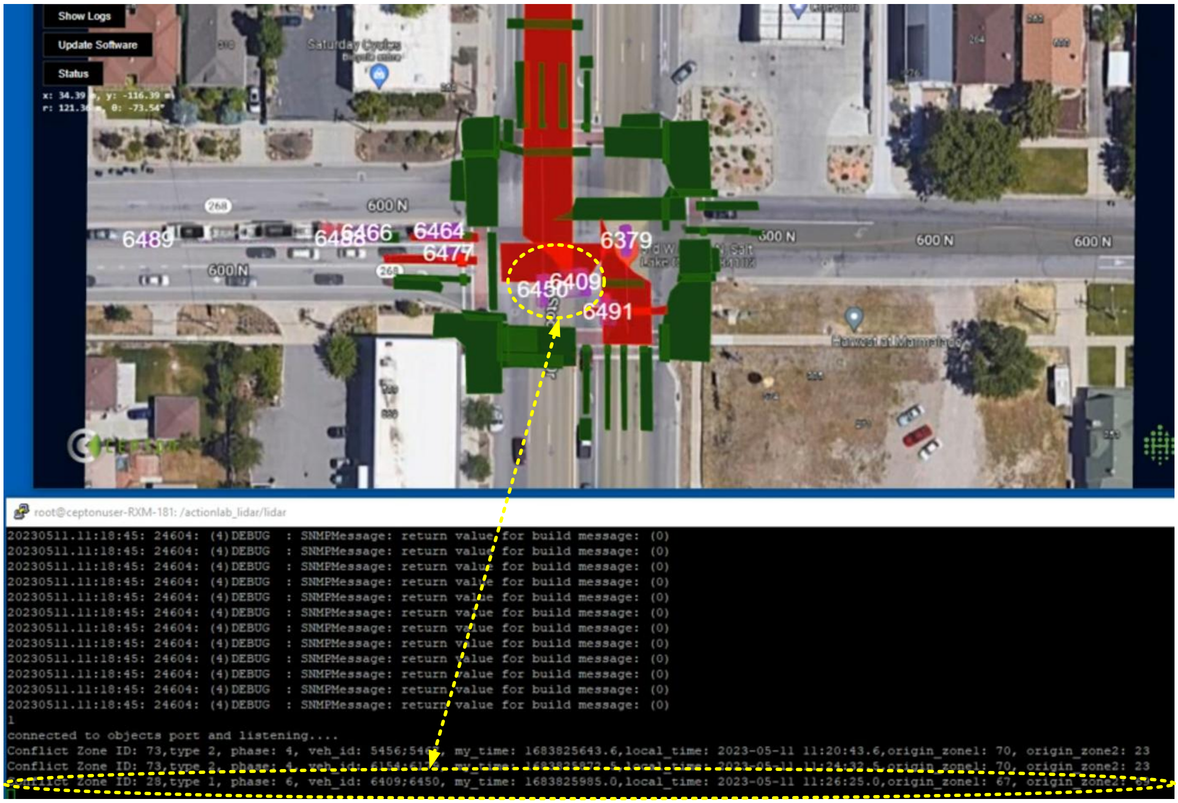
5.7.1 The Preliminary Experiment

Preliminary validation was first conducted on the LiDAR perception software user interface (UI). Most LiDAR manufacturers or independent perception software vendors provide a web interface for users to set up their applications. Using a special version of the developed algorithm software, the researchers could observe and verify the tracked vehicle maneuvers, their occupancies of zones, and the reported near-misses by the algorithm.

The purpose of this preliminary experiment was to determine the experiment plan for validation with the installed Pan-Tilt-Zoom cameras at Intersection 7122, such as how long the researchers need to record the video to capture sufficient near-misses. Out of this preliminary experiment, it was found that it is more reliable in capturing the near-misses among moving vehicles (e.g., permissive NBL vehicles vs. SBT vehicles) than capturing the near-misses between vehicles and pedestrians, especially the right-turn vehicles and the corresponding concurrent pedestrians. The reason for this pattern could be that relatively slow right-turn vehicles, while making turning maneuvers, are tracked at the far ends of all LiDAR sensors, and therefore, the tracking reliability slightly deteriorates. In addition, tracking pedestrians at street corners is more challenging than tracking vehicles due to the small size and random behaviors of pedestrians. The researchers think that this inferior performance is rooted in the inefficiency of LiDAR sensors' point cloud and the perception algorithm. Solving the perception problem is beyond the scope of this research. Nonetheless, this problem of the perception software is anticipated to be effectively resolved soon, based on the recent demonstration of commercial LiDAR hardware/software development for the smart infrastructure[3]. There are two types of computing technologies for LiDAR perception: CPU-based and GPU-based. CPU-based perceptions consume fewer computing resources while it is often less accurate to perceive closely spaced objects such as

[3] This finding was formed in 2023. As of 2025, this problem has been solved with satisfaction among most commercial perception software solutions.

queueing vehicles or crowded pedestrians. By contrast, the GPU-based solutions consume much more computing resources, but they perform better in detecting waiting vehicles and crowded pedestrians.



```
20230511.11:18:45: 24604: (4)DEBUG  : SNMPMessage: return value for build message: (0)
20230511.11:18:45: 24604: (4)DEBUG  : SNMPMessage: return value for build message: (0)
20230511.11:18:45: 24604: (4)DEBUG  : SNMPMessage: return value for build message: (0)
20230511.11:18:45: 24604: (4)DEBUG  : SNMPMessage: return value for build message: (0)
20230511.11:18:45: 24604: (4)DEBUG  : SNMPMessage: return value for build message: (0)
20230511.11:18:45: 24604: (4)DEBUG  : SNMPMessage: return value for build message: (0)
20230511.11:18:45: 24604: (4)DEBUG  : SNMPMessage: return value for build message: (0)
20230511.11:18:45: 24604: (4)DEBUG  : SNMPMessage: return value for build message: (0)
20230511.11:18:45: 24604: (4)DEBUG  : SNMPMessage: return value for build message: (0)
20230511.11:18:45: 24604: (4)DEBUG  : SNMPMessage: return value for build message: (0)
20230511.11:18:45: 24604: (4)DEBUG  : SNMPMessage: return value for build message: (0)
20230511.11:18:45: 24604: (4)DEBUG  : SNMPMessage: return value for build message: (0)
1
connected to objects port and listening....
Conflict Zone ID: 73,type 2, phase: 4, veh_id: 5456;546[illegible], my_time: 1683825643.6,local_time: 2023-05-11 11:20:43.6,origin_zone1: 70, origin_zone2: 23
Conflict Zone ID: 73,type 2, phase: 4, veh_id: 6154;61[illegible], my_time: 1683825872.5,local_time: 2023-05-11 11:24:32.5,origin_zone1: 70, origin_zone2: 23
Conflict Zone ID: 28,type 1, phase: 6, veh_id: 6409;6450, my_time: 1683825985.0,local_time: 2023-05-11 11:26:25.0,origin_zone1: 67, origin_zone2: 69
```

Figure 5-7 Observing and verifying the captured near-misses using the LiDAR UI

5.7.2 Near-Miss Validation with the Pan-Tilt-Zoom Camera

The researchers were requested to validate at least 100 LiDAR-reported near-misses by observing and comparing the synchronous videos recorded at Intersection 7122. A high-resolution Pan-Tilt-Zoom (PTZ) camera was installed at the NW corner of Intersection 7122 and well-positioned to capture the near-misses associated with the NB approach. The PTZ camera is watermarked with timestamps as well. Note that the video had to be recorded multiple times because the PTZ camera's largest field of view could not cover all the conflict areas with near-misses. Fig. 5-8 demonstrates the approximate conflicting areas where the researchers verified the LiDAR-reported near-misses from the recorded videos. According to the timestamps of LiDAR-reported near-misses and their types, the researchers replayed the recorded video clips around the same timestamp (watermarked at the top right corner of the video clips) and verified whether there were conflicting vehicles and/or pedestrians that were too close. If so, then the LiDAR-reported near-miss was identified. If not, then it was not verified.

Type II near-misses (EBR vehicles vs. pedestrians) turned out to be rare, and the researchers had to make the conflict zone larger to let the LiDAR report the Type II near-miss (Fig. 5-8).

Type III near-misses (the red-light running) identification requires the traffic signal status. The researchers used the opposing traffic light (phase 6) to verify the red-light runners of phase 2 because phases 2 and 6 start and end yellow at the same time. (Fig. 5-9)

Although the researchers did observe the NB U-turn vehicles from the PTZ cameras occasionally, the NB U-turn movement was prohibited at Intersection 7122. The occurrence of Type-V near-misses was very rare (less than 20 per month according to the LiDAR report), and there were no LiDAR-reported Type-V near-misses (U-turn vehicles vs. permissive right-turn vehicles) when the live stream videos were recorded.

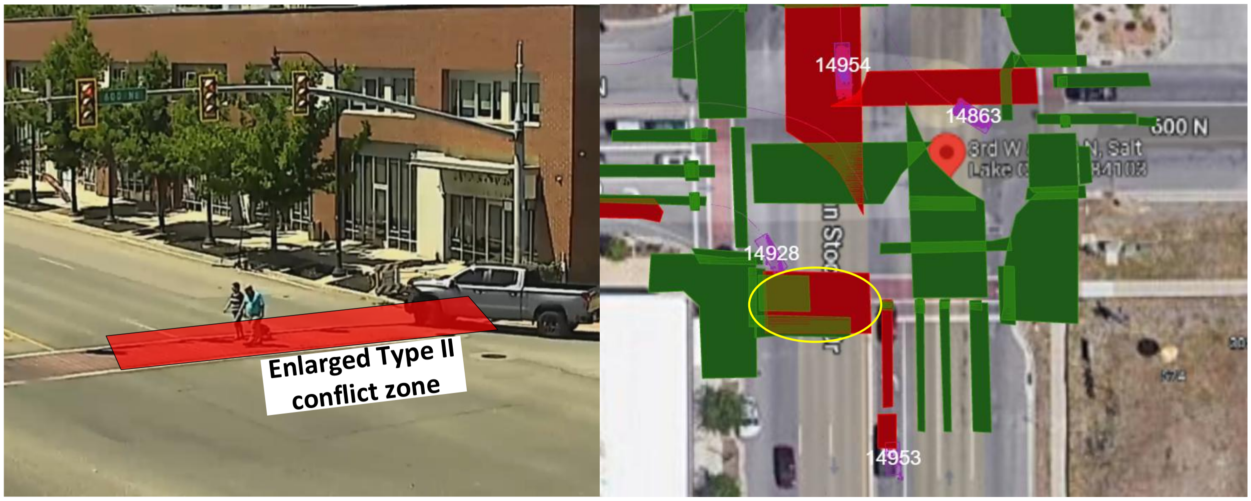


Figure 5-8 Extended Conflict Zone for Type-II near-miss verification

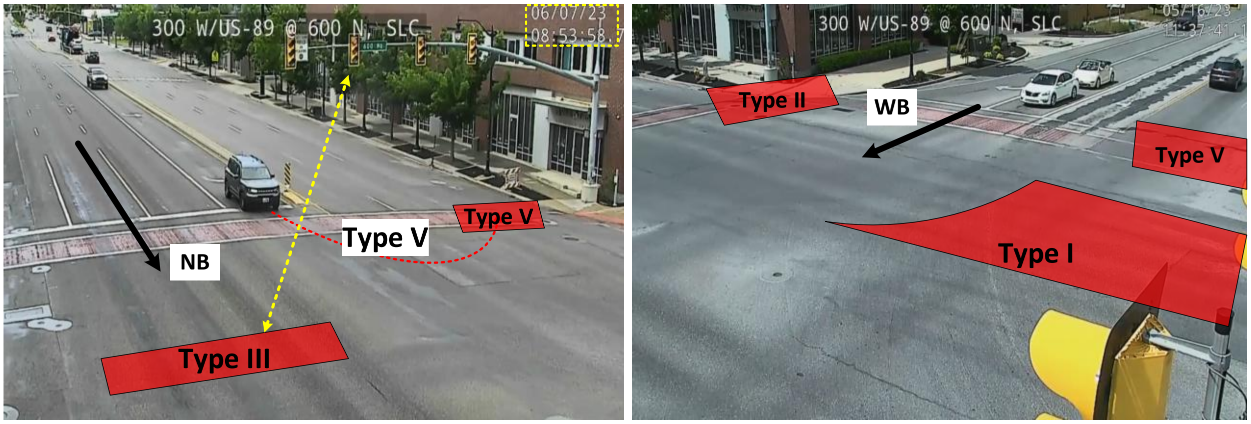


Figure 5-9 Videos and the corresponding conflict zones

Table 5-1 shows the summary of the verification of 103 LiDAR-reported near-misses. The ground truth was from the researchers' manual observation and verification from the recorded videos, according to the timestamps of LiDAR-reported near-misses. The short video clips can be accessed from the following YouTube links:

https://youtu.be/dsOerookmTM?t=2

https://youtu.be/P38_1c-mvKs

Table 5-1 Summary of LiDAR-reported near-miss verification.

Near-miss	Type Description	Total	Verified	False Alarm	Accuracy (%)
Type I	Perm LT vehs vs. thru traffic	53	48	5	90.6
Type II	Perm RT vehs vs. pedestrians	4	3	1	75
Type III	Red-light runner	40	38	2	95
Type IV	Perm LT vehicles vs. pedestrians	6	4	2	66.7
Type V	U-turn vs. right turn	0	0	0	-

Table 5-1 reveals the following information, accompanied by the researchers' reasoning and explanations.

Type I (permissive LT traffic vs. opposing through traffic) and Type III (red-light running) near-misses dominate the overall near-misses at intersections. The conflicting objects are vehicles that meet the minimal speed thresholds.

The researcher furthermore investigated the causes of the false alarms. A noticeable reason for Type-I false alarms is that there was an exceptionally high ratio of heavy-duty trucks at intersection 7122. Excessive Class 7 to Class 13 heavy-duty trucks (according to the Federal Highway Administration's definition)[4] passed that intersection every day because of the nearby quarry and oil refineries. The LiDAR perception software had a hard time identifying and binding those vehicles. The perceived heavy-duty trucks also overreached multiple zones within the intersections and triggered false alarms.

The red-light running (Type III): There is a through-right shared lane on the NB approach. From time to time, the permissive right-turn vehicles also met all the conditions set for the red-light running of the NBT vehicles (origin zone, conflict zone, minimal speed threshold, turning during red, etc.). To minimize such false alarms, the researchers moved the conflict zone for identifying the red-light runners further into the intersection to prevent the NRT vehicles from reaching it. While this approach can exclude the false red-light running by the permissive RT vehicles, it also increases the chance of identifying the normal vehicles that had left the stop line before the expiration of the yellow, such as the red-light runners. Fig. 5-10 demonstrates the dilemma of red-light running with the presence of a shared through-right lane. A red-light runner can be identified if a vehicle reaches the conflict zone but has not completely left the origin zone when the yellow expires. With the presence of a through-right shared lane, the permissive RT vehicle during red may trigger a false red-light running if the conflict zone is close to the stop line. On the other hand, if the conflict zone is placed further into the intersection, the false alarm caused by permissive right-turn (RT) vehicles will be minimized. However, a small percentage of through vehicles are likely to be mistaken for red-light runners when they reach the conflict zone (When this vehicle is examined for if it is running the red light). The researchers verified these speculations by observing the red-light running videos. Some passenger cars, which had left the stop bar before the yellow expired, were still identified as red-light runners when they ran into the conflict zone during the all-red. However, all red-light running trucks were verified as they could reach out origin and conflict zones at the same time. In this situation, additional zones (s) other than the origin and conflict zones must be introduced into the red-light-running capturing algorithm. The improvement of the red-light-running capturing algorithm will be left for further development.

[4] https://www.fhwa.dot.gov/policyinformation/tmguide/tmg_2013/vehicle-types.cfm

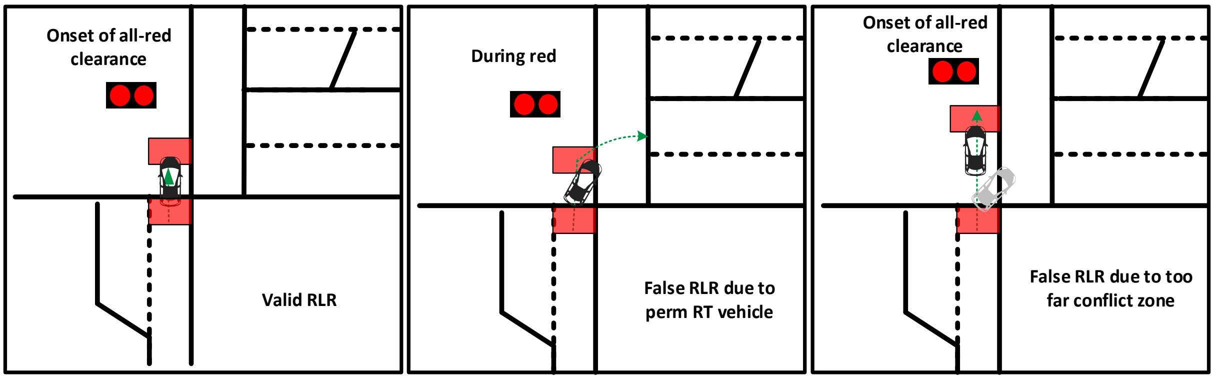

Onset of all-red clearance
Valid RLR
During red
False RLR due to perm RT vehicle
Onset of all-red clearance
False RLR due to too far conflict zone

Figure 5-10 False zone red-light running capturing with the presence of through-right shared lane

The sample sizes of Type-II and Type-IV pedestrian-involved near-misses were too low for meaningful statistical analysis. Some near-misses were indeed verified, though. The low accuracy was caused by the perception software's inefficiency instead of the algorithms developed by the researchers[5]. The perception software had some difficulties identifying the conflicting vehicles and pedestrians. In such situations, the pedestrians were mostly occluded by conflicting vehicles, generating a unique challenge for the perception software. Solving this problem will rely on the improvement of LiDAR perception software, either by the robotics research community or the LiDAR industry. In general, all LiDAR perception solutions are derived either from the open-source platform of robotics operation systems (ROS)/ROS2 or from Apollo (Seyond SIMPL). These platforms were originally developed for robotics and autonomous vehicles.

Lastly, it needs to be pointed out that the experiment designed for this task has limitations. It can only identify the accuracy rate and false alarms of LiDAR-reported near-misses. If the algorithm missed a near-miss, the experiment could not know and re-identify the missing near-miss, either. In other words, this validation can only evaluate the false alarm rate but not the missing rate. To address this issue, it will be necessary to engage multiple near-miss identification solutions, such as AI video analytics, to cross-compare and verify the different outputs. One example is the *AI-Empowered Video Analytics for Smart Transportation,* or AVAST, developed by the University of Texas, Arlington. The researchers on this project explored the possibility of capturing near-misses from one of the recorded (offline) video clips, and AVAST also captured many near-misses. The list of near-misses captured by the two systems can be compared to see if

[5] Note that the latest perception software on the market (2015) can handle such challenges better than before.

the LiDAR-based system missed any near-misses. Nonetheless, this effort is out of the scope of this project and will be reserved for the future.

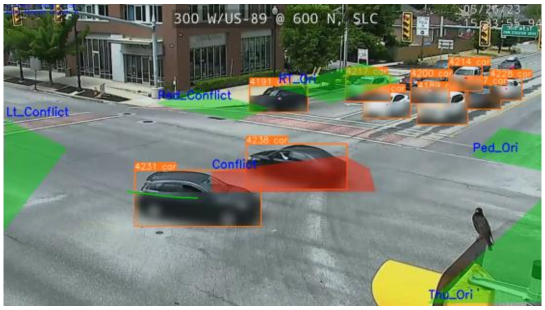


Figure 5-11 Near-miss capturing at UDOT Intersection 7122 using AVAST[6]

[6] AVAST is an alternative safety-centric traffic detection solution based on AI and video streams developed by the University of Texas at Arlington.

6 EVALUATIONS OF A NOVEL DYNAMIC FLASHING YELLOW ARROW STRATEGY (D-FYA) BASED ON LIDAR TRACKING

6.1 Background

Flash yellow arrow (FYA) is widely adopted for permissive left-turn movements after the related research concluded that the FYA would improve traffic safety (Noyce et al., 2014). However, the current FYA mechanism does not separate permissive left-turn vehicles from concurrent crossing pedestrians. As a result, pedestrian crashes reportedly increased at certain locations after the implementation of FYA. To address this issue, agencies either turn the FYA off or adopt a special feature in some brands of traffic signal controllers, referred to as "minus pedestrian." The concept is temporarily suppressing the FYA for a cycle if the corresponding pedestrian phase is called. Fig. 6-1 shows the concepts of FYA and the "minus pedestrian." Although the "minus pedestrian" feature separates left-turn vehicles from concurrent crossing pedestrians and has been broadly adopted, it also eliminates most permissive left-turn capability for that cycle. This mechanism often creates excessive left-turn queues during peak hours when both pedestrian volumes and left-turn vehicle volumes are high.

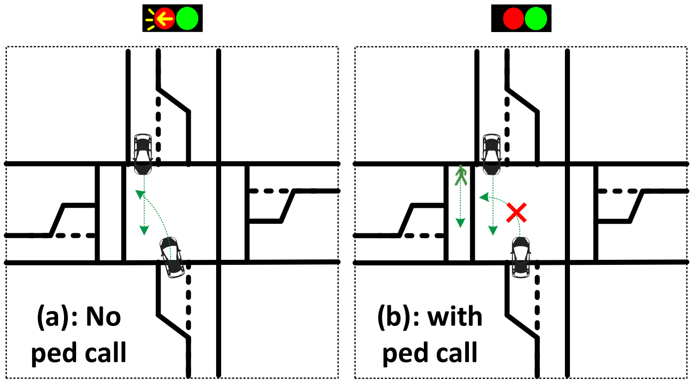


Figure 6-1 Demonstrations of FYA and "minus pedestrian"

Other ideas of delaying or canceling FYA were also reported but limited to pilot studies, including but not limited to: (1) canceling FYA or changing to a protected-only left-turn phase if the opposing traffic volume is too high in the last x minutes and recover after the volume comes down; (2) Delaying or canceling FYA if the opposing waiting left-turn vehicles block the driver's vision of this left-turn vehicle to find gaps. Note that canceling or delaying FYA is not a standard function in traffic signal controllers right now, while some controller manufacturers offer their proprietary method to realize this function. Therefore, this function can be safely implemented at an intersection controlled by certain brands of controllers.

In this research, the researchers aim to demonstrate an advanced algorithm developed by the University of Texas at Arlington to predict and respond to pedestrians' crossing intent based on tracking their behavior. Then it notifies the controller to delay or cancel FYA until the crossing pedestrians are safe. It is observed that pedestrians make or change their crossing decisions much more randomly than vehicles. To name a few, if a pedestrian wishes to go to the diagonal corner at an intersection, he or she may push both pedestrian buttons. The pedestrian will use the first activated pedestrian phase and "ghost" the second one. A pedestrian may push the button but change their mind and walk away. The latched pedestrian call will still activate the pedestrian phase even without pedestrians. Or a pedestrian pushes the button for the wrong crosswalk, and so he or she decides to ignore that pedestrian phase and wait for the right one. Such random decisions create a huge challenge for pedestrian-involved D-FYA operations driven by traditional pedestrian push buttons. By nature, the push buttons are just used to tell the controller the presence of waiting pedestrians. Then the controller carries out the programmed D-FYA operations following strong assumptions about pedestrian behaviors. Since the push buttons cannot detect and respond to pedestrians' unexpected behaviors, the corresponding D-FYA operations will likely be less effective than expected in practice.

6.2 Challenges and Mitigations

In the middle of this research task, the installed LiDAR sensors began to fail repeatedly. As a result, the proposed demonstration could not be finished despite many salvage efforts. UDOT acknowledged this issue and agreed to rescope this research task to be more instructive and introductory. The new scope includes: an introduction of LiDAR-based D-FYA operation and report on a previous evaluation before this project; a summary of a similar D-FYA operation designed by UDOT. UDOT's D-FYA is driven by traditional pedestrian push buttons.

6.3 Dynamic D-FYA Operations Based on LiDAR Tracking

The LiDAR-based D-FYA operation is based on tracking pedestrians (*31*). As shown in Fig. 6-2, concurrent crossing pedestrians have a conflict with left-turn vehicles only when they are within the so-called "hazard zone."

6.3.1 "Three-Zone" Pedestrian Tracking with LIDAR Sensors

In reality, many pedestrians push the pedestrian buttons, become impatient, and then choose to cross or "jaywalk" before the "WALK" sign starts. As a result, neither the pedestrian phase nor FYA suppression is needed for that cycle. In addition, the D-FYA can (and should) only protect those pedestrians who follow traffic lights, as overprotecting both legitimate and illegitimate crossing pedestrians will considerably interrupt traffic signal operations for vehicles. To address these issues, we design a "three-zone" method to filter and only track those legitimate crossing pedestrians as shown in Fig. 6-2. A pedestrian needs to enter the wait zone first and push the pedestrian button to be considered legitimate. The waiting zones (Zone 1 and Zone 1') of each pedestrian phase are defined as "far-end" (Zone 1) and "near-end" (Zone 1') according to their distances to the permissive left-turn vehicles. During WALK, if a pedestrian in Zone 1(far-end) and/or Zone 1'(near-end) enters the boundary zones (Zone 2 and Zone 3), then this pedestrian is considered a legitimate pedestrian. If the same pedestrian reaches the other end, then this pedestrian crossing is considered finished. If a ped call is placed (either by push buttons or recalls) but no legitimate pedestrians enter the intersection, the pedestrian request is then considered void and ignored. The three-zone method will discard those "jaywalking" pedestrians by default. However, if agencies wish to consider such jaywalkers as well, they can simply extend the waiting zones, boundary zones, and hazard zones according to observed jaywalking activities (e.g., extending further back from the intersection). This configuration is in essence a retrofit to the standard algorithm with the same logic to cover more legitimate and illegitimate pedestrians.

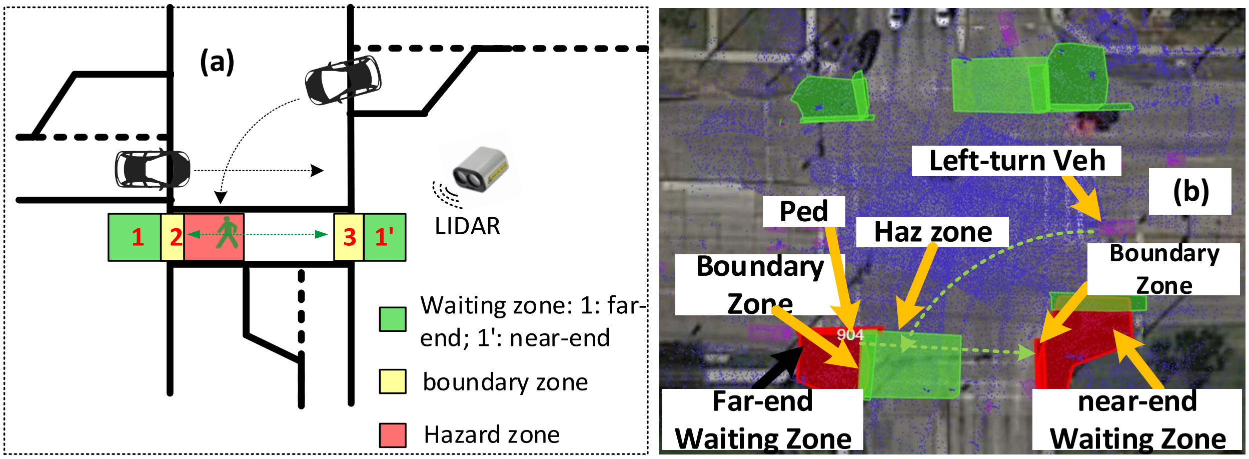


Figure 6-2 Three-zone pedestrian detection method at intersections: a: demonstration, b: zone settings in the field about WB left-turn vehicles (City of Irving, TX)

6.3.2 Dynamic Flash Yellow Arrow (D-FYA) Based on Pedestrian Tracking

Whenever a signal phase enters yellow, the controller will report what the next phase is. This mechanism allows the D-FYA algorithm to identify whether a FYA of interest is about to start. For instance, at Intersection 7122, if the next phase is phase 2, then FYA 1 is about to start. Then the algorithm will immediately perform the following actions:

- **STEP 1**: Check if this phase has a concurrent pedestrian phase. If yes, go to Step 2. If no, STOP
- **STEP 2**: Check if the pedestrian button is pushed or the pedestrian phase is called/recalled. If yes, go to Step 3. If no, STOP
- **STEP 3**: Examine the existence of pedestrians in far-end and near-end waiting zones. There are two scenarios:
  - No pedestrians are detected at either waiting zone, the D-FYA algorithm will keep the original FYA settings. Then go to STEP 4.
  - Pedestrians are detected at one or two waiting zones, then the D-FYA algorithm will hold FYA temporarily. Then go to STEP 4.
  - When green or WALK starts, the D-FYA algorithm will check STEP 4 to make the final decision on FYA for this cycle.
- **STEP 4**: At this step, there are four possibilities for pedestrians to enter the intersection from the two sides of the waiting zones:
  - During the WALK time, if pedestrians in the far-end waiting zone (e.g., Zone 1 in Fig. 6-2) enter the intersection (e.g., Zone 2 in Fig. 6-2), but no pedestrians in the near-end waiting zone (e.g., Zone 1' in Fig. 6-2) enter (e.g., Zone 3 in Fig. 6-2). The FYA is suspended until all pedestrians have left the “hazard zone” (See Fig. 6-2). Then the FYA is reactivated until the current phase ends.

- During the WALK time, if pedestrians in the near-end waiting zone enter the intersection while no pedestrians in the far-end waiting zone enter, then the FYA is suspended until all near-end pedestrians reach the other side of the intersection (e.g., enter the boundary zone on the other side). Then the FYA is reactivated until the current phase ends.
- During the WALK time, if pedestrians enter the intersection from both sides, the FYA is suspended until all pedestrians reach the other side.
- During the WALK time, if no pedestrians enter from either side, the FYA is activated until the current phase ends.

Step 4 is the final step of this algorithm for each phase.

Discussion:

1. The decisions on FYA at Step 3 are temporary because a detected person in the waiting zones may not cross, or a pedestrian may mistakenly push a pedestrian button. The final decision of keeping or suspending an FYA will be determined after the green/WALK starts.
2. Note that the decision to activate FYA can be made only once for each phase in each cycle. This request is mandated by the MUTCD and monitored by MMU.
3. If a pedestrian "jaywalks" and gets out of the boundary zone when reaching the intersection's other side, LIDAR sensors will lose tracking of it. The missing pedestrian will be allocated the longest walking time beyond which this person is considered to have crossed.
4. The proposed D-FYA is particularly effective when the associated green is much longer than the needed pedestrian crossing time. Once all pedestrians are cleared, the FYA is reactivated and can provide a significant permissive capacity for left-turn vehicles. By contrast, the current "minus pedestrian" mechanism will delay the FYA according to the predetermined timing (i.e., following strong assumptions of pedestrian behaviors). It may not be able to protect the slowest crossing pedestrians who are most vulnerable in this context.

6.3.3 Evaluation of D-FYA's Performance in the "Shadow" Mode

In this experiment, we evaluated the performance of the proposed D-FYA algorithm in the field by verifying its real-time decisions according to the observed pedestrian behaviors. The "shadow" mode means all the traffic signal inputs and pedestrian behaviors are instantaneously collected in real time to determine whether and when to activate FYA. However, the D-FYA decisions are not implemented in the controller but just logged for observers to evaluate. The purpose of this experiment is to evaluate the algorithm's reliability and accuracy in the field. The selected intersection was Cooper Street at UTA Blvd, a major intersection connecting two urban campuses of the University of Texas at Arlington. The daily pedestrians crossing Cooper Street (mainline) range from 1,000 to 1,500 in a school day. The phasing sequence and pedestrian tracking zones are shown in Fig.6-3. There are four flashing yellow arrows on all four approaches. Whenever a phase starts, the D-FYA algorithm will run and report its findings (e.g., the presence of waiting pedestrians) and decisions (e.g., holding or activating an FYA) on the console screen. At the same time, a researcher compared the reported decisions on screen with his observations in the field. For example, if the researcher saw that two pedestrians were waiting to cross in the far-end zone, he would expect the algorithm will report the same finding and propose delaying the FYA once the last phase entered yellow. The observation was carried out over 100 signal cycles with pedestrian crossings. Table 6-1 demonstrates how the D-FYA decisions were recorded and verified, using 5 cycles as an example.

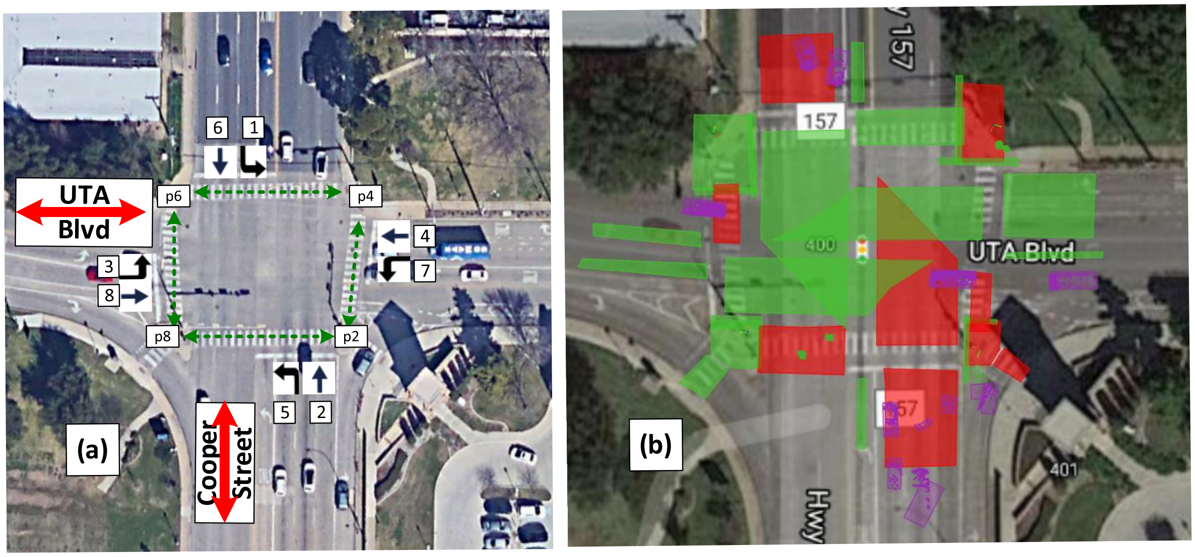


Figure 6-3 Phasing sequence (a) and pedestrian-sensing zone layout (b) Cooper Street at UTA Blvd, Arlington, TX

Table 6-1 Records and Verification of Shadow-\Mode D-FYA Decisions.

Signal Cycle	Corresponding signal phases	near-end ped presence	far-end ped presence	Both-ends ped presence	No ped presence	FYA started as scheduled	FYA delayed	FYA cancelled	Comment
1	8	1	0	0	0	0	0	1	1*
2	4	1	0	0	0	0	1	0	1*
3	4	0	0	0	1	1	0	0	2*
4	4	1	0	0	0	0	1	0	1*
5	4	0	1	0	0	0	1	0	1*

Note: 1: verified by the researcher in the field; 2* verified a ped phase call and the pedestrian presence*

The case study was conducted for 100 cycles in the field. There were 70 cycles for which at least one pedestrian phase was called. Among those 70 cycles, 25 cycles only had near-end pedestrians, 25 cases had far-end pedestrians, and 9 cases had pedestrians on both sides. Comparing the D-FYA reported on the screen with what we observed in the field, we concluded that the D-FYA algorithm could make correct decisions in 93 out of 100 cycles. Table 6-2 summarizes the D-FYA's performance under various scenarios.

Table 6-2 Performance Summary of D-FYA Algorithm Under Different Scenarios.

Cycles with no ped calls	Cycles only with near-end peds	Cycles only with far-end peds	Cycles with both-end peds	Cycles with ped calls but no ped presence	The accuracy rate of the D-FYA algorithm
30	25	25	9	11	93%

After finishing the experiment in the field, we further analyzed the recorded video and identified the possible reasons for incorrect D-FYA decisions. In those failed cases, the pedestrians either leaned on traffic light poles or multiple pedestrians stood too close for the LIDAR tracking algorithm to separate them effectively. This accuracy rate should further increase if the LIDAR tracking algorithm can improve in the future.

6.4 UDOT's D-FYA Operation Based on Pedestrian Push Buttons and Logic Processors in Controllers

In parallel to this research task, UDOT's traffic signal group also developed a similar D-FYA operation based on traditional pedestrian push buttons. UDOT's signal engineer, Matthew

Luker[7], used a special configuration of pedestrian push buttons and the Logic Processor function in Econolite/Q-Free controllers to identify not only the called pedestrian phase but also the street corner where the pedestrian stands. The programs, in the form of a spreadsheet, are appended to this report's appendix. At a high level, this approach is summarized as follows:

1. The default pedestrian phase terminal in traffic signal cabinets can only accept up to 4 inputs from push buttons, while this special D-FYA operation requires 8 inputs (two distinguishable inputs at each street corner). To address this issue, UDOT split the wire of one pedestrian phase (two push buttons) and connected it to the cabinet's back panel for vehicle detector inputs. The detector panel is only used when the traditional inductive loop detectors are used; they are not used for more recent vehicle detectors like video, radar, or LiDAR. This is the case in Utah. Note that these channels must not be used by any other platforms, like a connected vehicle system.

2. Using the proprietary Logic Processors in controllers, whenever a pedestrian pushes a pedestrian button, the controller will understand which pedestrian phase is called and at which street corner. In light of Fig. 6-2, a pedestrian at the far end needs to pass the hazard zone to be completely separate from the left-turn vehicles, while a pedestrian at the near end needs to cross the entire crosswalk to be safe. As a result, the corresponding FYA delay time is different. UDOT calculates the needed time for both directions (from far to near and from near to far) of a pedestrian phase to be separated from left-turn vehicles. Whenever a push button is pressed, the programmed Logic Processor will delay the corresponding time to the corresponding FYA according to the calculation.

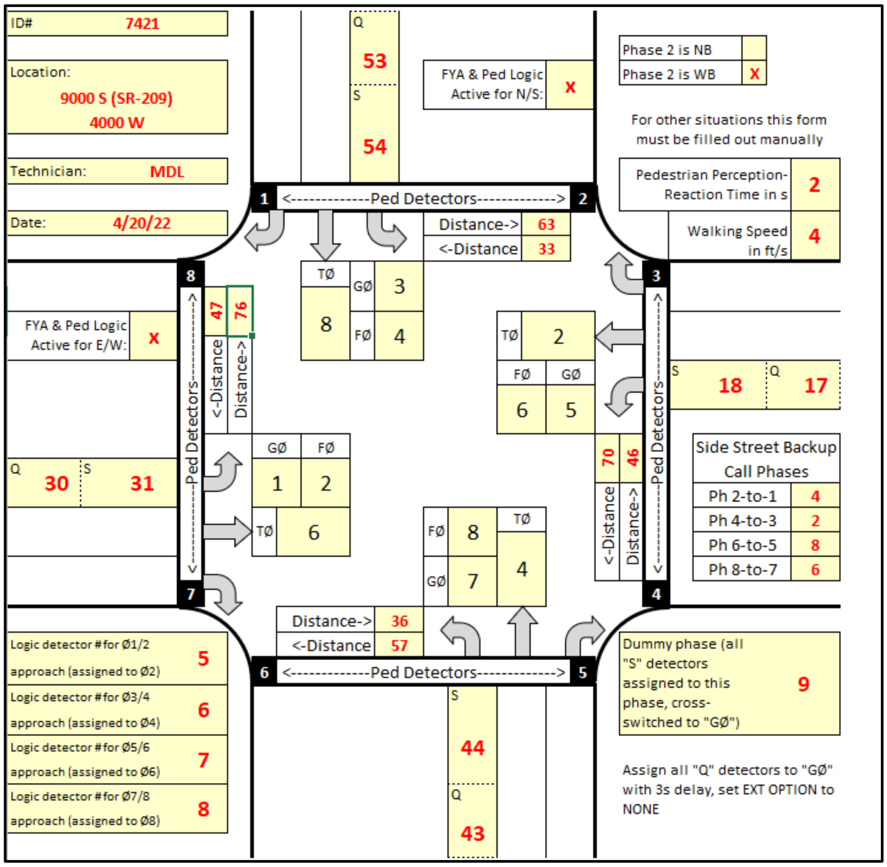
Fig. 6-3 is an illustration of UDOT-FYA based on split buttons and logic processors. According to the UDOT's internal memorandums, limitations are recognized, including but not limited to:

1. UDOT D-FYA operation is based on Logic Processors in the controller, which are provided for simple add-on features, such as peer-to-peer communications and virtual detector mappings. The UDOT D-FYA operation is beyond the normal usage of Logic Processors. 52 Logic processors were used out of the total provided 100 Logic Processors

[7] Mr. Luker is the ITS manager of Utah DOT as of 2025.

in each controller. As a result, it may be difficult to apply the UDOT D-FYA operations together with other logic-processor-based operations in one controller.

2. When the controller runs in the coordination mode, the coordination phase must disable WALK-REST and WALK-RECYCLE. In the coordination mode, WALK-REST can extend WALK to a very long duration during which pedestrians are allowed to enter the crosswalk legitimately. In the meantime, the FYA delay should only be reasonably long to avoid losing the respect of drivers or causing excessive delays. WALK-RECYCLE also turns on WALK multiple times during the coordination phase to allow pedestrians to cross. In these cases, FYA delay cannot fully separate later crossing pedestrians from left-turn vehicles.
3. FYA delay by the Logic Processors in controllers has a maximum value of 25.5 seconds. So, if the intersection is very large, then 25.5 seconds may not be sufficient for pedestrians to walk from one side to the other (e.g., from near to far). As such, adopting the UDOT D-FYA operation needs to be carefully evaluated if the intersection is very large (e.g., 8+ lanes wide).



ID#	7421
Location:	9000 S (SR-209) 4000 W
Technician:	MDL
Date:	4/20/22
FYA & Ped Logic Active for N/S:	X
FYA & Ped Logic Active for E/W:	X
Phase 2 is NB	
Phase 2 is WB	X

For other situations this form must be filled out manually

Pedestrian Perception-Reaction Time in s	2
Walking Speed in ft/s	4

Side Street Backup Call Phases	
Ph 2-to-1	4
Ph 4-to-3	2
Ph 6-to-5	8
Ph 8-to-7	6

Logic detector # for Ø1/2 approach (assigned to Ø2)	5
Logic detector # for Ø3/4 approach (assigned to Ø4)	6
Logic detector # for Ø5/6 approach (assigned to Ø6)	7
Logic detector # for Ø7/8 approach (assigned to Ø8)	8
Dummy phase (all "S" detectors assigned to this phase, cross-switched to "GØ")	9

Assign all "Q" detectors to "GØ" with 3s delay, set EXT OPTION to NONE

Figure 6-4 Illustration of UDOT D-FYA based on split push buttons and logic processors (Source: UDOT)

6.5 Remarks

Since the LiDAR-based D-FYA adopts modern programming methods and is based on traffic control standards, it does not have the No. 1 and No. 3 limitations in the UDOT D-FYA operation. However, it also requires disabling WALK-REST and WALK-RECYCLE to avoid delaying FYA for too long unless an internal timer is set up to limit the identification of legitimate crossing pedestrians only during the programmed WALK and the first few seconds of pedestrian clearance. After the timer expires, later arriving pedestrians will be discarded in the algorithm's decision process even if they enter during (extended) WALK.

By nature, both UDOT D-FYA and LiDAR-based D-FYA aim to protect the concurrent crossing pedestrians who have been waiting and will enter the crosswalk during the programmed WALK time and the first few seconds of pedestrian clearance.

7 SUMMARY OF RESEARCH FINDINGS

This chapter presents key takeaways and actionable recommendations derived from this multi-year research on using LiDAR sensors to detect traffic near-misses and enable safety-centric traffic control.

7.1 Key Findings

1. LiDAR-Based Near-Miss Detection Works:
    - The system reliably detected multiple types of near-misses in real-time.
    - Verified through synchronized PTZ camera footage.
    - Achieved over 99% object detection accuracy with average latency of just 0.13 seconds.
2. Near-Miss Frequency is High:
    - One intersection experienced 400–600 near-misses per approach per day.
    - Conflicts between permissive left-turn vehicles and opposing through vehicles (Type I), and red-light running (Type III), were most common.
3. Dynamic Conflict Zone Design Matters:
    - Traditional point-based conflict models are insufficient.
    - The study proposed "conflict zones" shaped by real trajectory data and vehicle kinematics.
    - Severity of near-misses assessed using time-to-collision (TTC) and post-encroachment time (PET).
4. D-FYA Concept is Promising but Needs Reliable Hardware and Better Perception
    - Intended to dynamically cancel or delay flashing yellow arrows based on real pedestrian movements.
    - Could not be fully demonstrated due to LiDAR hardware failures in Utah.
    - Concept documented and compared with UDOT's push-button-based logic.

7.2 Recommendations

1. Design Guidelines for Conflict Zones:
    - Develop standard but customizable templates for conflict zone layouts.
    - Use TTC and PET thresholds to capture both high- and low-severity near-misses.
2. Further integrate with Signal Control Systems:
    - Link LiDAR-derived data with Adaptive Traffic Signal Control or ATSPM.
    - Allow signals to react to emerging traffic conflicts and traveler risks in real time.
3. Leverage Near-Miss Data for Proactive Safety Planning:
    - Use collected near-miss metrics to guide engineering decisions, prioritization of safety projects, and education campaigns.

In conclusion, this research confirms that LiDAR systems offer a new and powerful way to identify and mitigate traffic risks before crashes occur. With further deployment and integration, they can become central tools in proactive intersection safety management.

REFERENCES

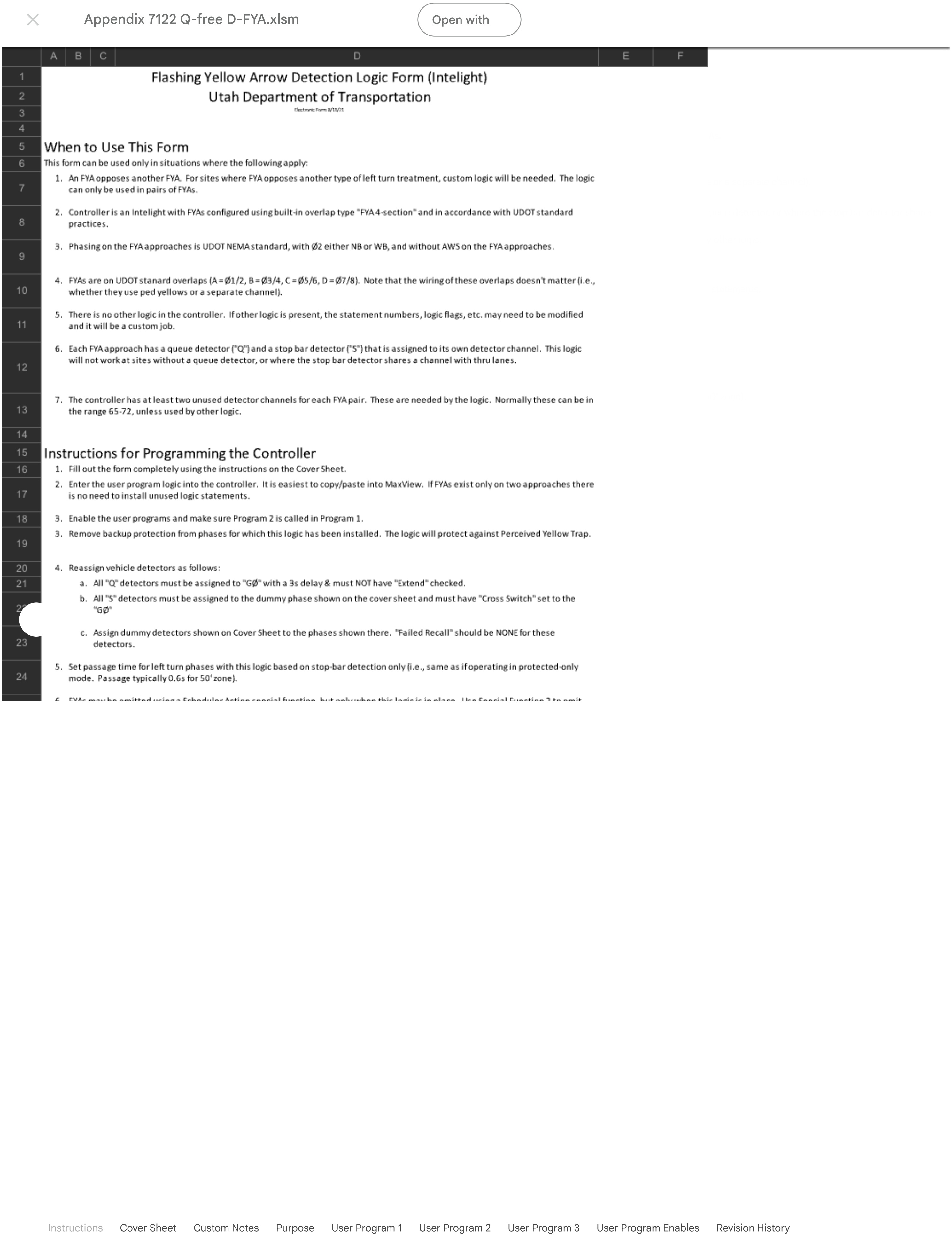
[1] Li, P. T., X. T. Yang, and C. Liu. Utilizing LiDAR Sensors to Detect Pedestrian Movements at Signalized Intersections.In, Utah. Dept. of Transportation. Division of Research, 2022.
[2] Cesme, B., A. Stevanovic, C. Day, B. Schroeder, J. Bonneson, and D. Hale. Traffic Signal Change and Clearance Interval Pooled Fund Study: Synthesis Report.In, United States. Department of Transportation. Federal Highway Administration, 2023.
[3] Perkins, S. R., and J. I. Harris. Traffic conflict characteristics-accident potential at intersections. *Highway Transportation Record,* Vol. 225, 1968, pp. 35-43.
[4] Older, S. J., and B. R. Spicer. Traffic conflicts—a development in accident research. *Human factors,* Vol. 18, No. 4, 1976, pp. 335-350.
[5] Baker, W. T. An evaluation of the traffic conflicts technique. *Highway Research Record*, No. 384, 1972.
[6] Glauz, W. D., and D. J. Migletz. Application of Traffic Conflict Analysis at Intersections. *National Cooperative Highway Research Program Report,* Vol. 219, 1980.
[7] Glauz, W. D., K. Bauer, and D. Migletz. Expected traffic conflict rates and their use in predicting accidents. *Transportation Research Record,* Vol. 1026, 1985, pp. 1-12.
[8] Gettman, D., and L. Head. Surrogate safety measures from traffic simulation models. *Transportation Research Record,* Vol. 1840, No. 1, 2003, pp. 104-115.
[9] Hyden, C. The development of a method for traffic safety evaluation: the Swedish conflicts technique.In*, No. 5*, Department of Traffic Planning, Engineering, Lund University, Sweden, 1987.
[10] Hayward, J. C. Near-miss Determination Through Use of a Scale of Danger. *Highway Research Record*, 1972, pp. 24-34.
[11] Allen, B. L., B. T. Shin, and P. J. Cooper. ANALYSIS OF TRAFFIC CONFLICTS AND COLLISIONS. *Transportation Research Record*, No. 667, 1978, pp. 67-74.
[12] Ozbay, K., H. Yang, B. Bartin, and S. Mudigonda. Derivation and validation of new simulation-based surrogate safety measure. *Transportation Research Record,* Vol. 2083, No. 1, 2008, pp. 105-113.
[13] Hossain, M., M. Abdel-Aty, M. A. Quddus, Y. Muromachi, and S. N. Sadeek. Real-time crash prediction models: State-of-the-art, design pathways and ubiquitous requirements. *Accident Analysis & Prevention,* Vol. 124, 2019, pp. 66-84.
[14] Cai, Q., M. Abdel-Aty, J. Yuan, J. Lee, and Y. J. T. r. p. C. e. t. Wu. Real-time crash prediction on expressways using deep generative models. Vol. 117, 2020, p. 102697.
[15] Wang, L., M. Abdel-Aty, J. Lee, and Q. Shi. Analysis of real-time crash risk for expressway ramps using traffic, geometric, trip generation, and socio-demographic predictors. *Accident Analysis & Prevention,* Vol. 122, 2019, pp. 378-384.
[16] Zheng, L., and T. Sayed. A novel approach for real time crash prediction at signalized intersections. *Transportation research part C: emerging technologies,* Vol. 117, 2020, p. 102683.
[17] Theofilatos, A., C. Chen, and C. Antoniou. Comparing machine learning and deep learning methods for real-time crash prediction. *Transportation Research Record,* Vol. 2673, No. 8, 2019, pp. 169-178.
[18] Basso, F., R. Pezoa, M. Varas, and M. Villalobos. A deep learning approach for real-time crash prediction using vehicle-by-vehicle data. *Accident Analysis & Prevention,* Vol. 162, 2021, p. 106409.

[19] Yuan, J., M. Abdel-Aty, Y. Gong, and Q. Cai. Real-time crash risk prediction using long short-term memory recurrent neural network. *Transportation Research Record,* Vol. 2673, No. 4, 2019, pp. 314-326.
[20] Li, P., M. Abdel-Aty, and J. Yuan. Real-time crash risk prediction on arterials based on LSTM-CNN. *Accident Analysis & Prevention,* Vol. 135, 2020, p. 105371.
[21] Khoda Bakhshi, A., and M. M. Ahmed. Real-time crash prediction for a long low-traffic volume corridor using corrected-impurity importance and semi-parametric generalized additive model. *Journal of transportation safety & security,* Vol. 14, No. 7, 2022, pp. 1165-1200.
[22] Elamrani Abou Elassad, Z., H. Mousannif, and H. Al Moatassime. Class-imbalanced crash prediction based on real-time traffic and weather data: A driving simulator study. *Traffic injury prevention,* Vol. 21, No. 3, 2020, pp. 201-208.
[23] Li, P., and M. Abdel-Aty. A hybrid machine learning model for predicting real-time secondary crash likelihood. *Accident Analysis & Prevention,* Vol. 165, 2022, p. 106504.
[24] Thapa, D., R. Paleti, and S. Mishra. Overcoming challenges in crash prediction modeling using discretized duration approach: An investigation of sampling approaches. *Accident Analysis & Prevention,* Vol. 169, 2022, p. 106639.
[25] Yu, R., Y. Wang, Z. Zou, and L. Wang. Convolutional neural networks with refined loss functions for the real-time crash risk analysis. *Transportation research part C: emerging technologies,* Vol. 119, 2020, p. 102740.
[26] Wang, Q., S. Gan, W. Chen, Q. Li, and B. Nie. A data-driven, kinematic feature-based, near real-time algorithm for injury severity prediction of vehicle occupants. *Accident Analysis & Prevention,* Vol. 156, 2021, p. 106149.
[27] Shangguan, Q., T. Fu, J. Wang, and T. Luo. An integrated methodology for real-time driving risk status prediction using naturalistic driving data. *Accident Analysis & Prevention,* Vol. 156, 2021, p. 106122.
[28] American Association of State Highway and Transportation Officials. *Policy on geometric design of highways and streets (7th Edition)*. 2018.
[29] Roess, R. P., E. S. Prassas, and W. R. McShane. *Traffic engineering (Fifth Edition)*. Pearson/Prentice Hall, 2019.
[30] Hancock, M. W., and B. Wright. A policy on geometric design of highways and streets. *American Association of State Highway and Transportation Officials: Washington, DC, USA,* Vol. 3, 2013.
[31] Chowdhury, F. R., P. S. Wang, and P. T. Li. Developing a tracking-based dynamic flash yellow arrow strategy for permissive left-turn vehicles to improve pedestrian safety at intersections. *Journal of transportation engineering, Part A: Systems,* Vol. 149, No. 4, 2023, p. 04023017.

Appendix A: UDOT Dynamic FYA Implementation with Controllers' Logic Processors

- Intersection 7122 (Q-Free MAXTIME)

- Intersection 7335 (Econolite ASC/3 or Cobalt)



Flashing Yellow Arrow Detection Logic Form (Intelight)

Utah Department of Transportation

Electronic Form 8/18/21

When to Use This Form

This form can be used only in situations where the following apply:

1. An FYA opposes another FYA. For sites where FYA opposes another type of left turn treatment, custom logic will be needed. The logic can only be used in pairs of FYAs.
2. Controller is an Intelight with FYAs configured using built-in overlap type "FYA 4-section" and in accordance with UDOT standard practices.
3. Phasing on the FYA approaches is UDOT NEMA standard, with Ø2 either NB or WB, and without AWS on the FYA approaches.
4. FYAs are on UDOT stanard overlaps (A = Ø1/2, B = Ø3/4, C = Ø5/6, D = Ø7/8). Note that the wiring of these overlaps doesn't matter (i.e., whether they use ped yellows or a separate channel).
5. There is no other logic in the controller. If other logic is present, the statement numbers, logic flags, etc. may need to be modified and it will be a custom job.
6. Each FYA approach has a queue detector ("Q") and a stop bar detector ("S") that is assigned to its own detector channel. This logic will not work at sites without a queue detector, or where the stop bar detector shares a channel with thru lanes.
7. The controller has at least two unused detector channels for each FYA pair. These are needed by the logic. Normally these can be in the range 65-72, unless used by other logic.

Instructions for Programming the Controller

1. Fill out the form completely using the instructions on the Cover Sheet.

2. Enter the user program logic into the controller. It is easiest to copy/paste into MaxView. If FYAs exist only on two approaches there is no need to install unused logic statements.

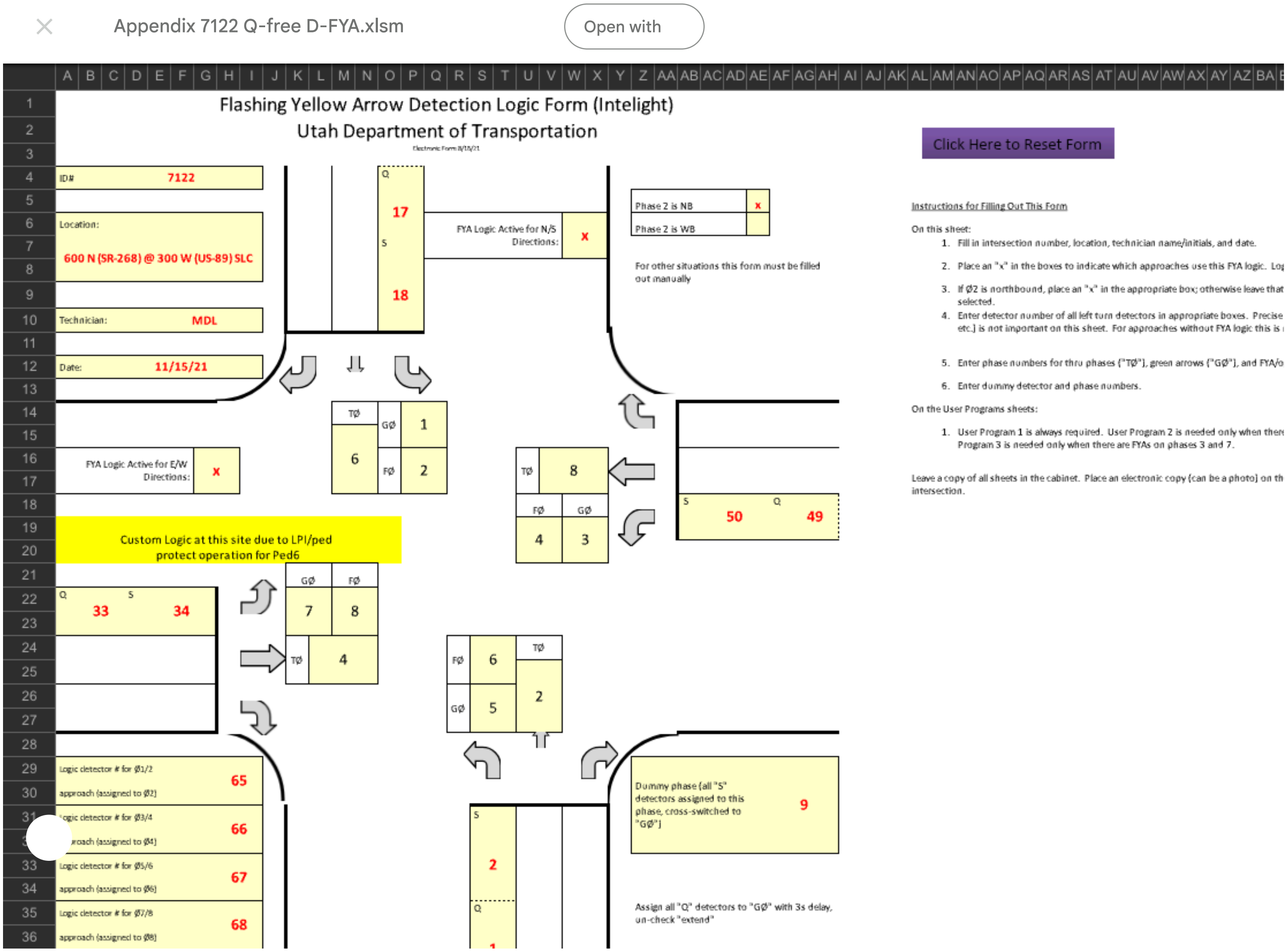
3. Enable the user programs and make sure Program 2 is called in Program 1.

3. Remove backup protection from phases for which this logic has been installed. The logic will protect against Perceived Yellow Trap.

4. Reassign vehicle detectors as follows:
   a. All "Q" detectors must be assigned to "GØ" with a 3s delay & must NOT have "Extend" checked.
   b. All "S" detectors must be assigned to the dummy phase shown on the cover sheet and must have "Cross Switch" set to the "GØ"
   c. Assign dummy detectors shown on Cover Sheet to the phases shown there. "Failed Recall" should be NONE for these detectors.

5. Set passage time for left turn phases with this logic based on stop-bar detection only (i.e., same as if operating in protected-only mode. Passage typically 0.6s for 50' zone).

6. FYAs may be omitted using a Scheduler Action special function, but only when this logic is in place. Use Special Function 2 to omit



Flashing Yellow Arrow Detection Logic Form (Intelight)

Utah Department of Transportation

Electronic Form 8/18/21

ID#	7122
Location:	600 N (SR-268) @ 300 W (US-89) SLC
Technician:	MDL
Date:	11/15/21

FYA Logic Active for N/S Directions: x

Phase 2 is NB	x
Phase 2 is WB	

For other situations this form must be filled out manually

FYA Logic Active for E/W Directions: x

Custom Logic at this site due to LPI/ped protect operation for Ped6

Logic detector # for Ø1/2 approach (assigned to Ø2)	65
Logic detector # for Ø3/4 approach (assigned to Ø4)	66
Logic detector # for Ø5/6 approach (assigned to Ø6)	67
Logic detector # for Ø7/8 approach (assigned to Ø8)	68

Dummy phase (all "S" detectors assigned to this phase, cross-switched to "GØ") 9

Assign all "Q" detectors to "GØ" with 3s delay, un-check "extend"

Instructions for Filling Out This Form

On this sheet:

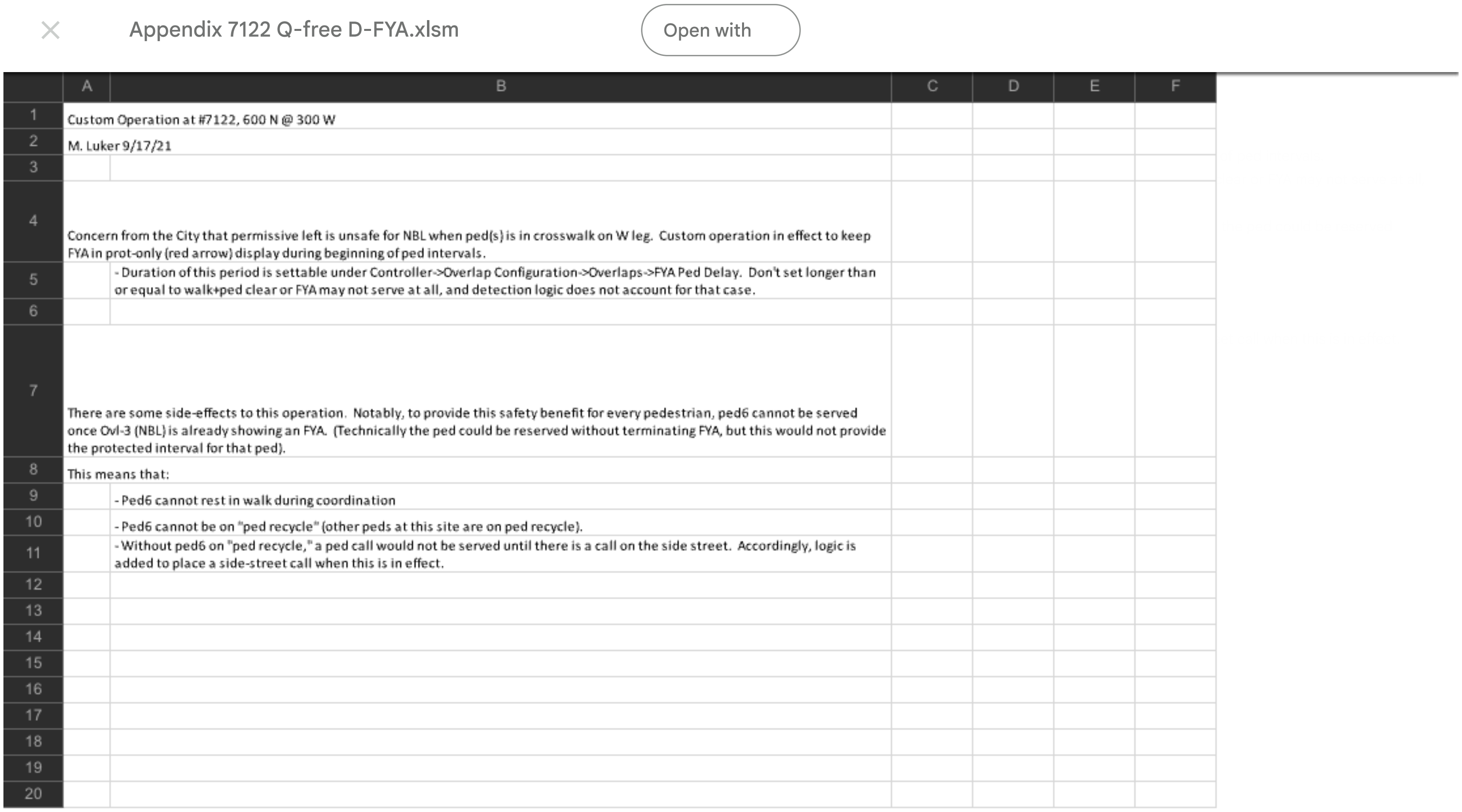
1. Fill in intersection number, location, technician name/initials, and date.
2. Place an "x" in the boxes to indicate which approaches use this FYA logic. Log
3. If Ø2 is northbound, place an "x" in the appropriate box; otherwise leave that selected.
4. Enter detector number of all left turn detectors in appropriate boxes. Precise etc.) is not important on this sheet. For approaches without FYA logic this is
5. Enter phase numbers for thru phases ("TØ"), green arrows ("GØ"), and FYA/o
6. Enter dummy detector and phase numbers.

On the User Programs sheets:

1. User Program 1 is always required. User Program 2 is needed only when there Program 3 is needed only when there are FYAs on phases 3 and 7.

Leave a copy of all sheets in the cabinet. Place an electronic copy (can be a photo) on th intersection.

Instructions Cover Sheet Custom Notes Purpose User Program 1 User Program 2 User Program 3 User Program Enables Revision History



Custom Operation at #7122, 600 N @ 300 W

M. Luker 9/17/21

Concern from the City that permissive left is unsafe for NBL when ped(s) is in crosswalk on W leg. Custom operation in effect to keep FYA in prot-only (red arrow) display during beginning of ped intervals.

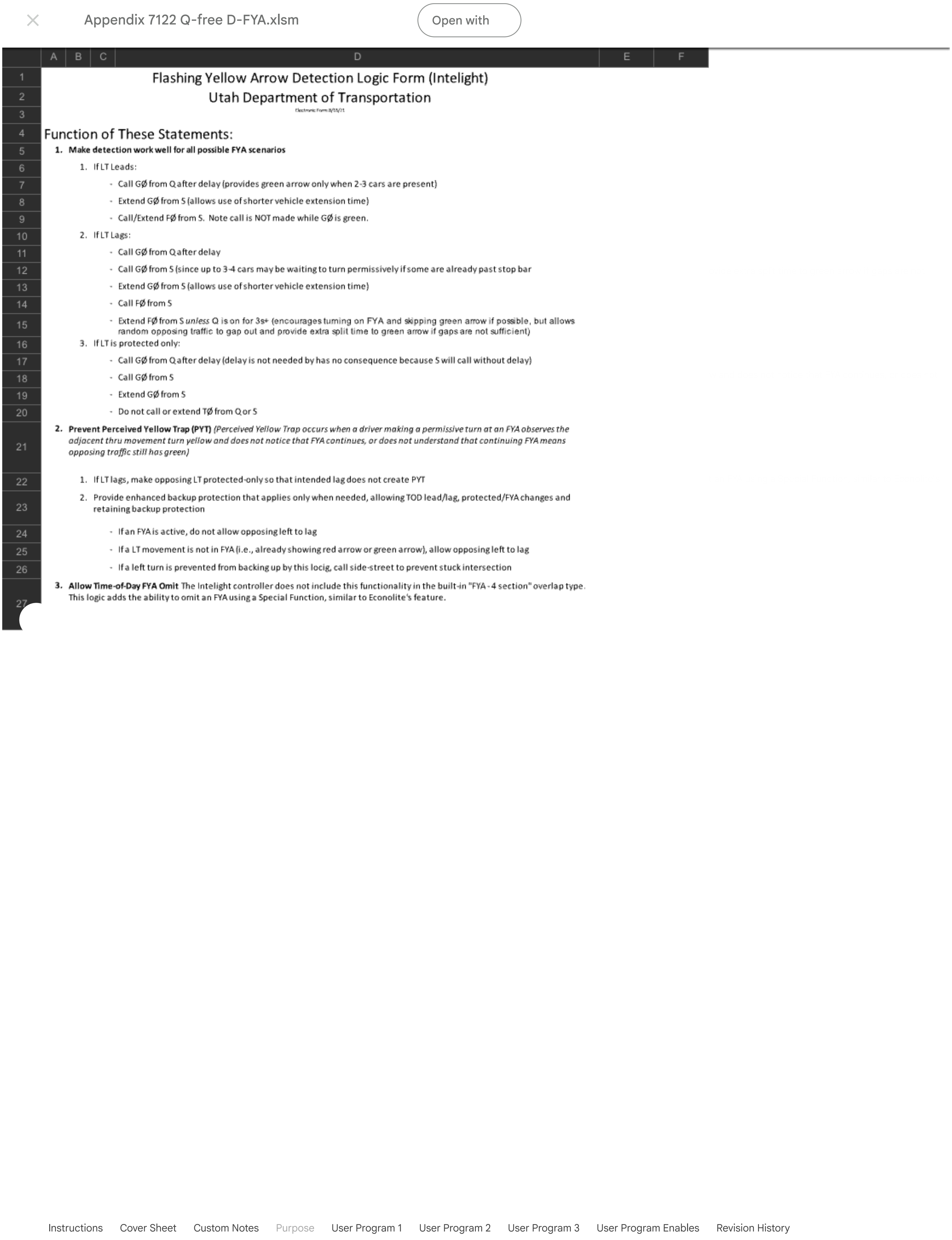
- Duration of this period is settable under Controller->Overlap Configuration->Overlaps->FYA Ped Delay. Don't set longer than or equal to walk+ped clear or FYA may not serve at all, and detection logic does not account for that case.

There are some side-effects to this operation. Notably, to provide this safety benefit for every pedestrian, ped6 cannot be served once Ovl-3 (NBL) is already showing an FYA. (Technically the ped could be reserved without terminating FYA, but this would not provide the protected interval for that ped).

This means that:

- Ped6 cannot rest in walk during coordination
- Ped6 cannot be on "ped recycle" (other peds at this site are on ped recycle).
- Without ped6 on "ped recycle," a ped call would not be served until there is a call on the side street. Accordingly, logic is added to place a side-street call when this is in effect.

Instructions Cover Sheet Custom Notes Purpose User Program 1 User Program 2 User Program 3 User Program Enables Revision History



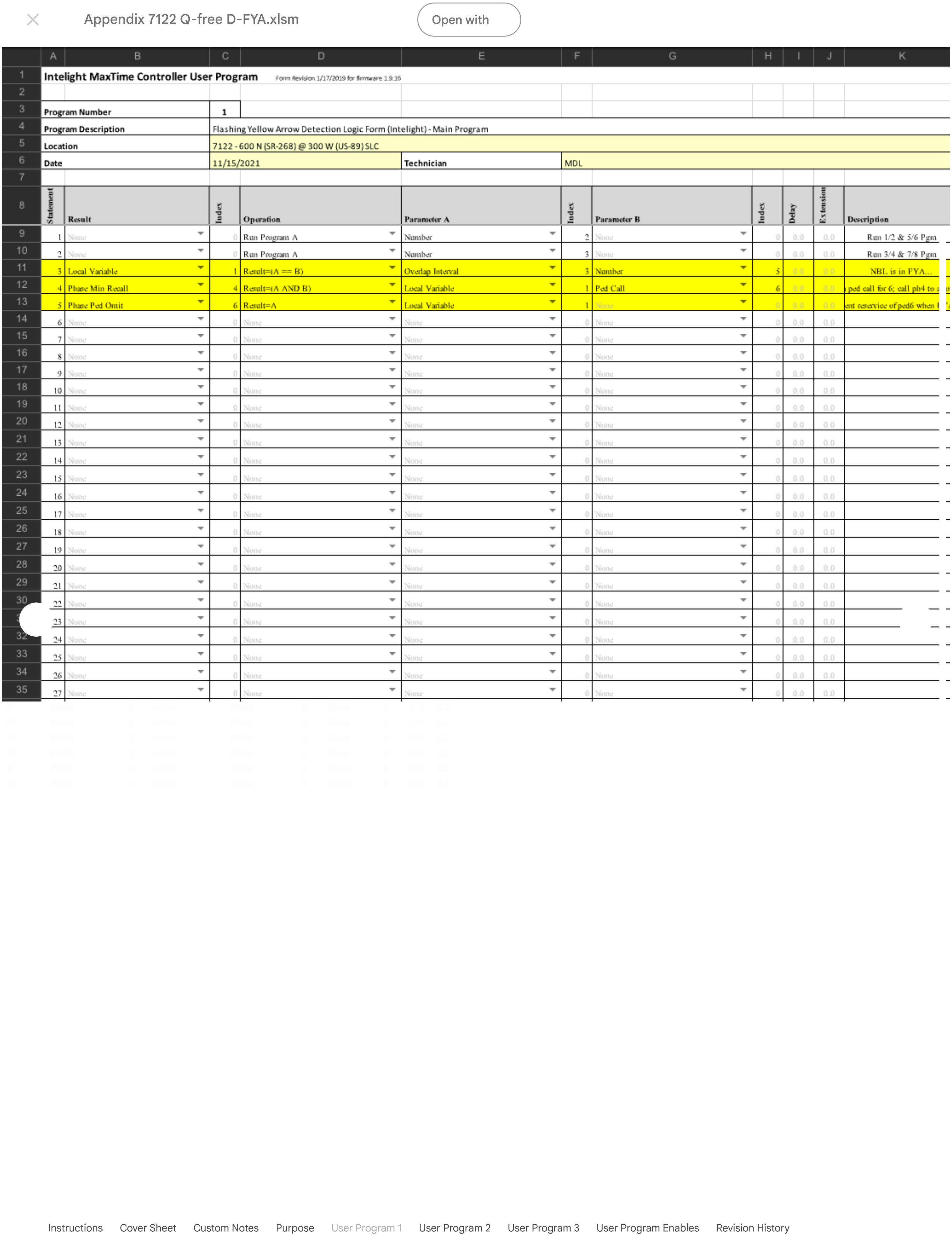
Flashing Yellow Arrow Detection Logic Form (Intelight)

Utah Department of Transportation

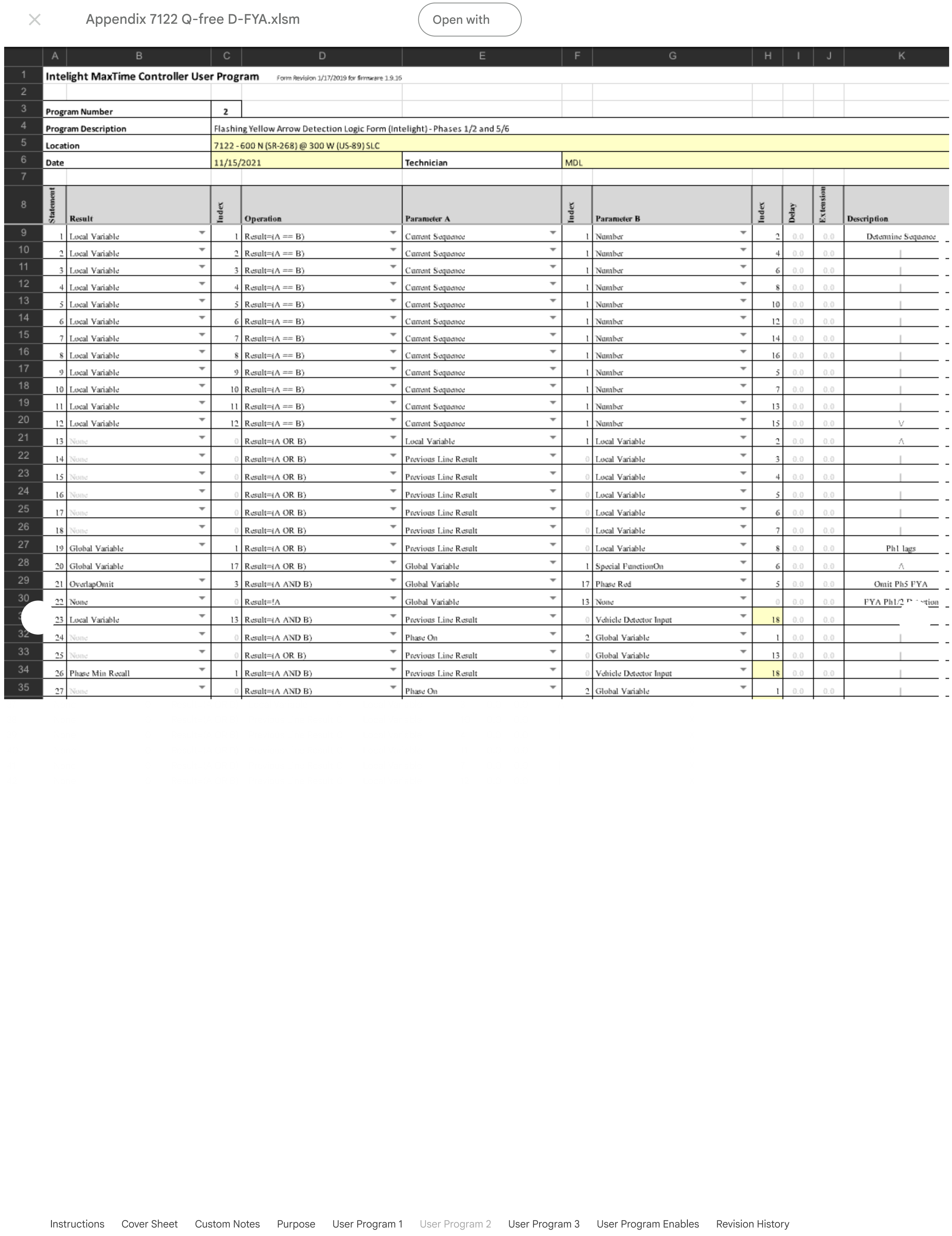
Electronic Form 8/13/21

Function of These Statements:

1. **Make detection work well for all possible FYA scenarios**
    1. If LT Leads:
        - Call GØ from Q after delay (provides green arrow only when 2-3 cars are present)
        - Extend GØ from 5 (allows use of shorter vehicle extension time)
        - Call/Extend FØ from 5. Note call is NOT made while GØ is green.
    2. If LT Lags:
        - Call GØ from Q after delay
        - Call GØ from 5 (since up to 3-4 cars may be waiting to turn permissively if some are already past stop bar
        - Extend GØ from 5 (allows use of shorter vehicle extension time)
        - Call FØ from 5
        - Extend FØ from 5 *unless* Q is on for 3s+ (encourages tuming on FYA and skipping green arrow if possible, but allows random opposing traffic to gap out and provide extra split time to green arrow if gaps are not sufficient)
    3. If LT is protected only:
        - Call GØ from Q after delay (delay is not needed by has no consequence because 5 will call without delay)
        - Call GØ from 5
        - Extend GØ from 5
        - Do not call or extend TØ from Q or 5
2. **Prevent Perceived Yellow Trap (PYT)** *(Perceived Yellow Trap occurs when a driver making a permissive turn at an FYA observes the adjacent thru movement turn yellow and does not notice that FYA continues, or does not understand that continuing FYA means opposing traffic still has green)*
    1. If LT lags, make opposing LT protected-only so that intended lag does not create PYT
    2. Provide enhanced backup protection that applies only when needed, allowing TOD lead/lag, protected/FYA changes and retaining backup protection
        - If an FYA is active, do not allow opposing left to lag
        - If a LT movement is not in FYA (i.e., already showing red arrow or green arrow), allow opposing left to lag
        - If a left turn is prevented from backing up by this locig, call side-street to prevent stuck intersection
3. **Allow Time-of-Day FYA Omit** The Intelight controller does not include this functionality in the built-in "FYA - 4 section" overlap type. This logic adds the ability to omit an FYA using a Special Function, similar to Econolite's feature.



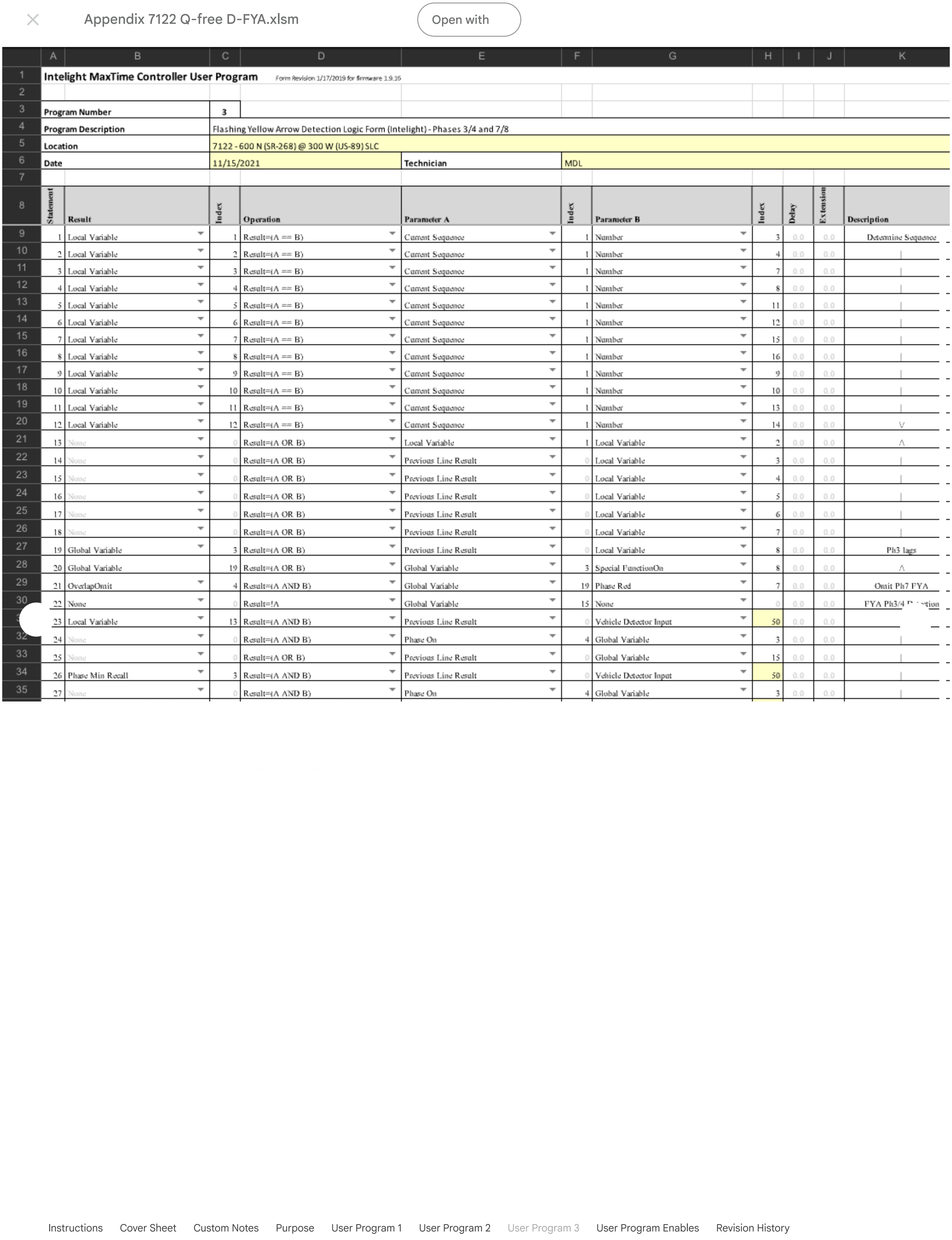
	A	B	C	D	E	F	G	H	I	J	K
1	**Intelight MaxTime Controller User Program**			Form Revision 1/17/2019 for firmware 1.9.16							
2											
3	**Program Number**		**1**								
4	**Program Description**		Flashing Yellow Arrow Detection Logic Form (Intelight) - Main Program								
5	**Location**		7122 - 600 N (SR-268) @ 300 W (US-89) SLC								
6	**Date**		11/15/2021		**Technician**	MDL					
7											
8	**Statement**	**Result**	**Index**	**Operation**	**Parameter A**	**Index**	**Parameter B**	**Index**	**Delay**	**Extension**	**Description**
9	1	None	0	Run Program A	Number	2	None	0	0.0	0.0	Run 1/2 & 5/6 Pgm
10	2	None	0	Run Program A	Number	3	None	0	0.0	0.0	Run 3/4 & 7/8 Pgm
11	3	Local Variable	1	Result=(A == B)	Overlap Interval	3	Number	5	0.0	0.0	NBL is in FYA...
12	4	Phase Min Recall	4	Result=(A AND B)	Local Variable	1	Ped Call	6	0.0	0.0	ped call for 6; call ph4 to a
13	5	Phase Ped Omit	6	Result=A	Local Variable	1	None	0	0.0	0.0	ent reservice of ped6 when F
14	6	None	0	None	None	0	None	0	0.0	0.0	
15	7	None	0	None	None	0	None	0	0.0	0.0	
16	8	None	0	None	None	0	None	0	0.0	0.0	
17	9	None	0	None	None	0	None	0	0.0	0.0	
18	10	None	0	None	None	0	None	0	0.0	0.0	
19	11	None	0	None	None	0	None	0	0.0	0.0	
20	12	None	0	None	None	0	None	0	0.0	0.0	
21	13	None	0	None	None	0	None	0	0.0	0.0	
22	14	None	0	None	None	0	None	0	0.0	0.0	
23	15	None	0	None	None	0	None	0	0.0	0.0	
24	16	None	0	None	None	0	None	0	0.0	0.0	
25	17	None	0	None	None	0	None	0	0.0	0.0	
26	18	None	0	None	None	0	None	0	0.0	0.0	
27	19	None	0	None	None	0	None	0	0.0	0.0	
28	20	None	0	None	None	0	None	0	0.0	0.0	
29	21	None	0	None	None	0	None	0	0.0	0.0	
30	22	None	0	None	None	0	None	0	0.0	0.0	
31	23	None	0	None	None	0	None	0	0.0	0.0	
32	24	None	0	None	None	0	None	0	0.0	0.0	
33	25	None	0	None	None	0	None	0	0.0	0.0	
34	26	None	0	None	None	0	None	0	0.0	0.0	
35	27	None	0	None	None	0	None	0	0.0	0.0	



Intelight MaxTime Controller User Program Form Revision 1/17/2019 for firmware 1.9.16

Program Number	2
Program Description	Flashing Yellow Arrow Detection Logic Form (Intelight) - Phases 1/2 and 5/6
Location	7122 - 600 N (SR-268) @ 300 W (US-89) SLC
Date	11/15/2021
Technician	MDL

	Statement	Result	Index	Operation	Parameter A	Index	Parameter B	Index	Delay	Extension	Description
9	1	Local Variable	1	Result=(A == B)	Current Sequence	1	Number	2	0.0	0.0	Determine Sequence
10	2	Local Variable	2	Result=(A == B)	Current Sequence	1	Number	4	0.0	0.0	|
11	3	Local Variable	3	Result=(A == B)	Current Sequence	1	Number	6	0.0	0.0	|
12	4	Local Variable	4	Result=(A == B)	Current Sequence	1	Number	8	0.0	0.0	|
13	5	Local Variable	5	Result=(A == B)	Current Sequence	1	Number	10	0.0	0.0	|
14	6	Local Variable	6	Result=(A == B)	Current Sequence	1	Number	12	0.0	0.0	|
15	7	Local Variable	7	Result=(A == B)	Current Sequence	1	Number	14	0.0	0.0	|
16	8	Local Variable	8	Result=(A == B)	Current Sequence	1	Number	16	0.0	0.0	|
17	9	Local Variable	9	Result=(A == B)	Current Sequence	1	Number	5	0.0	0.0	|
18	10	Local Variable	10	Result=(A == B)	Current Sequence	1	Number	7	0.0	0.0	|
19	11	Local Variable	11	Result=(A == B)	Current Sequence	1	Number	13	0.0	0.0	|
20	12	Local Variable	12	Result=(A == B)	Current Sequence	1	Number	15	0.0	0.0	V
21	13	None	0	Result=(A OR B)	Local Variable	1	Local Variable	2	0.0	0.0	Λ
22	14	None	0	Result=(A OR B)	Previous Line Result	0	Local Variable	3	0.0	0.0	|
23	15	None	0	Result=(A OR B)	Previous Line Result	0	Local Variable	4	0.0	0.0	|
24	16	None	0	Result=(A OR B)	Previous Line Result	0	Local Variable	5	0.0	0.0	|
25	17	None	0	Result=(A OR B)	Previous Line Result	0	Local Variable	6	0.0	0.0	|
26	18	None	0	Result=(A OR B)	Previous Line Result	0	Local Variable	7	0.0	0.0	|
27	19	Global Variable	1	Result=(A OR B)	Previous Line Result	0	Local Variable	8	0.0	0.0	Ph1 lags
28	20	Global Variable	17	Result=(A OR B)	Global Variable	1	Special FunctionOn	6	0.0	0.0	Λ
29	21	OverlapOmit	3	Result=(A AND B)	Global Variable	17	Phase Red	5	0.0	0.0	Omit Ph5 FYA
30	22	None	0	Result=!A	Global Variable	13	None	0	0.0	0.0	FYA Ph1/2 Detection
	23	Local Variable	13	Result=(A AND B)	Previous Line Result	0	Vehicle Detector Input	18	0.0	0.0	
32	24	None	0	Result=(A AND B)	Phase On	2	Global Variable	1	0.0	0.0	|
33	25	None	0	Result=(A OR B)	Previous Line Result	0	Global Variable	13	0.0	0.0	|
34	26	Phase Min Recall	1	Result=(A AND B)	Previous Line Result	0	Vehicle Detector Input	18	0.0	0.0	|
35	27	None	0	Result=(A AND B)	Phase On	2	Global Variable	1	0.0	0.0	|

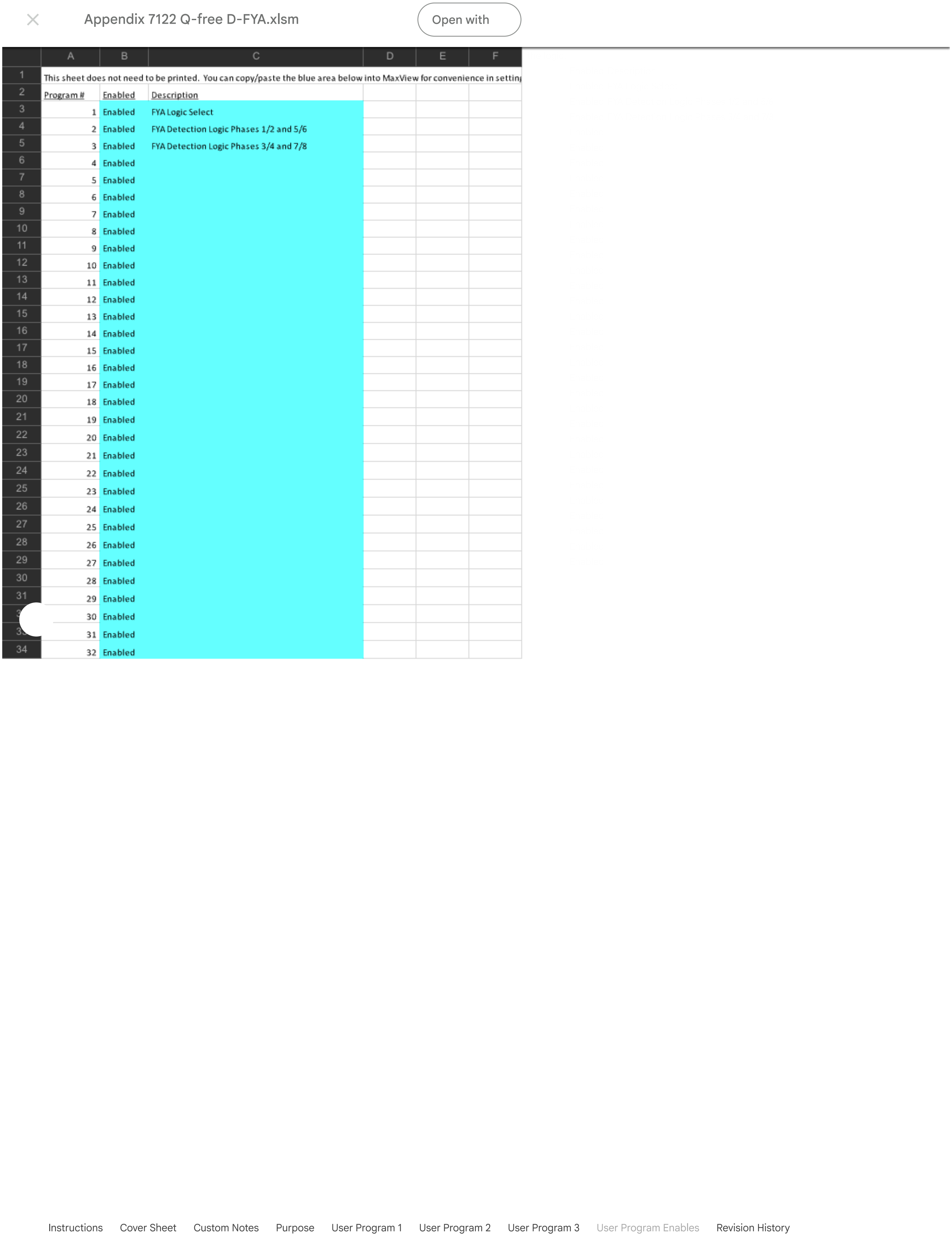


Intelight MaxTime Controller User Program Form Revision 1/17/2019 for firmware 1.9.16

Program Number	3		
Program Description	Flashing Yellow Arrow Detection Logic Form (Intelight) - Phases 3/4 and 7/8		
Location	7122 - 600 N (SR-268) @ 300 W (US-89) SLC		
Date	11/15/2021	**Technician**	MDL

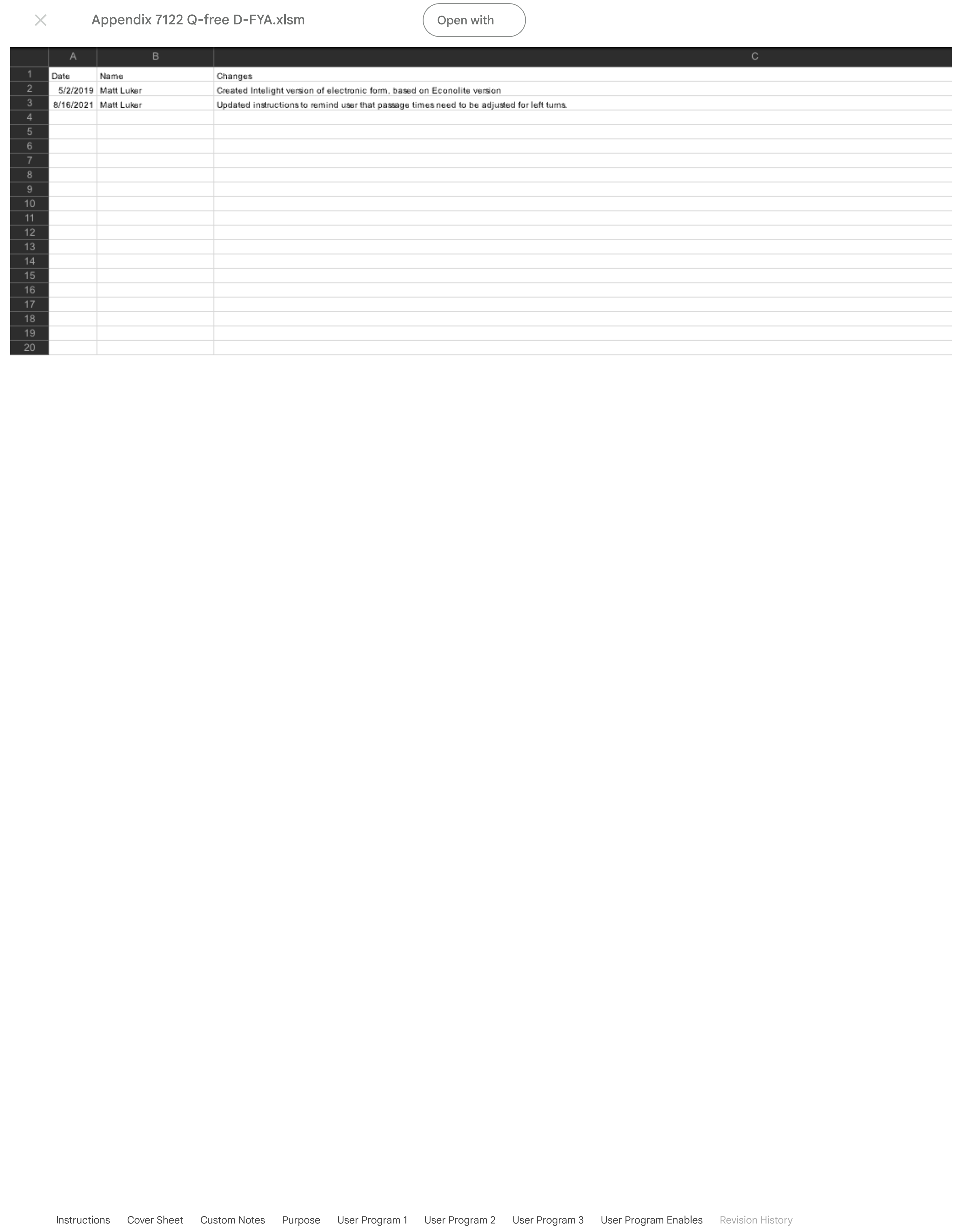
Statement	Result	Index	Operation	Parameter A	Index	Parameter B	Index	Delay	Extension	Description
1	Local Variable	1	Result=(A == B)	Current Sequence	1	Number	3	0.0	0.0	Determine Sequence
2	Local Variable	2	Result=(A == B)	Current Sequence	1	Number	4	0.0	0.0	|
3	Local Variable	3	Result=(A == B)	Current Sequence	1	Number	7	0.0	0.0	|
4	Local Variable	4	Result=(A == B)	Current Sequence	1	Number	8	0.0	0.0	|
5	Local Variable	5	Result=(A == B)	Current Sequence	1	Number	11	0.0	0.0	|
6	Local Variable	6	Result=(A == B)	Current Sequence	1	Number	12	0.0	0.0	|
7	Local Variable	7	Result=(A == B)	Current Sequence	1	Number	15	0.0	0.0	|
8	Local Variable	8	Result=(A == B)	Current Sequence	1	Number	16	0.0	0.0	|
9	Local Variable	9	Result=(A == B)	Current Sequence	1	Number	9	0.0	0.0	|
10	Local Variable	10	Result=(A == B)	Current Sequence	1	Number	10	0.0	0.0	|
11	Local Variable	11	Result=(A == B)	Current Sequence	1	Number	13	0.0	0.0	|
12	Local Variable	12	Result=(A == B)	Current Sequence	1	Number	14	0.0	0.0	V
13	None	0	Result=(A OR B)	Local Variable	1	Local Variable	2	0.0	0.0	Λ
14	None	0	Result=(A OR B)	Previous Line Result	0	Local Variable	3	0.0	0.0	|
15	None	0	Result=(A OR B)	Previous Line Result	0	Local Variable	4	0.0	0.0	|
16	None	0	Result=(A OR B)	Previous Line Result	0	Local Variable	5	0.0	0.0	|
17	None	0	Result=(A OR B)	Previous Line Result	0	Local Variable	6	0.0	0.0	|
18	None	0	Result=(A OR B)	Previous Line Result	0	Local Variable	7	0.0	0.0	|
19	Global Variable	3	Result=(A OR B)	Previous Line Result	0	Local Variable	8	0.0	0.0	Ph3 lags
20	Global Variable	19	Result=(A OR B)	Global Variable	3	Special FunctionOn	8	0.0	0.0	Λ
21	OverlapOmit	4	Result=(A AND B)	Global Variable	19	Phase Red	7	0.0	0.0	Omit Ph7 FYA
22	None	0	Result=!A	Global Variable	15	None	0	0.0	0.0	FYA Ph3/4 [illegible]tion
23	Local Variable	13	Result=(A AND B)	Previous Line Result	0	Vehicle Detector Input	50	0.0	0.0	
24	None	0	Result=(A AND B)	Phase On	4	Global Variable	3	0.0	0.0	|
25	None	0	Result=(A OR B)	Previous Line Result	0	Global Variable	15	0.0	0.0	|
26	Phase Min Recall	3	Result=(A AND B)	Previous Line Result	0	Vehicle Detector Input	50	0.0	0.0	|
27	None	0	Result=(A AND B)	Phase On	4	Global Variable	3	0.0	0.0	|

Instructions Cover Sheet Custom Notes Purpose User Program 1 User Program 2 User Program 3 User Program Enables Revision History

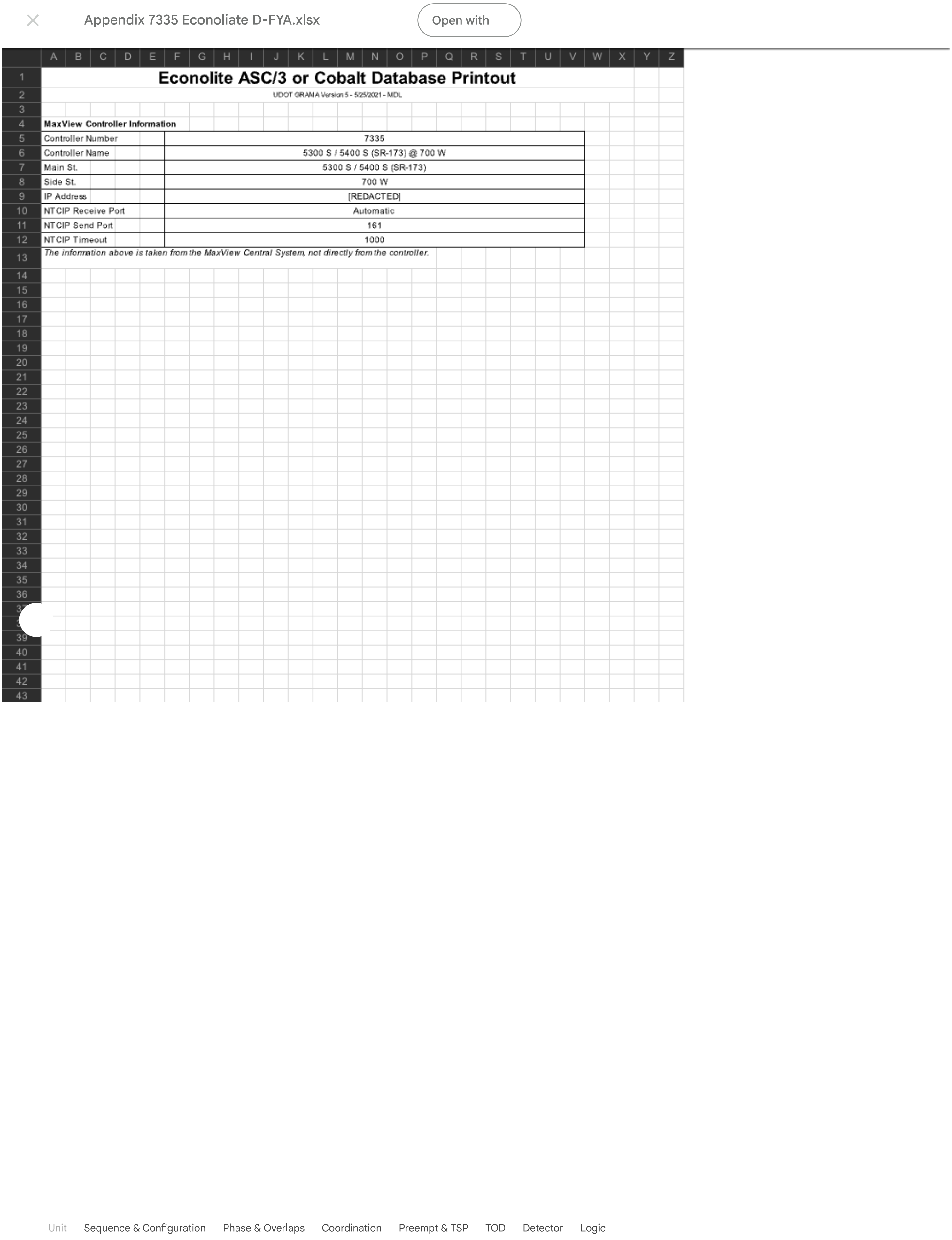


Appendix 7122 Q-free D-FYA.xlsm

	A	B	C	D	E	F
1	This sheet does not need to be printed. You can copy/paste the blue area below into MaxView for convenience in settin					
2	Program #	Enabled	Description			
3	1	Enabled	FYA Logic Select			
4	2	Enabled	FYA Detection Logic Phases 1/2 and 5/6			
5	3	Enabled	FYA Detection Logic Phases 3/4 and 7/8			
6	4	Enabled				
7	5	Enabled				
8	6	Enabled				
9	7	Enabled				
10	8	Enabled				
11	9	Enabled				
12	10	Enabled				
13	11	Enabled				
14	12	Enabled				
15	13	Enabled				
16	14	Enabled				
17	15	Enabled				
18	16	Enabled				
19	17	Enabled				
20	18	Enabled				
21	19	Enabled				
22	20	Enabled				
23	21	Enabled				
24	22	Enabled				
25	23	Enabled				
26	24	Enabled				
27	25	Enabled				
28	26	Enabled				
29	27	Enabled				
30	28	Enabled				
31	29	Enabled				
32	30	Enabled				
33	31	Enabled				
34	32	Enabled				



	A	B	C
1	Date	Name	Changes
2	5/2/2019	Matt Luker	Created Intelight version of electronic form, based on Econolite version
3	8/16/2021	Matt Luker	Updated instructions to remind user that passage times need to be adjusted for left turns.
4			
5			
6			
7			
8			
9			
10			
11			
12			
13			
14			
15			
16			
17			
18			
19			
20			



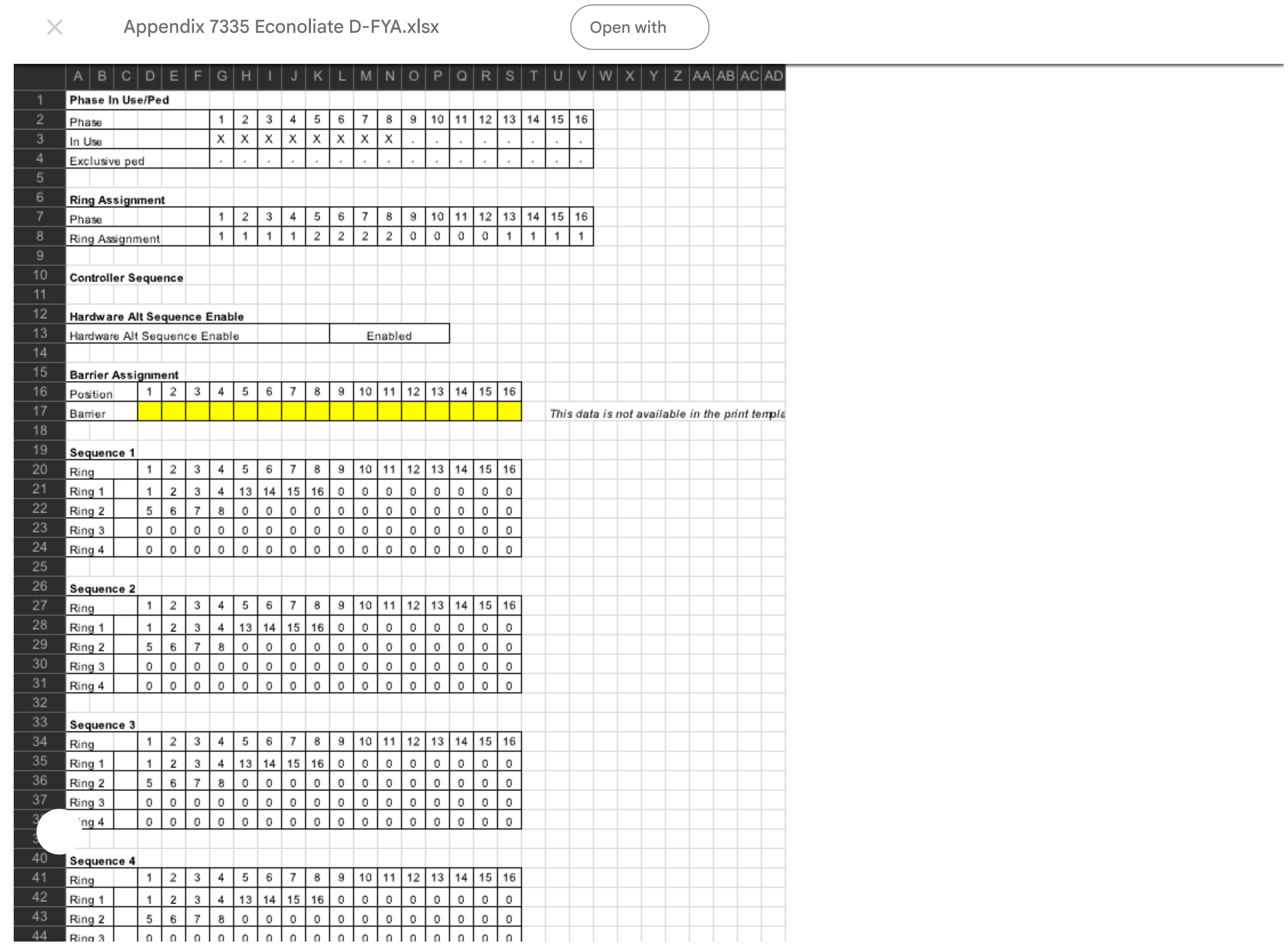
Econolite ASC/3 or Cobalt Database Printout

UDOT GRAMA Version 5 - 5/25/2021 - MDL

MaxView Controller Information

Controller Number	7335
Controller Name	5300 S / 5400 S (SR-173) @ 700 W
Main St.	5300 S / 5400 S (SR-173)
Side St.	700 W
IP Address	[REDACTED]
NTCIP Receive Port	Automatic
NTCIP Send Port	161
NTCIP Timeout	1000

The information above is taken from the MaxView Central System, not directly from the controller.



Appendix 7335 Econoliate D-FYA.xlsx

Phase In Use/Ped

Phase	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
In Use	X	X	X	X	X	X	X	X	.	.	.	.	.	.	.	.
Exclusive ped	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.

Ring Assignment

Phase	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Ring Assignment	1	1	1	1	2	2	2	2	0	0	0	0	1	1	1	1

Controller Sequence

Hardware Alt Sequence Enable

Hardware Alt Sequence Enable	Enabled

Barrier Assignment

Position	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Barrier																

This data is not available in the print templa

Sequence 1

Ring	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Ring 1	1	2	3	4	13	14	15	16	0	0	0	0	0	0	0	0
Ring 2	5	6	7	8	0	0	0	0	0	0	0	0	0	0	0	0
Ring 3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Ring 4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Sequence 2

Ring	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Ring 1	1	2	3	4	13	14	15	16	0	0	0	0	0	0	0	0
Ring 2	5	6	7	8	0	0	0	0	0	0	0	0	0	0	0	0
Ring 3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Ring 4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

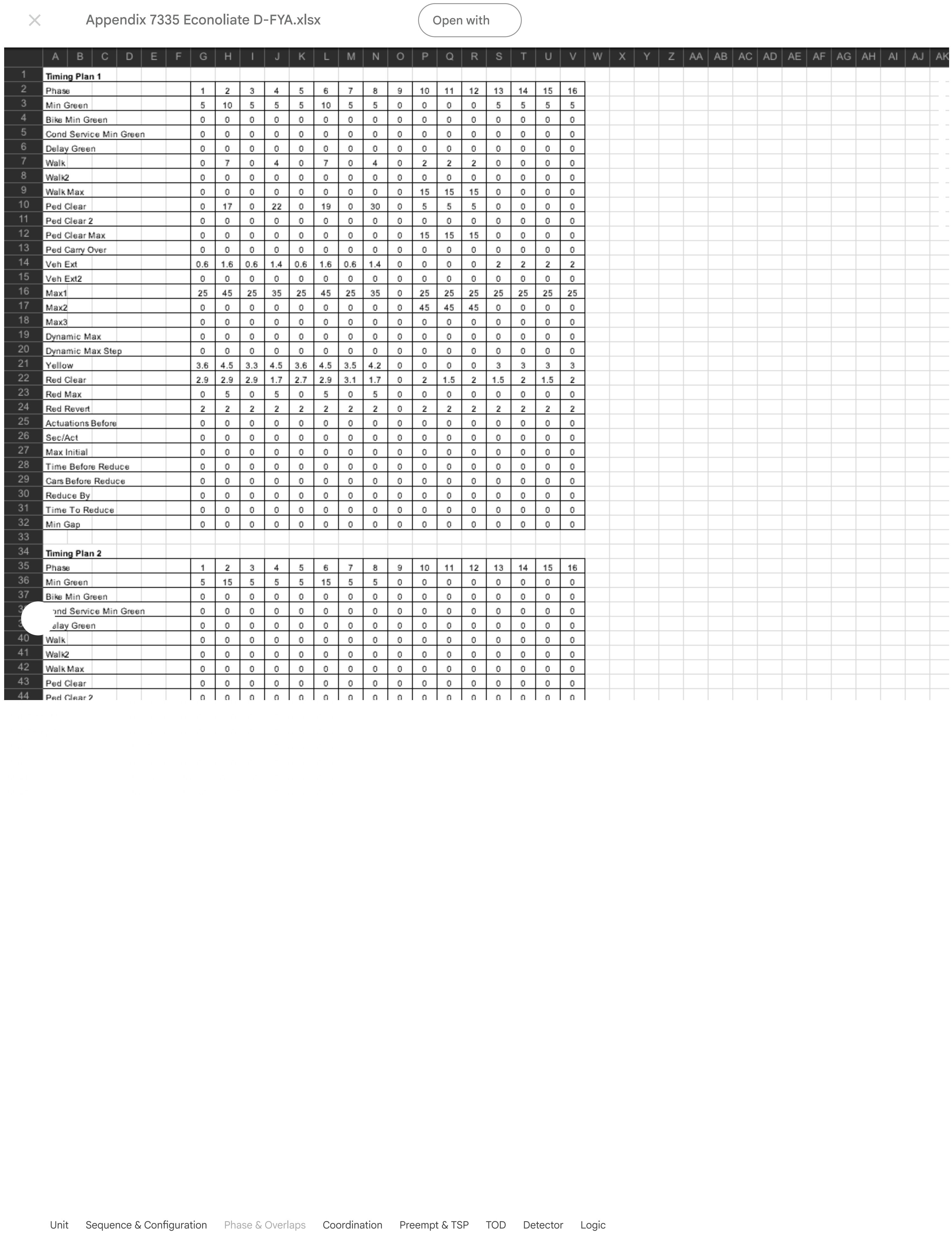
Sequence 3

Ring	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Ring 1	1	2	3	4	13	14	15	16	0	0	0	0	0	0	0	0
Ring 2	5	6	7	8	0	0	0	0	0	0	0	0	0	0	0	0
Ring 3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Ring 4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Sequence 4

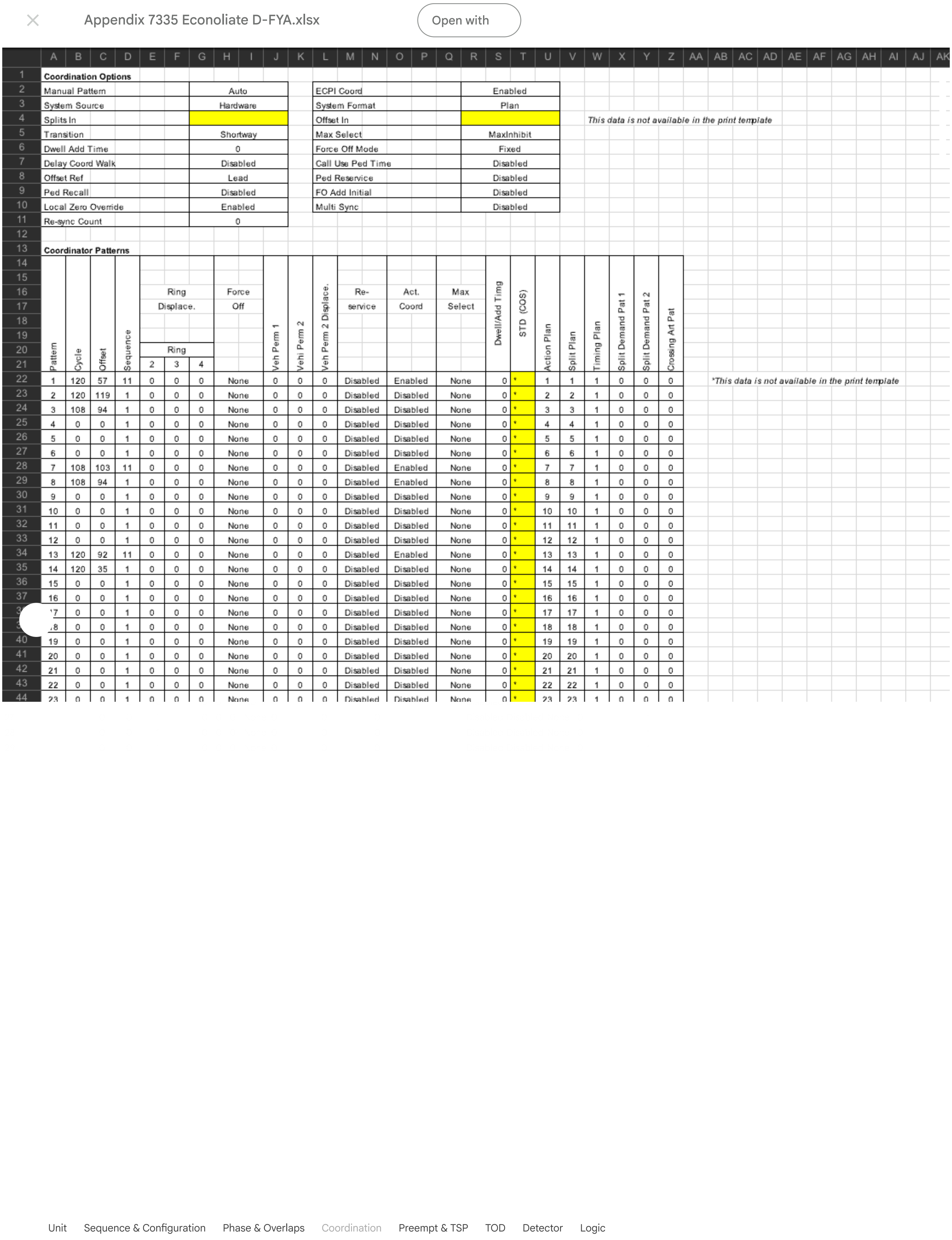
Ring	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Ring 1	1	2	3	4	13	14	15	16	0	0	0	0	0	0	0	0
Ring 2	5	6	7	8	0	0	0	0	0	0	0	0	0	0	0	0
Ring 3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Unit Sequence & Configuration Phase & Overlaps Coordination Preempt & TSP TOD Detector Logic



Appendix 7335 Econoliate D-FYA.xlsx

	A	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
1	**Timing Plan 1**																
2	Phase	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
3	Min Green	5	10	5	5	5	10	5	5	0	0	0	0	5	5	5	5
4	Bike Min Green	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	Cond Service Min Green	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	Delay Green	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	Walk	0	7	0	4	0	7	0	4	0	2	2	2	0	0	0	0
8	Walk2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	Walk Max	0	0	0	0	0	0	0	0	0	15	15	15	0	0	0	0
10	Ped Clear	0	17	0	22	0	19	0	30	0	5	5	5	0	0	0	0
11	Ped Clear 2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	Ped Clear Max	0	0	0	0	0	0	0	0	0	15	15	15	0	0	0	0
13	Ped Carry Over	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14	Veh Ext	0.6	1.6	0.6	1.4	0.6	1.6	0.6	1.4	0	0	0	0	2	2	2	2
15	Veh Ext2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	Max1	25	45	25	35	25	45	25	35	0	25	25	25	25	25	25	25
17	Max2	0	0	0	0	0	0	0	0	0	45	45	45	0	0	0	0
18	Max3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
19	Dynamic Max	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20	Dynamic Max Step	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
21	Yellow	3.6	4.5	3.3	4.5	3.6	4.5	3.5	4.2	0	0	0	0	3	3	3	3
22	Red Clear	2.9	2.9	2.9	1.7	2.7	2.9	3.1	1.7	0	2	1.5	2	1.5	2	1.5	2
23	Red Max	0	5	0	5	0	5	0	5	0	0	0	0	0	0	0	0
24	Red Revert	2	2	2	2	2	2	2	2	0	2	2	2	2	2	2	2
25	Actuations Before	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
26	Sec/Act	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
27	Max Initial	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
28	Time Before Reduce	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
29	Cars Before Reduce	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
30	Reduce By	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
31	Time To Reduce	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
32	Min Gap	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
33																	
34	**Timing Plan 2**																
35	Phase	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
36	Min Green	5	15	5	5	5	15	5	5	0	0	0	0	0	0	0	0
37	Bike Min Green	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	ond Service Min Green	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	elay Green	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
40	Walk	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
41	Walk2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
42	Walk Max	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
43	Ped Clear	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
44	Ped Clear 2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0



Coordination Options

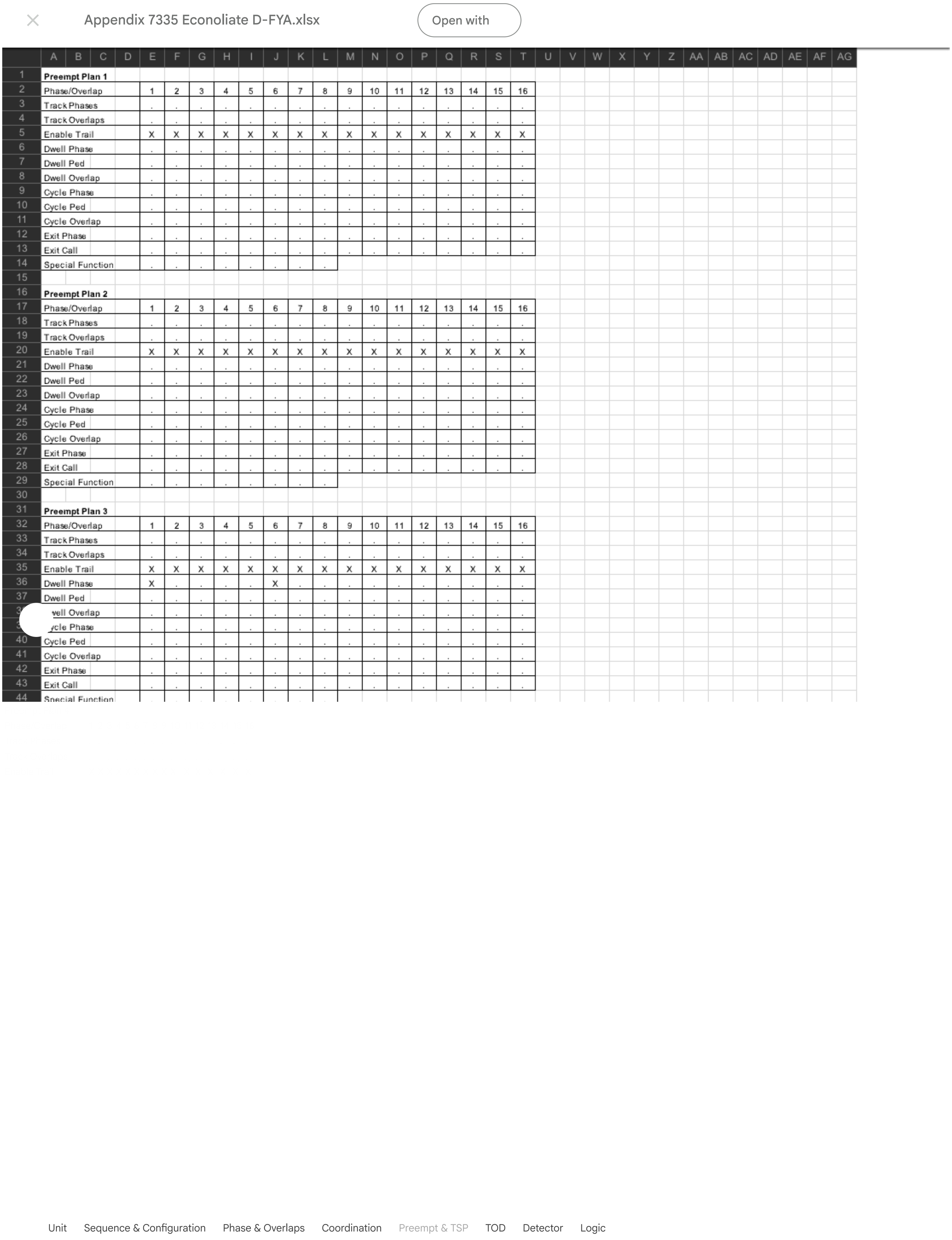
Manual Pattern	Auto	ECPI Coord	Enabled
System Source	Hardware	System Format	Plan
Splits In		Offset In	
Transition	Shortway	Max Select	MaxInhibit
Dwell Add Time	0	Force Off Mode	Fixed
Delay Coord Walk	Disabled	Call Use Ped Time	Disabled
Offset Ref	Lead	Ped Reservice	Disabled
Ped Recall	Disabled	FO Add Initial	Disabled
Local Zero Override	Enabled	Multi Sync	Disabled
Re-sync Count	0		

This data is not available in the print template

Coordinator Patterns

Pattern	Cycle	Offset	Sequence	Ring Displace. Ring			Force Off	Veh Perm 1	Vehi Perm 2	Veh Perm 2 Displace.	Re-service	Act. Coord	Max Select	Dwell/Add Timg	STD (COS)	Action Plan	Split Plan	Timing Plan	Split Demand Pat 1	Split Demand Pat 2	Crossing Art Pat
				2	3	4															
1	120	57	11	0	0	0	None	0	0	0	Disabled	Enabled	None	0	*	1	1	1	0	0	0
2	120	119	1	0	0	0	None	0	0	0	Disabled	Disabled	None	0	*	2	2	1	0	0	0
3	108	94	1	0	0	0	None	0	0	0	Disabled	Disabled	None	0	*	3	3	1	0	0	0
4	0	0	1	0	0	0	None	0	0	0	Disabled	Disabled	None	0	*	4	4	1	0	0	0
5	0	0	1	0	0	0	None	0	0	0	Disabled	Disabled	None	0	*	5	5	1	0	0	0
6	0	0	1	0	0	0	None	0	0	0	Disabled	Disabled	None	0	*	6	6	1	0	0	0
7	108	103	11	0	0	0	None	0	0	0	Disabled	Enabled	None	0	*	7	7	1	0	0	0
8	108	94	1	0	0	0	None	0	0	0	Disabled	Enabled	None	0	*	8	8	1	0	0	0
9	0	0	1	0	0	0	None	0	0	0	Disabled	Disabled	None	0	*	9	9	1	0	0	0
10	0	0	1	0	0	0	None	0	0	0	Disabled	Disabled	None	0	*	10	10	1	0	0	0
11	0	0	1	0	0	0	None	0	0	0	Disabled	Disabled	None	0	*	11	11	1	0	0	0
12	0	0	1	0	0	0	None	0	0	0	Disabled	Disabled	None	0	*	12	12	1	0	0	0
13	120	92	11	0	0	0	None	0	0	0	Disabled	Enabled	None	0	*	13	13	1	0	0	0
14	120	35	1	0	0	0	None	0	0	0	Disabled	Disabled	None	0	*	14	14	1	0	0	0
15	0	0	1	0	0	0	None	0	0	0	Disabled	Disabled	None	0	*	15	15	1	0	0	0
16	0	0	1	0	0	0	None	0	0	0	Disabled	Disabled	None	0	*	16	16	1	0	0	0
17	0	0	1	0	0	0	None	0	0	0	Disabled	Disabled	None	0	*	17	17	1	0	0	0
18	0	0	1	0	0	0	None	0	0	0	Disabled	Disabled	None	0	*	18	18	1	0	0	0
19	0	0	1	0	0	0	None	0	0	0	Disabled	Disabled	None	0	*	19	19	1	0	0	0
20	0	0	1	0	0	0	None	0	0	0	Disabled	Disabled	None	0	*	20	20	1	0	0	0
21	0	0	1	0	0	0	None	0	0	0	Disabled	Disabled	None	0	*	21	21	1	0	0	0
22	0	0	1	0	0	0	None	0	0	0	Disabled	Disabled	None	0	*	22	22	1	0	0	0
23	0	0	1	0	0	0	None	0	0	0	Disabled	Disabled	None	0	*	23	23	1	0	0	0

**This data is not available in the print template*



Appendix 7335 Econoliate D-FYA.xlsx

Preempt Plan 1

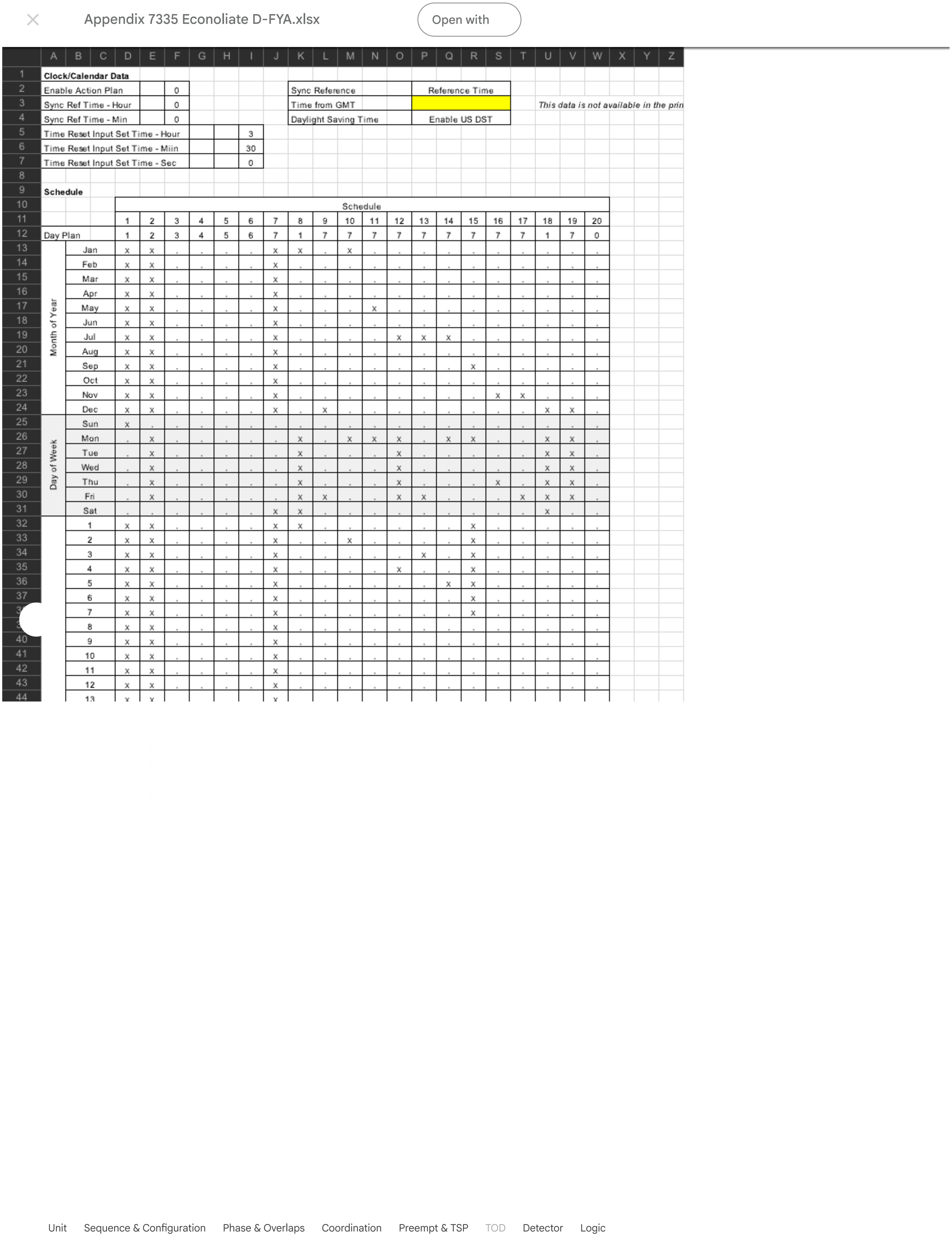
Phase/Overlap	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Track Phases	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
Track Overlaps	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
Enable Trail	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
Dwell Phase	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
Dwell Ped	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
Dwell Overlap	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
Cycle Phase	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
Cycle Ped	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
Cycle Overlap	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
Exit Phase	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
Exit Call	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
Special Function	.	.	.	.	.	.	.	.								

Preempt Plan 2

Phase/Overlap	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Track Phases	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
Track Overlaps	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
Enable Trail	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
Dwell Phase	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
Dwell Ped	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
Dwell Overlap	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
Cycle Phase	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
Cycle Ped	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
Cycle Overlap	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
Exit Phase	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
Exit Call	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
Special Function	.	.	.	.	.	.	.	.								

Preempt Plan 3

Phase/Overlap	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Track Phases	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
Track Overlaps	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
Enable Trail	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
Dwell Phase	X	.	.	.	.	X	.	.	.	.	.	.	.	.	.	.
Dwell Ped	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
Dwell Overlap	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
Cycle Phase	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
Cycle Ped	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
Cycle Overlap	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
Exit Phase	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
Exit Call	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
Special Function	.	.	.	.	.	.	.	.								

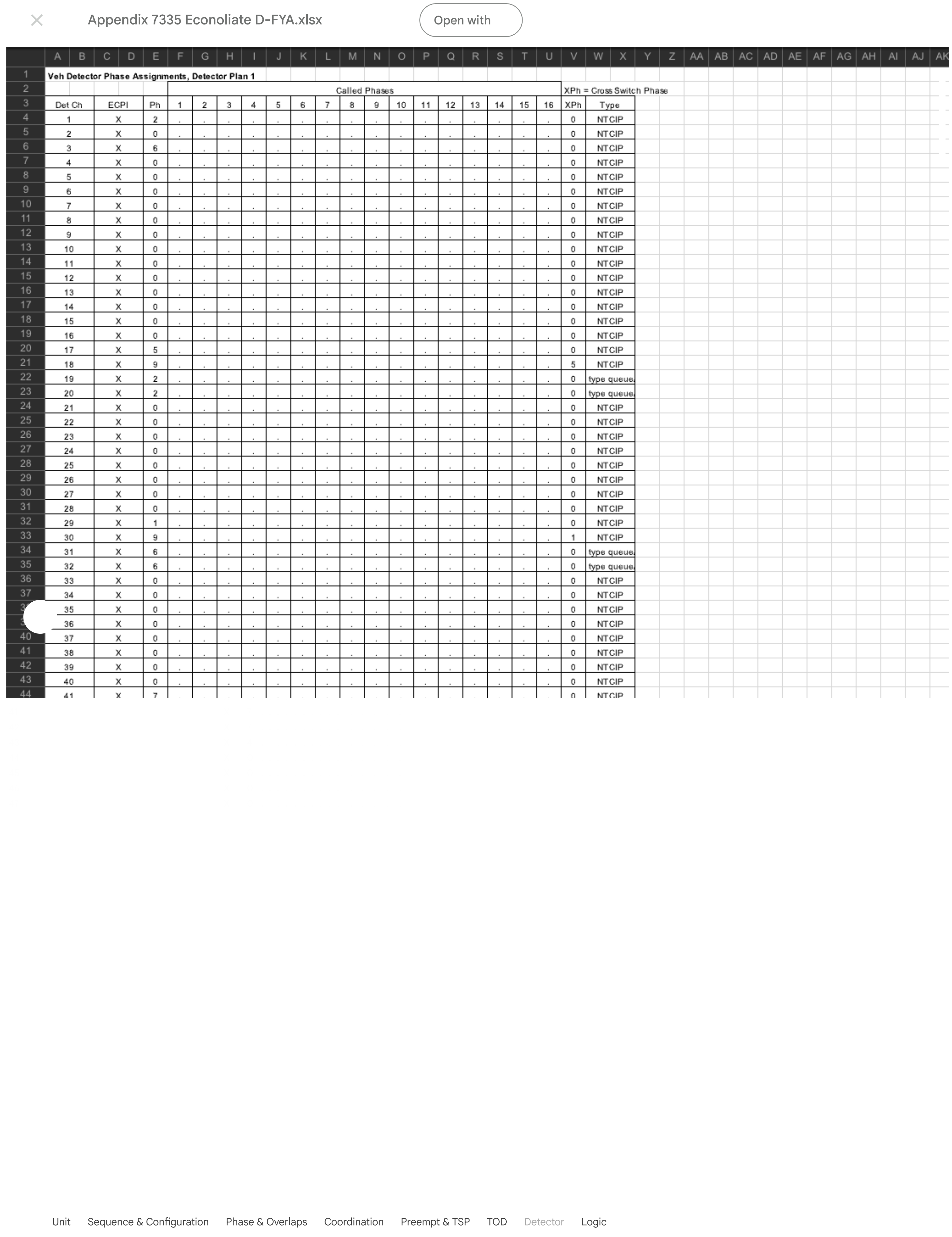


Appendix 7335 Econoliate D-FYA.xlsx

Clock/Calendar Data				
Enable Action Plan	0	Sync Reference	Reference Time	
Sync Ref Time - Hour	0	Time from GMT		*This data is not available in the prin*
Sync Ref Time - Min	0	Daylight Saving Time	Enable US DST	
Time Reset Input Set Time - Hour	3			
Time Reset Input Set Time - Miin	30			
Time Reset Input Set Time - Sec	0			

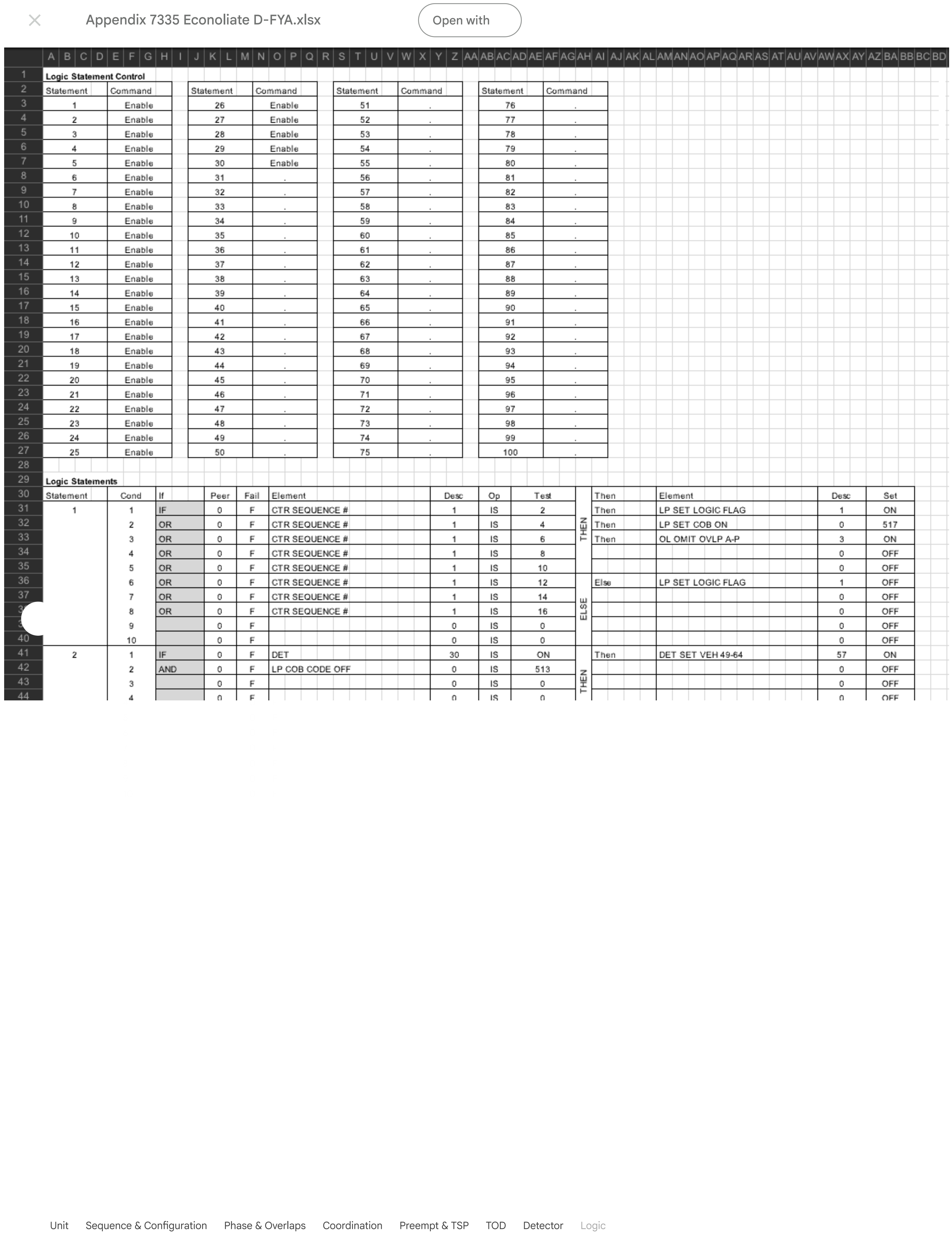
Schedule

		Schedule																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Day Plan		1	2	3	4	5	6	7	1	7	7	7	7	7	7	7	7	7	1	7	0
Month of Year	Jan	x	x	.	.	.	.	x	x	.	x	.	.	.	.	.	.	.	.	.	.
	Feb	x	x	.	.	.	.	x	.	.	.	.	.	.	.	.	.	.	.	.	.
	Mar	x	x	.	.	.	.	x	.	.	.	.	.	.	.	.	.	.	.	.	.
	Apr	x	x	.	.	.	.	x	.	.	.	.	.	.	.	.	.	.	.	.	.
	May	x	x	.	.	.	.	x	.	.	.	x	.	.	.	.	.	.	.	.	.
	Jun	x	x	.	.	.	.	x	.	.	.	.	.	.	.	.	.	.	.	.	.
	Jul	x	x	.	.	.	.	x	.	.	.	.	x	x	x	.	.	.	.	.	.
	Aug	x	x	.	.	.	.	x	.	.	.	.	.	.	.	.	.	.	.	.	.
	Sep	x	x	.	.	.	.	x	.	.	.	.	.	.	.	x	.	.	.	.	.
	Oct	x	x	.	.	.	.	x	.	.	.	.	.	.	.	.	.	.	.	.	.
	Nov	x	x	.	.	.	.	x	.	.	.	.	.	.	.	.	x	x	.	.	.
	Dec	x	x	.	.	.	.	x	.	x	.	.	.	.	.	.	.	.	x	x	.
Day of Week	Sun	x	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
	Mon	.	x	.	.	.	.	.	x	.	x	x	x	.	x	x	.	.	x	x	.
	Tue	.	x	.	.	.	.	.	x	.	.	.	x	.	.	.	.	.	x	x	.
	Wed	.	x	.	.	.	.	.	x	.	.	.	x	.	.	.	.	.	x	x	.
	Thu	.	x	.	.	.	.	.	x	.	.	.	x	.	.	.	x	.	x	x	.
	Fri	.	x	.	.	.	.	.	x	x	.	.	x	x	.	.	.	x	x	x	.
	Sat	.	.	.	.	.	.	x	x	.	.	.	.	.	.	.	.	.	x	.	.
	1	x	x	.	.	.	.	x	x	.	.	.	.	.	.	x	.	.	.	.	.
	2	x	x	.	.	.	.	x	.	.	x	.	.	.	.	x	.	.	.	.	.
	3	x	x	.	.	.	.	x	.	.	.	.	.	x	.	x	.	.	.	.	.
	4	x	x	.	.	.	.	x	.	.	.	.	x	.	.	x	.	.	.	.	.
	5	x	x	.	.	.	.	x	.	.	.	.	.	.	x	x	.	.	.	.	.
	6	x	x	.	.	.	.	x	.	.	.	.	.	.	.	x	.	.	.	.	.
	7	x	x	.	.	.	.	x	.	.	.	.	.	.	.	x	.	.	.	.	.
	8	x	x	.	.	.	.	x	.	.	.	.	.	.	.	.	.	.	.	.	.
	9	x	x	.	.	.	.	x	.	.	.	.	.	.	.	.	.	.	.	.	.
	10	x	x	.	.	.	.	x	.	.	.	.	.	.	.	.	.	.	.	.	.
	11	x	x	.	.	.	.	x	.	.	.	.	.	.	.	.	.	.	.	.	.
	12	x	x	.	.	.	.	x	.	.	.	.	.	.	.	.	.	.	.	.	.
	13	x	x					x													



Veh Detector Phase Assignments, Detector Plan 1

			Called Phases																XPh = Cross Switch Phase	
Det Ch	ECPI	Ph	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	XPh	Type
1	X	2	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	0	NTCIP
2	X	0	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	0	NTCIP
3	X	6	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	0	NTCIP
4	X	0	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	0	NTCIP
5	X	0	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	0	NTCIP
6	X	0	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	0	NTCIP
7	X	0	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	0	NTCIP
8	X	0	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	0	NTCIP
9	X	0	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	0	NTCIP
10	X	0	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	0	NTCIP
11	X	0	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	0	NTCIP
12	X	0	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	0	NTCIP
13	X	0	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	0	NTCIP
14	X	0	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	0	NTCIP
15	X	0	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	0	NTCIP
16	X	0	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	0	NTCIP
17	X	5	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	0	NTCIP
18	X	9	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	5	NTCIP
19	X	2	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	0	type queue
20	X	2	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	0	type queue
21	X	0	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	0	NTCIP
22	X	0	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	0	NTCIP
23	X	0	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	0	NTCIP
24	X	0	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	0	NTCIP
25	X	0	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	0	NTCIP
26	X	0	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	0	NTCIP
27	X	0	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	0	NTCIP
28	X	0	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	0	NTCIP
29	X	1	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	0	NTCIP
30	X	9	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	1	NTCIP
31	X	6	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	0	type queue
32	X	6	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	0	type queue
33	X	0	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	0	NTCIP
34	X	0	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	0	NTCIP
35	X	0	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	0	NTCIP
36	X	0	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	0	NTCIP
37	X	0	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	0	NTCIP
38	X	0	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	0	NTCIP
39	X	0	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	0	NTCIP
40	X	0	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	0	NTCIP
41	X	7	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	0	NTCIP



Logic Statement Control

Statement	Command	Statement	Command	Statement	Command	Statement	Command
1	Enable	26	Enable	51	.	76	.
2	Enable	27	Enable	52	.	77	.
3	Enable	28	Enable	53	.	78	.
4	Enable	29	Enable	54	.	79	.
5	Enable	30	Enable	55	.	80	.
6	Enable	31	.	56	.	81	.
7	Enable	32	.	57	.	82	.
8	Enable	33	.	58	.	83	.
9	Enable	34	.	59	.	84	.
10	Enable	35	.	60	.	85	.
11	Enable	36	.	61	.	86	.
12	Enable	37	.	62	.	87	.
13	Enable	38	.	63	.	88	.
14	Enable	39	.	64	.	89	.
15	Enable	40	.	65	.	90	.
16	Enable	41	.	66	.	91	.
17	Enable	42	.	67	.	92	.
18	Enable	43	.	68	.	93	.
19	Enable	44	.	69	.	94	.
20	Enable	45	.	70	.	95	.
21	Enable	46	.	71	.	96	.
22	Enable	47	.	72	.	97	.
23	Enable	48	.	73	.	98	.
24	Enable	49	.	74	.	99	.
25	Enable	50	.	75	.	100	.

Logic Statements

Statement	Cond	If	Peer	Fail	Element	Desc	Op	Test		Then	Element	Desc	Set
1	1	IF	0	F	CTR SEQUENCE #	1	IS	2	THEN	Then	LP SET LOGIC FLAG	1	ON
	2	OR	0	F	CTR SEQUENCE #	1	IS	4		Then	LP SET COB ON	0	517
	3	OR	0	F	CTR SEQUENCE #	1	IS	6		Then	OL OMIT OVLP A-P	3	ON
	4	OR	0	F	CTR SEQUENCE #	1	IS	8				0	OFF
	5	OR	0	F	CTR SEQUENCE #	1	IS	10				0	OFF
	6	OR	0	F	CTR SEQUENCE #	1	IS	12	ELSE	Else	LP SET LOGIC FLAG	1	OFF
	7	OR	0	F	CTR SEQUENCE #	1	IS	14				0	OFF
	8	OR	0	F	CTR SEQUENCE #	1	IS	16				0	OFF
	9		0	F		0	IS	0				0	OFF
	10		0	F		0	IS	0				0	OFF
2	1	IF	0	F	DET	30	IS	ON	THEN	Then	DET SET VEH 49-64	57	ON
	2	AND	0	F	LP COB CODE OFF	0	IS	513				0	OFF
	3		0	F		0	IS	0				0	OFF
	4		0	F		0	IS	0				0	OFF