JOINT TRANSPORTATION
RESEARCH PROGRAM
INDIANA DEPARTMENT OF TRANSPORTATION
AND PURDUE UNIVERSITY

Development of Automated Incident Detection System Using Existing ATMS CCTV

Stanley Chien, Yaobin Chen, Qiang Yi, Zhengming Ding

SPR-4305 • Report Number: FHWA/IN/JTRP-2019/23 • DOI: 10.5703/1288284317101

RECOMMENDED CITATION

Chien, S., Chen, Y., Yi, Q., & Ding, Z. (2019). *Development of automated incident detection system using existing ATMS CCTV* (Joint Transportation Research Program Publication No. FHWA/IN/JTRP-2019/23). West Lafayette, IN: Purdue University. https://doi.org/10.5703/1288284317101

AUTHORS

Stanley Chien
Professor of Electrical and Computer Engineering
School of Engineering & Technology
Indiana University–Purdue University Indianapolis

Yaobin Chen, PhD
Director of TASI
School of Engineering & Technology
Indiana University–Purdue University Indianapolis
(317) 274-4032
ychen@iupui.edu
Corresponding Author

Qiang Yi
Assistant Research Professor
Transportation Active Safety Institute
School of Engineering & Technology
Indiana University–Purdue University Indianapolis

Zhengming Ding
Assistant Professor of Computer and Information Technology
School of Engineering & Technology
Indiana University–Purdue University Indianapolis

JOINT TRANSPORTATION RESEARCH PROGRAM

The Joint Transportation Research Program serves as a vehicle for INDOT collaboration with higher education institutions and industry in Indiana to facilitate innovation that results in continuous improvement in the planning, design, construction, operation, management and economic efficiency of the Indiana transportation infrastructure. https://engineering.purdue.edu/JTRP/index_html

Published reports of the Joint Transportation Research Program are available at http://docs.lib.purdue.edu/jtrp/.

NOTICE

TECHNICAL REPORT DOCUMENTATION PAGE

1. Report No. FHWA/IN/JTRP-2019/23	**2. Government Accession No.**	**3. Recipient's Catalog No.**
4. Title and Subtitle Development of Automated Incident Detection System Using Existing ATMS CCTV		**5. Report Date** October, 2019 **6. Performing Organization Code**
7. Author(s) Stanley Chen, Yaobin Chen, Qiang Yi, and Zhengming Ding		**8. Performing Organization Report No.** FHWA/IN/JTRP-2019/23
9. Performing Organization Name and Address Joint Transportation Research Program Hall for Discovery and Learning Research (DLR), Suite 204 207 S. Martin Jischke Drive West Lafayette, IN 47907		**10. Work Unit No.** **11. Contract or Grant No.** SPR-4305
12. Sponsoring Agency Name and Address Indiana Department of Transportation (SPR) State Office Building 100 North Senate Avenue Indianapolis, IN 46204		**13. Type of Report and Period Covered** Final Report **14. Sponsoring Agency Code**

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

16. Abstract
Indiana Department of Transportation (INDOT) has over 300 digital cameras along highways in populated areas in Indiana. These cameras are used to monitor traffic conditions around the clock, all year round. Currently, the videos from these cameras are observed by human operators. The main objective of this research is to develop an automatic real-time system to monitor traffic conditions using the INDOT CCTV video feeds by a collaborative research team of the Transportation Active Safety Institute (TASI) at Indiana University-Purdue University Indianapolis (IUPUI) and the Traffic Management Center (TMC) of INDOT. In this project, the research team developed the system architecture based on a detailed system requirement analysis. The first prototype of major system components of the system has been implemented. Specifically, the team has successfully accomplished the following:

1. An AI based deep learning algorithm provided in YOLO3 is selected for vehicle detection which generates the best results for daytime videos.
2. The tracking information of moving vehicles is used to derive the locations of roads and lanes.
3. A database is designed as the center place to gather and distribute the information generated from all camera videos. The database provides all information for the traffic incident detection.
4. A web-based Graphical User Interface (GUI) was developed.
5. The automatic traffic incident detection will be implemented after the traffic flow information being derived accurately.

The research team is currently in the process of integrating the prototypes of all components of the system together to establish a complete system prototype.

17. Key Words traffic incident detection, vehicle classification, deep learning	**18. Distribution Statement** No restrictions. This document is available through the National Technical Information Service, Springfield, VA 22161.

19. Security Classif. (of this report) Unclassified	**20. Security Classif. (of this page)** Unclassified	**21. No. of Pages** 34	**22. Price**

EXECUTIVE SUMMARY

Introduction

The Indiana Department of Transportation (INDOT) has over 300 digital cameras along highways in populated areas of Indiana. These cameras are used to monitor traffic conditions around the clock, all year round. Currently, the videos from these cameras are observed one at a time by human operators. It is very time consuming for the operators to scan through all the video data coming from all the cameras in real time. The main objective of this research was to develop an automatic and real-time system to facilitate the tracking process.

The Transportation Active Safety Institute (TASI) of the Purdue School of Engineering and Technology at Indiana University-Purdue University Indianapolis (IUPUI) and the Traffic Management Center of INDOT have worked together to conduct a one-year research project to develop a system that will monitor the traffic conditions based on the INDOT CCTV video feeds. Specifically, the proposed system will perform traffic flow estimation, incident detection, and classification of vehicles involved in an incident. The goal was to develop a prototype system and prepare for future implementation.

Findings

In this project, the research team developed a detailed system requirement analysis. The requirement study includes the use of case analysis, which specifies (1) the users and their accessibility to the system, the specific features of the system, and use case scenarios; (2) the hardware and software requirements; (3) the user interface requirement; and (4) the constraints of the system.

According to this requirement document, the research team designed a system architecture that includes the needed hardware and software components in addition to the currently existing INDOT CCTV system, the relationship between the added system and the currently existing INDOT system, the database structure for traffic data extracted from the videos, and a user-friendly web-based server for showing the incident locations automatically. The design also considers that the real-time data obtained from this project could be used for future research and applications such as real-time traffic control, accident/congestion alert, quick emergency response, accident investigation, and so forth.

Implementation

The first prototype of most system components has been implemented as follows.

1. Vehicle detection

Automatic detection of vehicles is an essential part of this project. Various methods were tried and compared. The artificial intelligence method in YOLO3 for vehicle detection currently generates the best results for daytime videos. The performance vehicle detection at an easy vehicle detection horizontal line by YOLO_v3 was checked with 21 videos at different light conditions. The results were as follows:

- Daytime detection rate is high (>90%)
- Detection rate at daytime decreases to the 20% range if the videos are not stable (i.e., vibrating)
- Detection rate at evening time (before completely dark) is in the 60%–98% range
- Detection rate in dark lit conditions is in the 40%–70% range
- Detection rate at night without streetlights is extremely low
- Two types of vehicles, car and truck, can be classified
- Raindrops on camera lenses cause significant vehicle detection problem

2. Road detection

Since the aiming direction of each camera can change, it is not possible to specify, a priori, the road and lane locations on the video image. Therefore, the team used the tracking information of moving vehicles to determine roads. Assuming that most vehicles are driven within their lanes, the automatic detection of lanes is also implemented based on the number of cars passing each part of the road. The traffic direction of each lane is also automatically detected. Its performance is not stable yet due to unstable input videos and imperfect vehicle tracking methods. Additional algorithms are being developed to solve this problem.

3. Database development

This project used a database as a central place to gather and distribute the information generated from camera videos and the incident detection results derived from the sensory information in the database. The database also provided the information for the user interface. Tables and their relationships to the database have been designed. The database has been implemented in MySQL and can be converted to PostgreSQL if needed. The design of the database is still being updated.

4. The automatic traffic incidents detection

The automatic traffic incidents detection requires the location of the cameras, the adjacency graph of all cameras in the road network, the number of lanes in each road, the traffic direction of each lane, the traffic flow rates for each road/lane, past incidents, and so on. Automatic traffic incidents detection has not yet been implemented due to the current lack of real-time traffic data. This part will be implemented as soon as the traffic flow information is being derived accurately and uploaded to the database.

5. Web-based Graphical User Interface (GUI)

A web-based Graphical User Interface (GUI) was developed with input and suggestions from INDOT. The GUI reads data from the database and generates the user-interested information on the output display. The GUI displays four types of information: (1) location of all installed cameras on the Google map; (2) all traffic incident locations (shown in red) on the Google map; (3) the real-time video of the focused incident location; and (4) all traffic information at the focused incident location. This GUI supports the selection of a focused traffic incident location among all incident locations.

6. Hardware specification study

After the implementation of various components of the system architecture, the research team ran the system on different computer hardware and software. It concluded that GPU is essential for the real-time implementation of the system. More tests are currently being conducted to decide how many cameras can be supported by each computer. The research team is in the process of integrating all system components to establish a complete system prototype. After that, the performance for each component of the system will be evaluated and further improved as needed.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

1. INTRODUCTION

Indiana Department of Transportation (INDOT) has installed over 300 cameras along highways in populated areas in Indiana. These cameras are used to observe and monitor traffic conditions around the clock, all year round. Currently, the videos from these cameras are observed individually by human operators, and thus, it is really time consuming for the operators to scan through all the real-time video data coming from all the cameras. The main objective of this research is to develop an automatic and real-time system to facilitate the tracking process.

1.1 Trade Study

TrafficVision is a company based in Clemson, South Carolina that was specifically established for Intelligent Transportation Systems (ITS). By using the current camera infrastructure, TrafficVision helps traffic managers make proactive decisions based on immediate incident alerts that are currently monitored manually. TrafficVision has the following features:

- Real-time incidents detection
- Proactive traffic congestion management
- Special events and other applications: monitoring for incidents, construction zones, and problem areas, such as drivers driving the wrong way, stopped vehicles, volume increases, or frequent slow-downs
- Customizable incident alerts

TrafficVision is capable of operating on multiple hardware platforms to fit the organizational needs of clients, in this case, it can be in the form of rack mounted appliances, hardened equipment in the field or even on virtual machines. Web-based application allows users designated by traffic managers to access and configure analytics from anywhere, as long as they are granted network access rights. This removes the need to download and load extra software to all workstations.

Based on the communication with TrafficVision, the cost of licensing the traffic monitoring solution (software and consulting) from an external company is listed below in Table 1.1. On top of the listed cost, the customer will be responsible for providing the all hardware and data storage space.

The annual software license cost of TrafficVision is quite high. The Transportation Active Safety Institute (TASI) at Indiana University-Purdue University Indianapolis (IUPUI) proposed to develop a similar system to meet the need of INDOT.

TABLE 1.1
Cost associated with attaining software license and support

Cost Associated	Duration of Service	Number of Cameras Supported
$500/per camera	1 year	<500
$300/per camera	1 year	>1000

1.2 Proposed Solutions

INDOT and IUPUI proposed to develop a system that will check the traffic conditions based on the INDOT CCTV video feeds. Specifically, it will perform the following:

1. Traffic flow estimation
2. Incident detection
3. Classification of vehicles involved in an incident

In addition, the following tasks are performed:

1. Database will be developed to store the incident data
2. User-friendly web-based server will be built to show the incident locations automatically

An incident is a situation that traffic controller may be interested. These situations may need to be responded. Incidents include the following:

1. Vehicle collisions (the response priority is highest)
2. Obstacles on road
3. Traffic jams

This system enables the long-term development of the intelligent traffic monitoring and control. The real-time data obtained from this project could be used for future research and applications such as real-time traffic control, accident/congestion alert, quick emergency response, accident investigation etc.

2. REQUIREMENTS OF THE PROPOSED SYSTEM

The first step for the development of the system is to develop a set of detailed and accurate system requirements that will be used to guide the development of the whole project period.

2.1 Use Case Analysis

This section describes the intended users and features of the proposed Intelligent Roadway Analyzer (IRA).

2.1.1 Use-Case Diagram

Figure 2.1 is a use-case diagram that describes who will use the system, the features provided by the system and the how the features are used by the users. In this diagram:

- The rectangle in the figure is the boundary of our proposed IRA system
- The stick figures are users of our proposed IRA system
- The oval blocks are the proposed features for the IRA system

2.1.2 Description of Actors (All Possible People Who Will Access This System)

Supervisors (super users). Supervisors are super-traffic operators who have access to all the features in the system through provided user interface (except

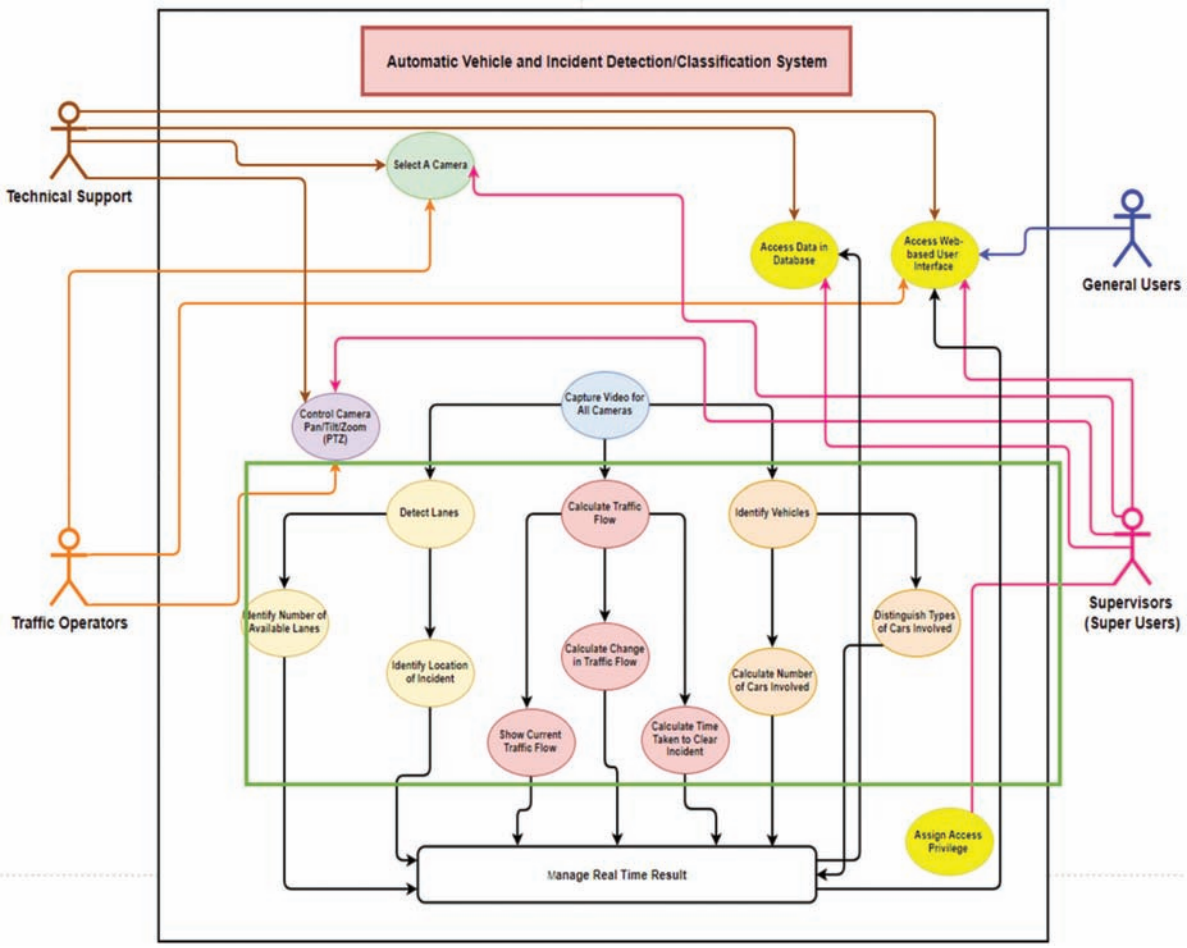

Figure 2.1 Use-case diagram of the proposed automatic system.

direct access of database table). They can control camera Pan/Tilt/Zoom motion, select target camera, access both data in the database and web-based user interface. Additionally, they can also grant network access privileges to different users.

Traffic operators. Traffic operators have access to all the features in the system through provided user interface. They can control camera Pan/Tilt/Zoom motion, select target camera, access both data in the database and web-based user interface. However, they cannot access database tables directly and cannot grant network access privileges to different users.

Technical support. The technical support are people who maintain the computer, database and network. They also have access to select target camera and control camera Pan/Tilt/Zoom for maintenance purposes. They cannot assign user privileges.

General users. General users are people who are only granted access to the web-based server. They can access but cannot edit the real-time video.

2.1.3 Description of Use Cases

1. *Select A Camera* has the following capabilities:
   a. Shall allow users with access privilege to select the video from any target cameras
2. *Control Camera Pan/Tilt/Zoom (PTZ)* has the following capabilities:
   a. Shall allow users with access privilege to Pan/Tilt/Zoom any chosen camera
3. *Capture Video for All Cameras* has the following capabilities:
   a. Shall be able to interface various types of camera system
   b. Shall be able to store last X minutes (e.g., 10 minutes) video
   c. Shall support any number of cameras
   d. Shall remember the current status (angle, PTZ) of all cameras
   e. Shall provide access for real-time video stream for any camera

f. Shall provide access for real-time video stream for some cameras at the same time
g. Shall provide access for real-time video stream for all cameras at the same time
h. Shall provide access of stored video for any camera
i. Shall provide access of stored video for some cameras at the same time
j. Shall provide access of stored video for all cameras at the same time
k. Shall provide the access of the current status (angle, PTZ) of any camera
l. Shall provide the GPS coordinates of each camera

4. *Access Data in Database* has the following capabilities:

   a. Shall support the storage all the static data for each camera, such as the install height, GPS information of the cameras
   b. Shall store the lane information for the camera image when camera is at the predefined PTZ value
   c. Shall support storage of all the static data for each camera, for example, the height of installation, GPS information, lane information
   d. Shall support the storage of videos of interest, such as incident related videos
   e. Shall support the storage of real-time traffic information, including traffic flow, statistics of vehicle types
   f. Shall provide a database model

5. *Access Web-based User Interface* has the following capabilities:

   a. Shall provide a login page
   b. Shall provide a page for adding/removing users
   c. Shall provide a GUI of all locations of cameras
   d. Shall provide the information of the cameras by clicking at a selected camera
   e. Shall have an easy one-click action to let the camera return to preset PTZ
   f. Shall be able to notify correspondents for help (e.g., an ambulance when there is a fatal accident)
   g. Shall be able to display all incidents
   h. Shall be able to sort the incidents according to different criteria, such as severity, occurring time, type of support needed etc.
   i. Shall be able to show the videos of interested cameras
   j. Shall be able to show the GPS locations of incidents
   k. Shall be able to control the Pan/Tilt/Zoom on interested camera

6. *Image Processing* has the following capabilities:

   a. Shall be able to detect the lanes automatically in the condition of different weather, different lighting, and after PTZ changes
   b. Shall be able to identify the vehicle types in the lanes
   c. Shall be able to calculate the real-time traffic flow
   d. Shall be able to detect occurrence of incidents

   6.1 *Detect Lanes* has the following capabilities:

      a. Shall automatically detect lanes at each road section accurate at X meter (e.g., 25 m)
      b. Shall accept manually entered lanes at each road section (e.g., 25 m)
      c. Shall automatically determine lanes at any time (day, night, dawn, and dusk)
      d. Shall automatically determine lanes at any weather conditions (sunny, rain, snow, and fog)
      e. Shall automatically determine lanes within X seconds after PTZ change
      f. Shall be able to distinguish multiple roads in the same view
      g. Shall determine the lanes of multiple roads in the same view
      h. Shall work for straight and curved road

   6.2 *Identify Number of Available Lanes* has the following capabilities

      a. Shall automatically determine actual maximum number of lanes are at each road section accurate at X meter (e.g., 25 m)
      b. Shall accept manually entered maximum number of lanes at each road section (e.g., 25 m)
      c. Shall automatically determine current usable number of lanes are at each road section accurate at X meter (e.g., 25 m)
         i. Shall automatically determine number of current usable lanes at any time (day, night, dawn, and dusk)
         ii. Shall automatically determine number of currently usable lanes in any weather conditions (sunny, rain, snow, fog)
         iii. Shall automatically determine current usable number of lanes within x seconds after PTZ change
      d. Shall be able to distinguish multiple roads in the same view
      e. Shall determine the number of lanes of multiple roads in the same view
      f. Shall work for straight and curved road

   6.3 *Calculate Traffic Flow* has the following capabilities:

      a. Shall be able to determine the traffic flow rate for each location of identified lane
      b. Shall be able to determine automatically the traffic direction on each lane
      c. Shall be able to determine the traffic flow rate for each location of an identified lane
      d. The traffic flow rate for each location of identified lane shall be accurate to 25 m
      e. Shall be able to detect traffic flow rate equal to 0 (no traffic flow)

   6.4 *Identify Location of Incident* has the following capabilities:

      a. Shall be able to identify the any incident defined above
      b. Shall be able to identify the GPS location of sudden increase of flowrate change
      c. Shall be able to identify the GPS location of sudden decrease of flowrate change

   6.5 *Show Current Traffic Flow* has the following capabilities:

      a. Shall be able to show the traffic flow rate at each location
      b. Shall be able to show the location of traffic flow rate change

6.6 *Calculate Change in Traffic Flow* has the following capabilities:

a. Shall be able to determine the traffic flow change location
b. Shall be able to determine the positive traffic flow change rate on each lane
c. Shall be able to determine the negative traffic flow change rate on each lane

6.7 *Calculate Time Taken to Clear Incident* has the following capabilities:

a. Shall be able to predict the current location of sudden flowrate increase after at a jam
b. Shall be able to predict the current location of sudden flowrate decrease after at a jam
c. Shall be able to predict the time to no flowrate change after a jam at the current location of sudden flowrate decrease

6.8 *Identify Vehicles* has the following capabilities:

a. Shall be able to identify the type of vehicles on the road and roadside. The type includes truck, passenger car, bus, etc.
b. Shall be able to identify the orientation of the vehicle relative to the lane direction

6.9 *Calculate Number of Cars Involved* has the following capabilities:

a. Shall be able to determine the number of vehicles involved in an incident that caused traffic jam

6.10 *Distinguish Types of Cars Involved* has the following capabilities:

a. Shall be able to determine the type of each vehicle involved in an incident that caused traffic jam

7. *Manage Real-Time Result* has the following capabilities:

a. Shall store the temporary results from use cases and convert these results to the correct accessible formats desired by users

8. *Assign Access Privilege* has the following capabilities:

a. Shall allow Supervisors to grant access to specific users

2.1.4 Use Case Scenarios

2.1.4.1 Normal standard operation

1. Each traffic monitoring camera continuously feeds its video stream
2. The video is temporarily stored for a predetermined time period
3. Each video stream is processed in real time
4. The generated information is stored in the database
5. If an incident is detected:
   a. The stored video x minutes before the incidents are copied and permanently stored in database
   b. The incident is reported
   c. The associated incident is determined

2.1.4.2 Once congestion starts. In this project, congestion is a form of incident. Through our image-processing algorithm, the Intelligent Roadway Analyzer (IRA) shall be able to recognize the starting location of a congestion. There are multiple reasons for a congestion, mainly collision, obstacles on interstates, road construction and heavy traffic volume due to peak hours. IRA shall recognize the congestion through the change of traffic flow (a sudden decrease of traffic flow shows that a congestion is likely happened. Further, with the data previously stored on average traffic flow and monitored traffic change, IRA also propose an estimated time it will take for the congestion to clear.

2.1.4.3 Once congestion ends. Once congestion ends, the total time taken for the congestion to clear will be recorded and compared to the duration proposed by IRA for further learning and improvement.

2.1.4.4 Background light change. Background light change is a condition that will happen and definitely plays a huge factor when it comes to image processing. Since the lighting condition changes, the algorithm may need to be flexible to accepting the changes in lighting condition and be able to process images even when the quick changes occur.

2.1.4.5 Road surface change. The reflections of road surface changes through the day and is vastly different between day and night. Our solution shall take this into consideration and pre-amp for this change in lighting condition and ensure that our algorithm can overcome this change.

2.1.4.6 Weather change. Weather changes can affect several things: road conditions (slippery, not slippery), change in traffic flow (slower traffic) and change in visibility from cameras amongst many things. As such, our solution may have real-time update of weather (potentially synchronizing weather information from a weather channel), and in the case of forecasted rainy weather, the solution will be ready to identify potentially more incidents that may occur due to harsher weather conditions.

2.1.4.7 Season change. The measures taken in the solution to handle changes in season is the same as the measures taken in 2.1.4.6 for changes in weather.

2.1.4.8 User wants to see overall traffic summary. Our solution will have a web-based user interface that will facilitate the monitoring process for the traffic monitors. There shall be an option in the user interface to allow user to select traffic summary from a dropdown list.

2.1.4.9 User wants to see specific location. In the event that the traffic monitor would like to zoom in on a specific location to look at the traffic condition, he or she shall be able to select the locations through the web-based user

interface, which will contain all the locations of the cameras that shall be made accessible to the traffic monitor.

2.2 Hardware Requirements

2.2.1 Current Hardware System Being Used at TMC of INDOT

The applications of the IRA will be robust and operational on multiple hardware platforms. Instead of turning the clients' current hardware setup obsolete, IRA will be built on the current hardware structure that the clients have to reduce the cost and minimize the interruption of current operation. Figure 2.2 describes the currently existing hardware structure in INDOT.

The interface between the filed cameras and control center is internet (optical, wired or wireless).

The interface between the control centers to each user is also via internet (optical, wired, or wireless).

The current PTZ setup for INDOT's cameras is

- Linux-based
- Controlled by joystick, which sends directional controls to the program that runs locally on a processing computer. The movements are UP and DOWN
- As cameras are bought from different manufacturers, we currently cannot receive camera angle values for the cameras. All settings of cameras are done by technical support of camera manufacturers
- PTZ angles are preset, and there are currently 8 settings relative to road directions
- INDOT does not want to turn PTZ cameras into fixed cameras
- All cameras to be updated in the future are digitally controlled cameras

2.2.2 Proposed Hardware System Diagram

The idea is to build the automatic vehicle and incident detection and classification system upon the currently existing video capturing system and computer network. IRA will simply be an external add-on to any current INDOT hardware setup.

Proposed system structure

Location

- The image-processing computer will be located in the field, as shown in Figure 2.3

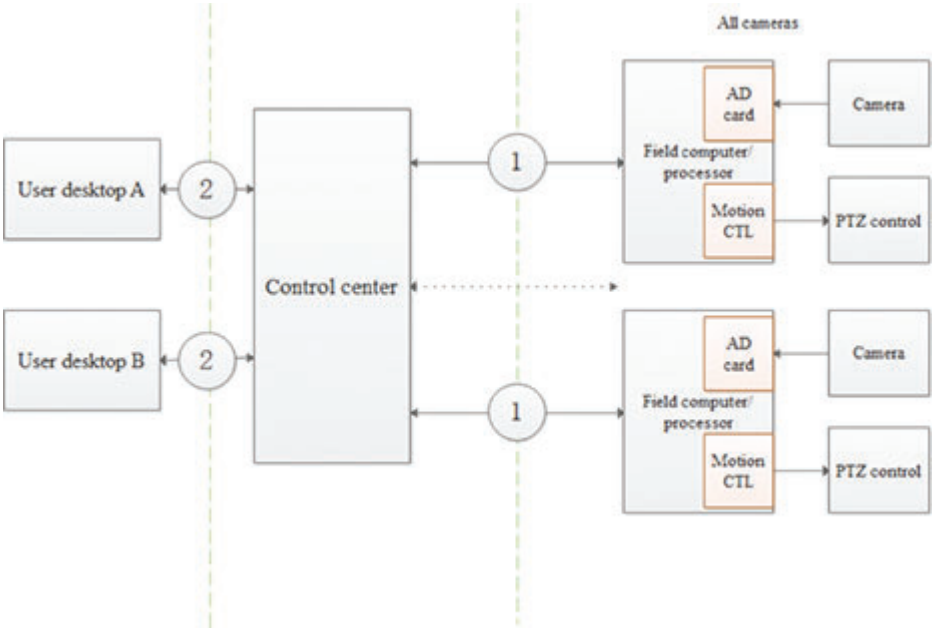

Figure 2.2 INDOT's existing hardware system diagram.

User desktop A/B: These are PCs located in-house at INDOT's office; they can be used by all users. The user can monitor and control the PTZ of each camera.

Control center: Computers host servers and network switches.

Field computer/processor: Computers in the field house for camera interface.

Camera: Cameras are currently installed on all major interstates in Indianapolis, and will be capturing and providing video streams into the system as input. There are different types of cameras.

PTZ control: PTZ control are together with camera. It can be controlled remotely through command protocols. It is assumed that the PTZ angles can be obtained from control center.

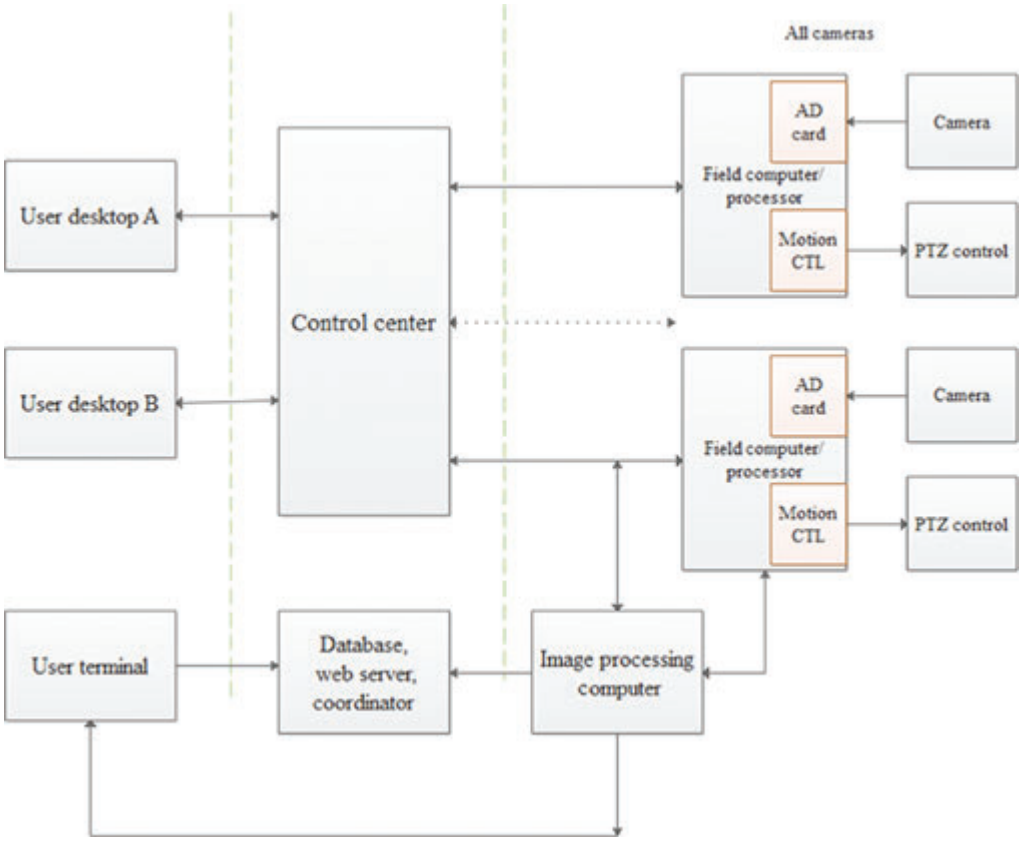

Figure 2.3 Proposed system structure.

- A database server and web server host computer(s) will be located in the control center

System description

- The image-processing computer is responsible for lane detection, vehicle detection, vehicle count, vehicle tracking, and vehicle type recognition and incident detection
- Only the processing results are transferred to the database and user terminal to reduce the data transfer load
- The protocol should be specifically defined
- According to the request from user terminal, the computer could transfer the real-time video to the terminal
- The user terminal could be a desktop, a laptop, and even a smart device
- All the data are transferred through internet

2.3 Software Requirement

This section describes software requirement of the proposed system. The software has three major components (see Figure 2.4).

1. *Intelligent Information extraction component:* This component is responsible for satisfying all camera video processing requirements. This component has two parts:
   a. Lane and vehicle information detection unit
   b. Traffic incident detection unit
2. *Database component:* This component is responsible for storage of all interested video and incident data
3. *Web server (GUI) component:* This component is responsible for all GUI

Camera video data are processed in the lane and vehicle information detection unit to generate number of lanes, traffic flow, and types of vehicles information from each camera video stream in real time. The lane and vehicle information detection unit is a distributed program, each instance of this program is associated to one or several cameras. The results generated by the lane and vehicle detection unit are transmitted in real time to the database using a protocol to be defined in this project. The traffic incident detection unit will reside in the computer in the control center and will detect the traffic incidents based on the latest traffic information in the database reported by the lane and vehicle information detection unit.

Current video streams can be temporarily stored in the image processing computer for a specified period of time. The video stream will be transferred to database for permanent storage if it meets certain conditions (e.g., related to traffic incidences, specific weather condition, etc.). The result of image processing stored in the database can be accessed directly by the user through the web server/GUI platform. When a user makes a request through the server/GUI, the server will communicate with the database and fetch the required

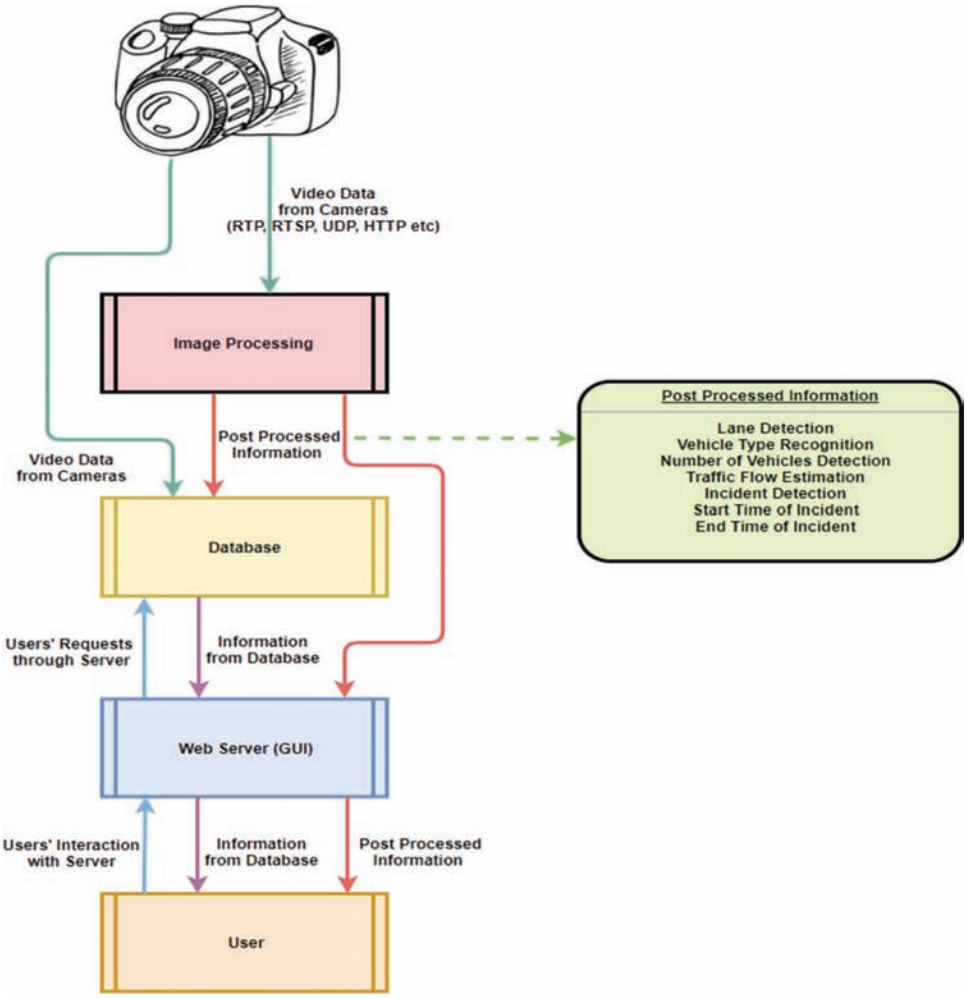

Figure 2.4 Software component diagram.

data back to the server, which will then be transmitted back to the user.

In following sections, these software components are broken down into sub-components.

2.3.1 Lane and Vehicle Information Detection Unit

The tasks to be done in the lane and vehicle information detection unit are described in following subsections.

2.3.1.1 Video stream. The first task is to obtain physical parameters of each camera and verify the accuracy of the input information. These camera parameters include its GPS location, height, resolution, field of view, and orientation with respect to the roads (PTZ angles). If the radar information is available, the capability of the radar will also be gathered. They are essential for evaluating the traffic flow and calculating the vehicle speed. The input of each sensor will be verified. Its potential sub-tasks are listed as follows:

a. The accuracy of the information for each camera needs to be verified and calibrated with objects at known locations on the ground
b. If there is inaccuracy in the camera direction information, a calibration procedure needs to be performed to correct the error

This part is only for one camera video data processing.

2.3.1.2 Capture data

a. Shall support different video stream protocols (mainly RSTP)
b. Shall have access to control the PTZ
c. Shall have the ability to capture the PTZ information from other users

2.3.1.3 Image preprocessing. The image preprocessing includes camera calibration, color space conversion and pre-filtering.

2.3.1.4 Road and lane boundary detection. Road and lane boundary detection is used to determine the ROI (region of interest). It is a one-time subtask when system starts and the PTZ status changes.

a. Shall be able to detect the road and lane boundary in different weather conditions, including sunny, raining and snowing
b. Shall be able to detect the road and lane boundary in different lighting conditions, including daytime, nighttime, and dawn/dusk
c. Shall be able to determine the traffic directions
d. Shall work for straight and curved road

2.3.1.5 Vehicle detection and recognition. Vehicles in the ROI should be detected and recognized for the vehicle tracking.

a. Shall be able to detect and identify the types of vehicles on the road, including truck, passage cars, bus, motorcycle, etc.
b. Shall be able to identify the vehicle types in different lanes
c. Shall be able to identify the motion direction of vehicles
d. Shall be able to detect and recognize vehicles in different weather conditions, including sunny, raining, and snowing
e. Shall be able to detect and recognize vehicles in different lighting conditions, including daytime, nighttime, and dawn/dusk
f. The accuracy for the vehicle recognition should be higher than 90%

2.3.1.6 Multiple vehicle tracking. Vehicle tracking is the intermediate step for the traffic flow estimation and the determination of the vehicle moving direction.

2.3.1.7 Traffic flow estimation. Traffic flow is a key parameter for the incident detection.

a. Shall be able to estimate the traffic flow in each single lane
b. The accuracy of the average traffic flow should be higher than 90%

2.3.1.8 Real-time data management. The real-time data management is used to record the results of video processing, cooperated with database and web servers.

a. Defines the communication protocol with webserver
b. Shall be able to access the database
c. Shall be able to record the results of the video processing
d. Shall be able to send the alarm message to the webserver

2.3.2 Traffic Incident Detection Unit

This unit runs on a computer in the central control center. The unit reads the current traffic flow rate in the database and uses the variation of the traffic flow for incident detection.

a. Shall be able to detect the incident in each single lane
b. Shall be able to identify the involved vehicles in the incident
c. Shall be able to estimate the location of the incident
d. Shall be able to identify partially the incident types
e. Shall be able to record the incident time and locations
f. Shall be able to record the videos when the incident happens
g. Shall be able to estimate the recovery times of the incident
h. According to the literature reviews, the accuracy of the incident detection is tentatively set as 70%
i. Indicate which video corresponding to a traffic incident should be permanently stored
j. Aiming the camera to incident location to get close look and more information

2.4 Database Requirement

The proposed database shall have following features:

- Store all camera static information with following parameters:
  - Camera ID, make, model, technical specifications
  - Installation location information
  - Maintenance/calibration information
  - Default and preset angles
- Store camera-associated road information:
  - Camera ID
  - Location ID
  - Number of roads at each viewing angle
  - Road type (road, ramp, flyover etc.)
  - Speed limit of each road
  - Number of lanes on each road
  - Number of usable lanes
  - The condition of each lane
- Store current weather-related information:
  - Location ID
  - Wind speed
  - Wind direction
  - Temperature
  - Weather type: sunny, rainy, snowy, foggy
  - Road surface conditions: normal, slippery
- Store selected images according a defined file naming system. The file name should contain the following:
  - Date, time, camera ID, location ID, duration
- Store the traffic incident data. The database will store the data for automatic incident video documentation, which may be desirable by INDOT, and definitely of interest to the police force and insurance companies. The data saved to the database will be classified into different data sets:
  - Location of incident
  - Lane numbers where incident has happened
  - Direction of traffic
  - Status: resolved or awaiting cleanup support
  - The identity of involved vehicles
  - Passenger auto, minivan, van, motorcycle, large truck, truck, light truck, bus
  - The number of cars involved in the incident
  - Current traffic flow for all lanes in the incident location
  - Percentage of different car types
  - Image resolution
  - Start time of the incident
  - End time of the incident
  - Incident status

- Emergency response information:
  - Emergency response types
  - Incident types
  - Emergence response associated to incident types
- The initial database shall be developed using Indiana University (IU) available database MySQL on window-based system
- The final database shall be developed according to INDOT specified environment
- The protocol for interfacing all other related components should be defined

Store temporary data. The database storage is broken down into several components, as shown in Figure 2.5.

2.5 User Interface Requirement

2.5.1 Web-Based Server Software

IRA shall have a web-server and a well-designed graphical user interface. The information available through the web-based server shall be organized to cater to the different interests of users. The example user interests defined in this project are as follows:

User Interest 1

a. Is there any incident?
b. In the event of an incident, what is the level of response priority?

User Interest 2

a. Is there any incident for some area/location of interest?

User Interest 3

a. What is the location of the specific incident?
b. What is the time when the specific incident happened?
c. Is there a video footage of the specific incident?

User Interest 4

a. What is the overall traffic situation (across all cameras)?

User Interest 5

a. What is the statistics for each type of traffic situation?

User Interests X: Need to talk to users to find them out

To address these potential interests of the users, there are many features identified, which need to be addressed.

2.5.2 Graphical User Interface

GUI shall provide the following information (settings that change frequently).

- An interactive map with locations of all cameras (see Figure 2.6)
- GPS locations and/or addresses of incidents
- IDs of cameras that can see the incidents
- Type of incident (e.g., fatal, road blocks)
- The start time of each incident
- The end time of incident of each incident
- Response priority of incident

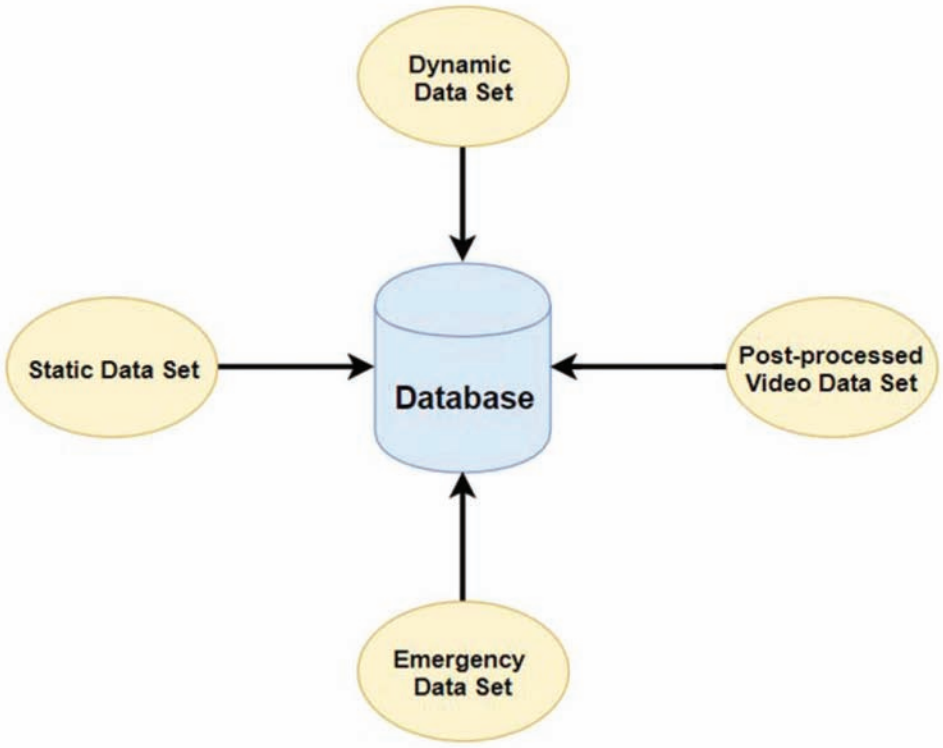

Figure 2.5 Breakdown of database storage.

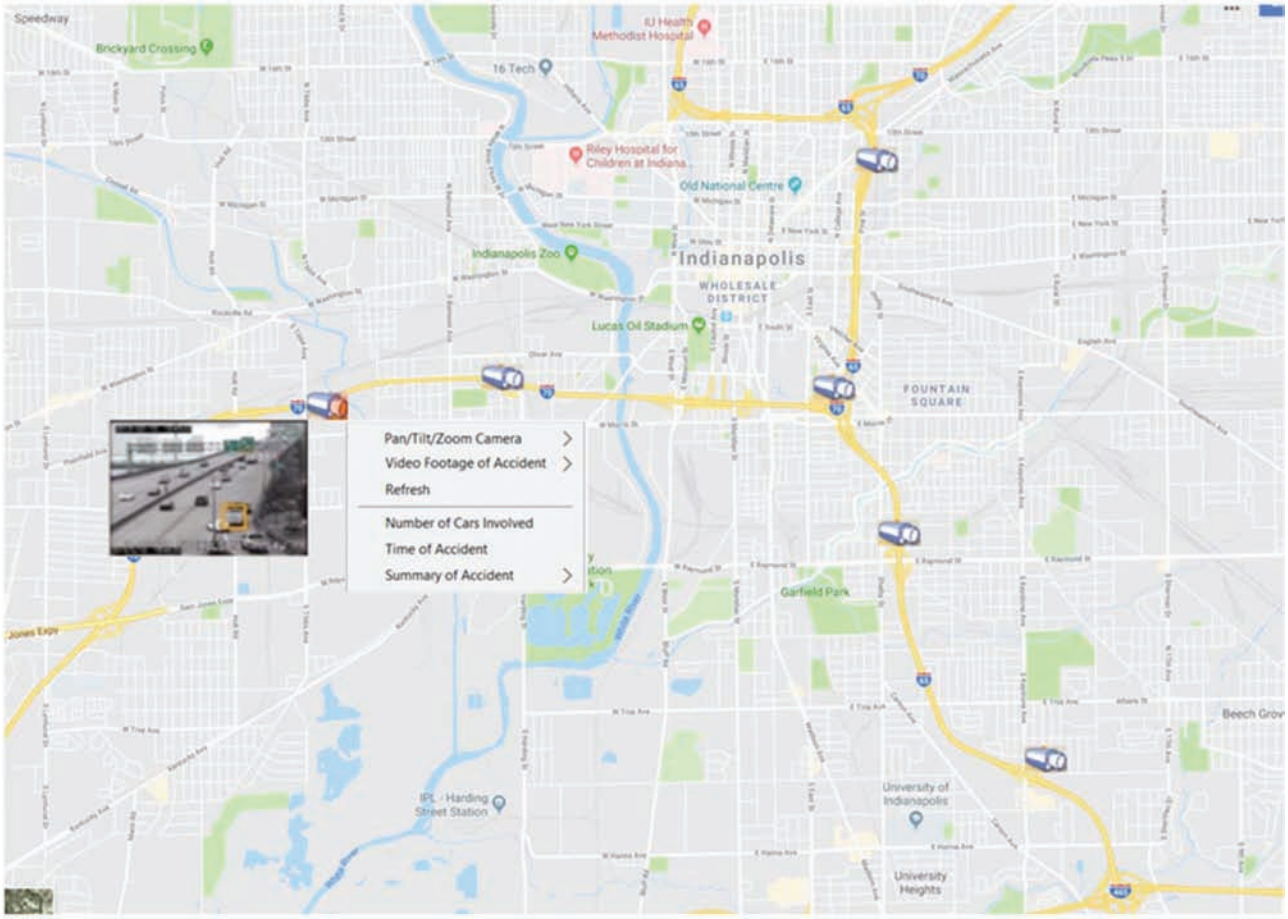

Figure 2.6 Interactive map-based user interface development.

2.6 Deployment Diagram

Figure 2.7 demonstrates the deployment diagram. This figure shows how the hardware components and the software components work together, specifically, what software works on what hardware. Non-shaded boxes are the hardware components. The orange shaded boxes are the software components.

2.7 Constraints and Standards

Hardware

- Must use the cameras provided by INDOT
- Must use the communication protocol specified in current INDOT CCTV system
- The interface between the camera and data processing computer is internet
- The interface between the data processing computer and database is internet
- The interface between the database and web server is internet
- The interface between the web server and data storage is internet

Software

- Shall use INDOT specified database (PostgreSQL)
- Shall run on Linux environment

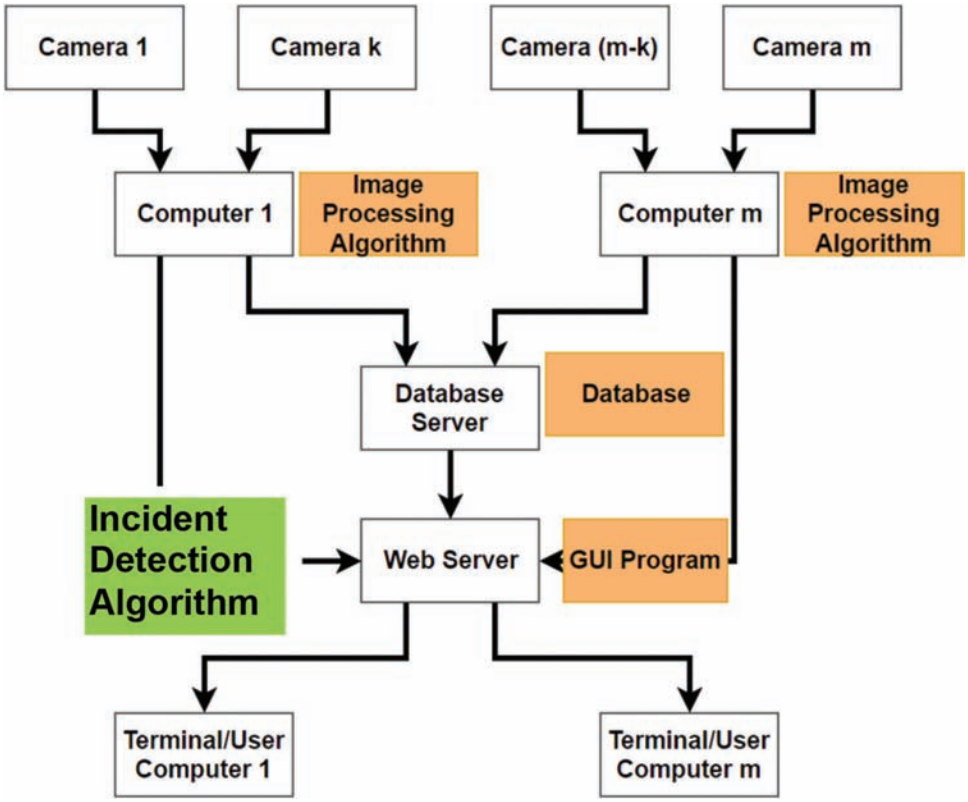

Figure 2.7 The deployment diagram.

3. SYSTEM ARCHITECTURE AND DEVELOPMENT

In this project we developed the system structure based on the requirements described in Section 2. The structure is robust and can be easily applied to a larger system. The design is derived from the knowledge that INDOT currently uses approximately 300 cameras all over the state of Indiana. Figure 3.1 shows the overall system structure for the entire system that is implemented in this project. The entire system has three subsystems, as indicated by the numbers 1 through 3. The first sub-system is the Image Processing portion of the implementation (page numbers). The second subsystem is the communication between the Central Computer and the Database Server (page numbers). The third sub-system is the connection between the Graphical User Interface and the Web Server (page numbers). All these three subsystems will be further broken down and explained in the following sections.

3.1 Camera/Pan_Tilt_Zoom

The control of camera/Pan_Tilt_Zoom needs to be experimented in the later stage of this project. INDOT has provided the protocol document for sending command to the camera but not the privilege of doing the operation yet.

3.2 Field Computer Operation

The input into computers is the raw video data from the cameras. The IP address of over 100 cameras on the road are provided by INDOT. There are always some cameras not providing the video. TASI will write a program to detect automatically the operation status of each camera.

The outputs of each of the computers will be real-time data obtained from video streams. The result is passed and stored in the Database Server every 30 sec. Only the following relevant information will be passed to the database:

- Number of lanes and lane locations
- Traffic direction of each lane
- Vehicle flow rate on each lane
- Percentage of each type of vehicles on each lane

3.2.1 Video Processing—Road Detection

Since the aiming direction of each camera can change, it is not possible to specify a priori the road and lane locations on the video image. Therefore, automatic detection of moving vehicles and using moving vehicles to derive roads and lanes on the image is essential. We have tried two ways to identify roads:

- **Method 1.** Since the highway video images are mostly still except the moving vehicles, this property can be used to

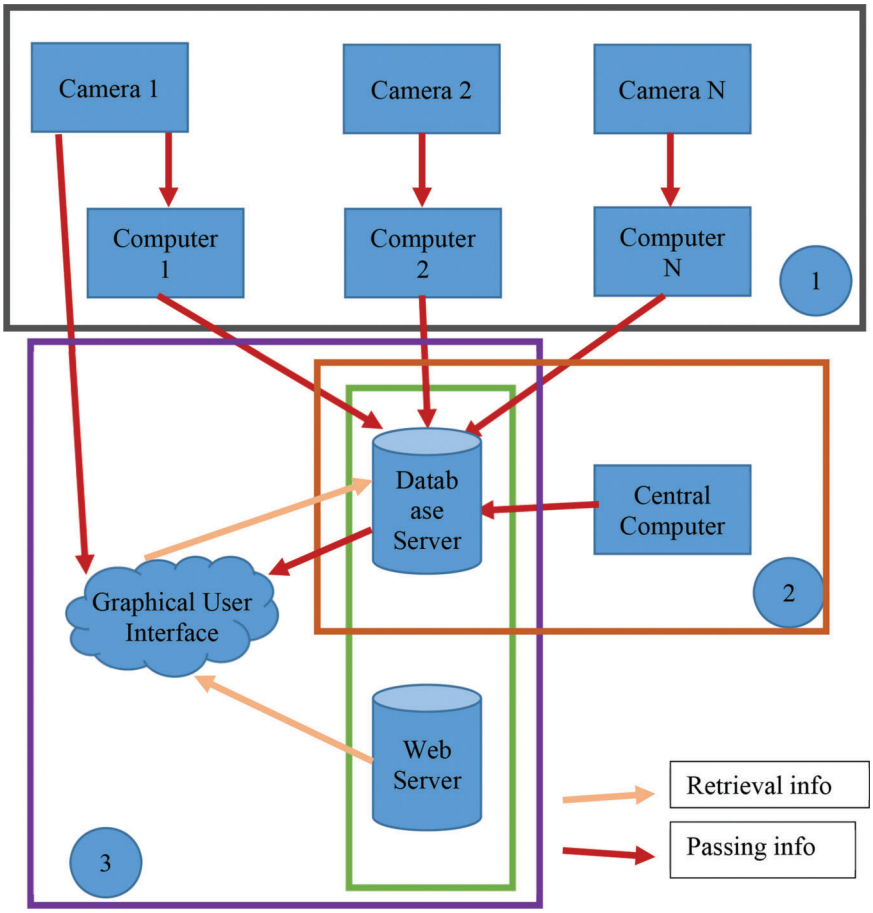

Figure 3.1 Overall system structure.

identify roads. The road can be identified as the areas on the image that change over time. The change is mostly due to vehicle motion. Even though the tree motion can also cause image changes, it can be removed using some filtering methods.

- **Method 2.** Identify vehicles on video images first and find the vehicle location at a defined horizontal line crossing the image. The vehicle location means the location of the road.

The first method is fast but the accuracy is affected heavily by the vehicle shadow generated by the sunlight. The second method depends on the accuracy of the vehicle detection (to be discussed in Sections 3.2.4 and 3.2.5)

3.2.2 Video Processing—Lane Location

Lane detection is actually the detection of lanes at a selected position on the road. It is based on the detection of vehicles on the road (to be discussed in Sections 3.2.4 and 3.2.5) and the assumption that most vehicles are driven within the lanes most time. The approach is to track the center location of the vehicle passing a horizontal line in the image, and count the number of vehicles passing each location on the horizontal line. The horizontal line on the image is chosen for each vehicle detection. Over the time (e.g., 30 seconds), or over the number of vehicles passing the horizontal line, we can get a histogram where the horizontal axis is the pixel location on the horizontal line and the vertical axis is the number of cars passed the horizontal line. Assuming that most vehicles are driven within a lane most time, the peaks on the plot indicate the location of each lane on the road.

3.2.3 Video Processing—Lane Direction

The lane direction is based on the vehicle motion direction. If the vehicle is not moving super-fast, the same vehicle can be seen at the same location of the horizontal line in several consecutive video images. By tracking the vehicle positions on the consecutive images, we can determine the vehicle moving direction. If many vehicles on the same lane are moving in the same direction, we can determine that the direction of the most vehicles on that lane is the lane direction (Figure 3.2).

The advantage of this method is that the lane can be detected for any camera direction. The disadvantage is that it takes some time to determine the lane direction at each new camera direction. However, the camera direction is constant unless the operator changes it. If the operator is changing the camera direction, it means

Figure 3.2 Automatically detected lanes and lane direction.

that the operator is looking for the traffic condition and the automated traffic condition tracking by the computer is not essential.

3.2.4 Video Processing—Car Detection Based on Consecutive Images

Car detection is the key for the success of this project. It needs to be accurate, fast, and not computation cost prohibitive. Two methods were considered and tested in this project. The first is based on the pixel difference of consecutive video images. The second is based on an AI object detection and classification technique YOLO. The approach for these two methods are described in following sub sections.

The block diagram of the first method is shown in Figure 3.3. The operation of each block is described as follows:

A. Video format conversion and frame extraction

a. The input is a raw RGB stream supplying images at a rate of 25 per second (Figure 3.4)
b. The output is a single frame obtained by the OpenCV VideoReader::read() method called on a VideoReader object, where the given parameter is the URL of the camera stream for its construction. The output frame/image is a two dimensional array of either a 3-tuple in case of RGB or a single value in case of Gray Scale

B. Background subtraction and generation of road mask. This step is executed once after the camera orientation is changed.

a. The inputs are N consecutive image frames starting from the first frame obtained from the camera whenever the camera changes its physical orientation (i.e., repositions itself). (The bigger the number N, the better the quality of output will be. However, we must also consider the time complexity of such an operation. In our current program N = 500)
b. The output is a grey scale image, which depicts the road
c. The algorithm is:

   i. For each frame: Use an OpenCV::Background SubtractorMOG object to obtain the foreground in each frame in its grey scale equivalent (Figure 3.5). For $n = 1$, the 0th image is a black image, n starts from 1. The nth image operates on (n-1)th raw (grey scale) image
   ii. For each output frame in Step i: Apply the Open CV::blur function with pre-specified parameters (a kernel size of 10 in our program) to obtain its filtered equivalent (Figure 3.6)
   iii. Logically OR all filtered images obtained in Step ii with each other, thus obtain a single grey scale image, which depicts the detected roads and other noisy shapes (Figure 3.7)

C. Contour detection and noise removal. Contours is a curve joining all the continuous points along the boundary of an area that has the same color or intensity. Here we are interested to use the contour to describe the boundary of roads.

a. The input is the final output image as shown in Figure 3.8
b. The output is a contour describes by an array of points (x-, y-coordinates) of the boundary of only the roads, having removed any noise from the input image; and image showing only the road
c. The algorithm is:

   i. Use the OpenCV library function, **findcontours**, with the input image as argument to obtain the contours (as arrays of points) of all white shapes in the image. (*Do not use any chaining approximation while finding the contour, every single border point must be obtained. This is essential for ROI finding*.) Note that a contour traces white so it may circle either a white area or

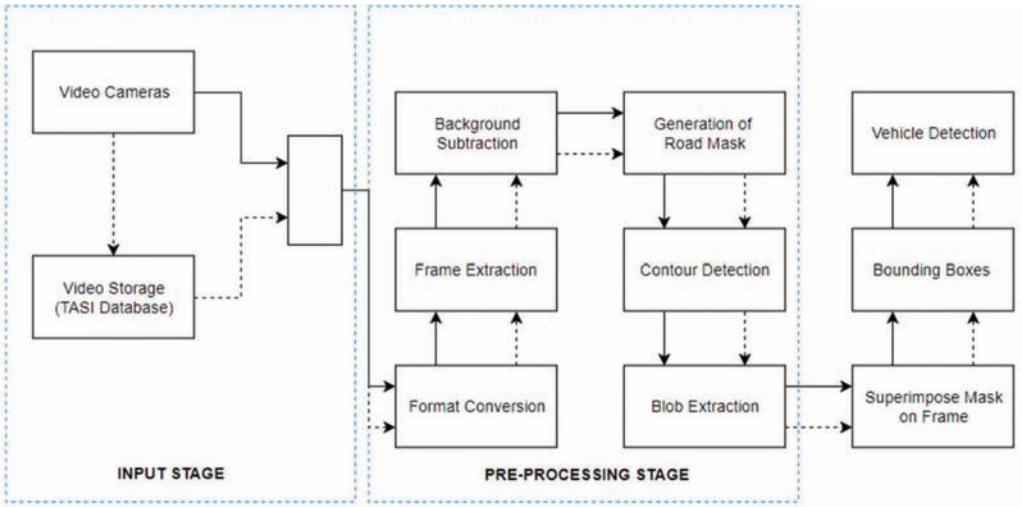

Figure 3.3 Design flowchart of vehicle detection.

Figure 3.4 An example frame.

Figure 3.5 Current frame n operates on (n-1) frame, $n > 1$.

Figure 3.6 Blurred foreground.

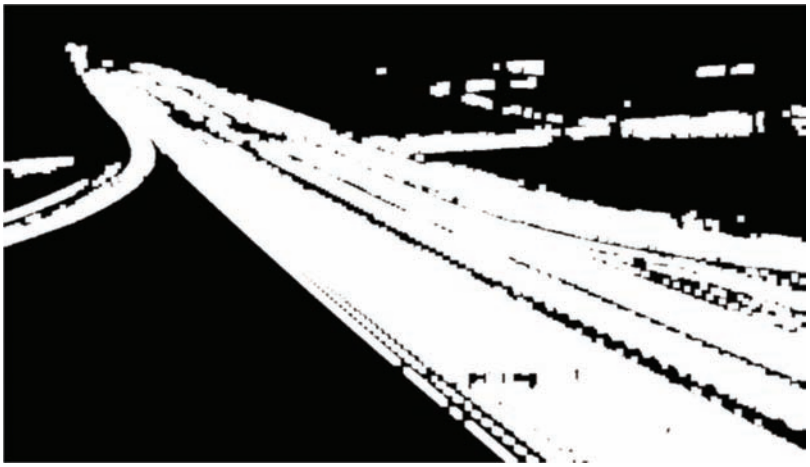

Figure 3.7 Generated road mask.

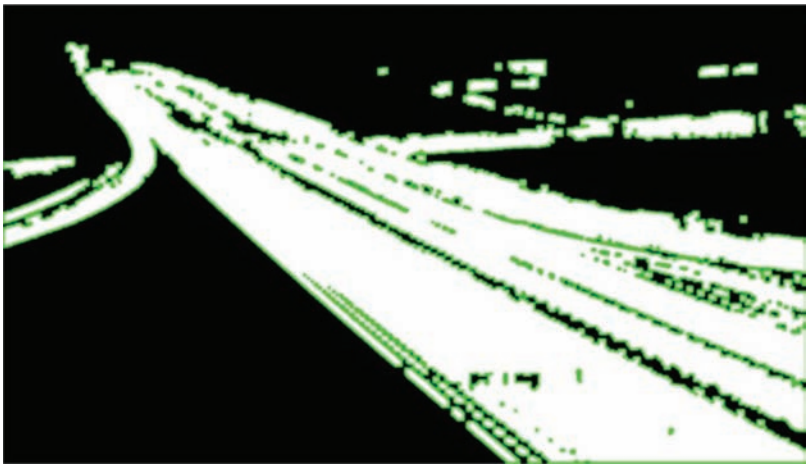

Figure 3.8 Image depicting initial contours of the mask image (illustration).

black area. Image boundary cannot be part of a contour. Figure 3.8 is an illustration

ii. Calculate the area of each contour and find the maximum area among all contours
iii. Delete those contours from the list of contours that have an area less than a specified percentage (e.g., 10% currently used in our program, but may need to be changed after testing) of the maximum found in step ii
iv. Generate a black/white image with the same dimensions as the original input image. In this image, fill the shapes formed by the contours obtained

Figure 3.9 Clear road image.

from Step iii with white using the OpenCV function, drawContours. The result will be like the image in Figure 3.9

D. Finding the region of interest (ROI). To improved speed of operation and avoid unnecessary computation, a region of interest is defined. The way to define the ROI is as follows:

a. The input are the contours and the output image from Step C
b. The output is a grey scale image with only the part of the road within the ROI, the bounds of the ROI, and the area of this part of the road
c. The algorithm is:

   i. Scan successive columns of the input image from left to right. The first column that contains a white pixel is the left extreme of the white region of the image
   ii. Scan successive columns of the image from left to right. The last column to contain a white pixel is the right extreme of the white region of the image
   iii. Scan successive rows of the image from top to bottom. The first instance when a row contains a white pixel is the top extreme of the white region of the image
   iv. Scan successive rows of the image from bottom to top. The first instance when a row contains a white pixel is the bottom extreme of the white region of the image
   v. The indices of the columns form the left and right give the x coordinates of the ROI and similarly the indices of the rows form the top and bottom gives the y coordinates of the ROI (Figure 3.10)
   vi. Since top of the image is far away from the camera, it is hard to detect vehicles in this region. So the top part of the image is clipped (e.g., 20% on the top of the image is clipped currently in our program)
   vii. Black all part of the image outside of the white region by making a copy image having only the rectangle filled with white and AND-ing it with the original image
   viii. Find the contour of the result of Step vii and find its area using the OpenCV function, contourArea

Figure 3.10 Actual ROI image to be used in later steps.

E. Bounding boxes on detected cars. This step should be run for every input image in real time.

a. The inputs are the current frame, and the ROI image obtained in Step D
b. The output is a list of bounding boxes of vehicles, where each box is a 4-tuple describing top-left x coordinate, top-left y coordinate, box width, box height, in sequence
c. The algorithm is as follows:

   i. Use the MOG Background subtractor as in Step C on current frame to obtain its foreground. (*n*th image operates on (*n*-1)th image, Figure 3.11)
   ii. Logically AND the output image in step i with the ROI image (Figure 3.12)
   iii. Find contours of vehicles in the image obtained in step ii using the OpenCV function, boundingRect, on each contour, whose area is greater than 100 and discard the others. The remaining contours are our output (Figure 3.13)
   iv. Then clear any overlapping boxes by checking each box and comparing it with the bounds of all other boxes. If any box overlaps with another box with an area of overlap above 20%, discard the smaller box. Repeat this step until no boxes overlap with an area more than 20% of the box area (Figure 3.14)

Figure 3.11 Current frame i operates on (i-1) frame $i > 1$.

Figure 3.12 Current frame.

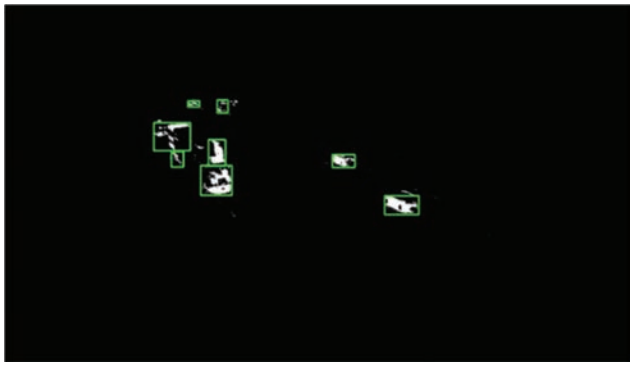

Figure 3.13 Bounding boxes for vehicles. (Boxes shown only for illustrative purposes.)

v. (For illustration in test programs only) Add the bounding boxes and ROI lines to the original frame (Figure 3.15)

3.2.5 Video Processing—AI Car Detection Based on a Single Image

3.2.5.1 Introduction of YOLO. To detect and classify the vehicles in video frames, we also explored various deep learning algorithms. One deep learning algorithm, You Only Look Once (Redmon & Farhadi, 2017), seems to match what we need. YOLO can be trained to recognize hundreds of different objects (such as human, car, truck, train, motorcycles, etc.) from a video frame and directly generates bounding boxes around recognized objects in real time. However, it needs a graphic processing unit (GPU) in a computer to achieve real-time operation. In this INDOT project, we are interested in YOLO's ability to recognize various vehicles.

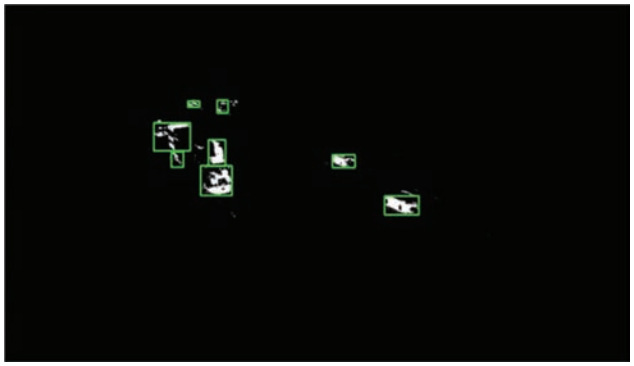

Figure 3.14 Overlapping boxes removed (illustration).

Prior object detection systems apply the model of interested objects to an image at multiple locations and scales. High scoring regions of the image are considered detections. YOLO is a totally different approach, which applies a single neural network to the full image. This network divides the image into regions and predicts bounding boxes and probabilities for each region. These bounding boxes are weighted by the predicted probabilities.

YOLO has several advantages over classifier-based systems. It looks at the whole image at test time so its predictions are informed by global context in the image. It also makes predictions with a single network evaluation unlike other systems (such as R-CNN) which require thousands evaluations for a single image. This makes it more than $1000\times$ faster than R-CNN and $100\times$ faster than Fast R-CNN. YOLO has been improved in several versions. The third version of the algorithm, YOLOv3, uses a few tricks to improve training and increase performance, including multi-scale predictions, a better backbone classifier, etc. (Redmon & Farhadi, 2018). By default, YOLOv3 only displays objects detected with a confidence of .25 or higher. We can change this confidence threshold by passing the -thresh <val> flag to the yolo command. For example, to display all detection, we can set the threshold to 0. YOLOv3 uses a 106 layer fully convolutional underlying neural network architecture (Figure 3.16).

YOLOv3 also has many implementation versions based on different input image sizes, such as, YOLO v3-320×320 (the input image is 320×320), YOLOv 3-416×416, YOLOv3-608×608, and YOLOv3-tiny. Figure 3.17 shows the mean Average Precision (mAP) performance and inference time on Microsoft COCO for different algorithms. Microsoft COCO is a large-scale object detection, segmentation, and captioning dataset (http://cocodataset.org/#home).

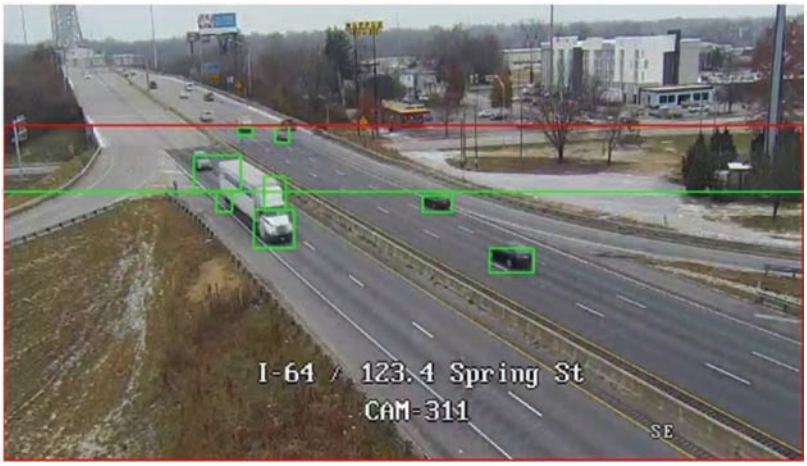

Figure 3.15 Result of step v.

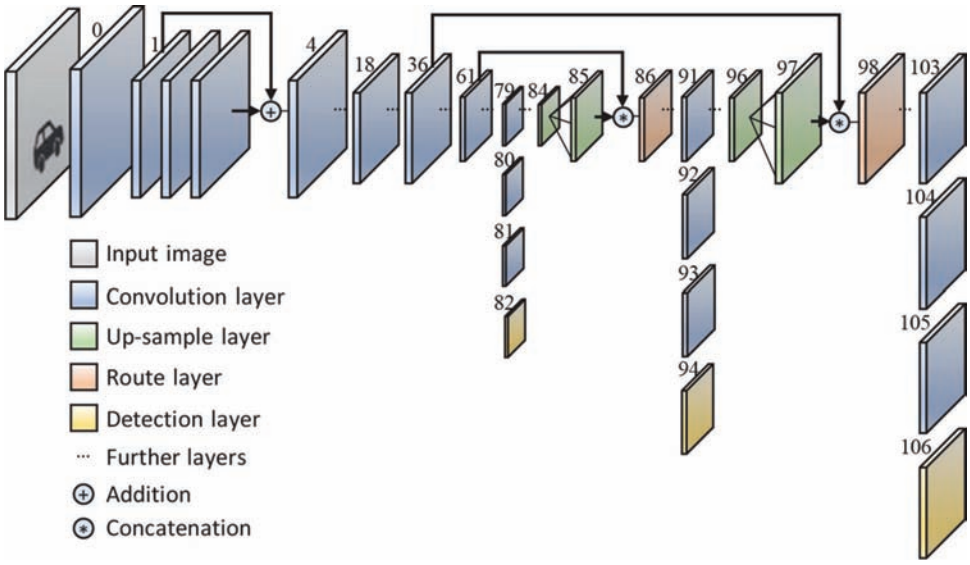

Figure 3.16 YOLOv3 network architecture.

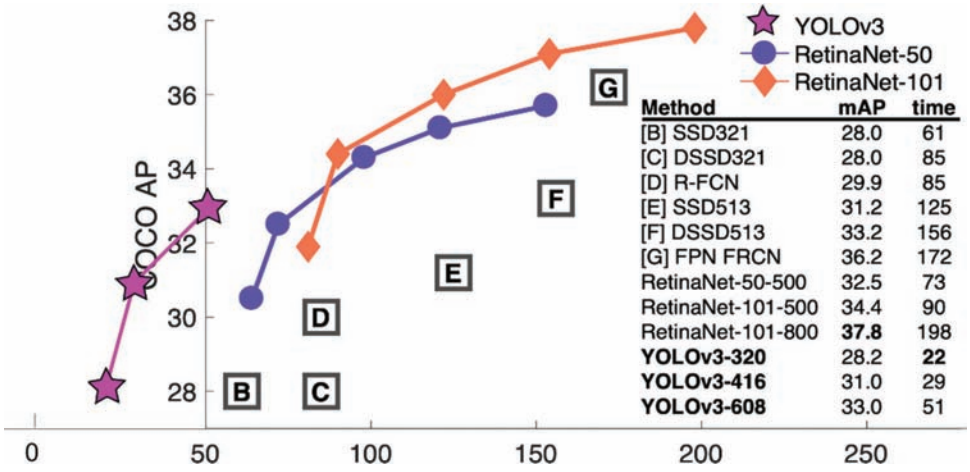

Figure 3.17 Inference time (ms) and performance on COCO dataset.

There are two different implementations of YOLOv3, i.e., TensorFlow (https://www.tensorflow.org/) and Keras (https://keras.io/). TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library and is also used for machine learning applications such as neural networks. It is used for both research and production at Google. Keras is a free and open-source neural-network library written in Python. It is capable of running on top of TensorFlow, Microsoft Cognitive Toolkit, Theano, or PlaidML. It was developed as part of the research effort of project Open-ended Neuro-Electronic Intelligent Robot Operating System (ONEIROS) and its primary author and maintainer is François Chollet, a Google engineer. Chollet also is the author of the XCeption deep neural network model.

Both TensorFlow and Keras implementations of YOLOv3 can be run on both Linux Ubuntu and Windows 10 computers. It can be executed on computers with or without GPU. So far, GPU version on Linux Ubuntu has some issues due to the GPU configuration.

3.2.5.2 YOLOv3 evaluation on INDOT's video frame. Figure 3.18 shows an example frame of vehicle, lane, and lane direction detection using YOLOv3. The box is generated by the YOLOv3 and the number associated with each box is the vehicle type and the confidence percentage of the vehicle recognition. Figure 3.19 shows another example of YOLOv3 based vehicle detection in lit light condition at night. The traffic flow rate on each lane can be achieved by tracking each

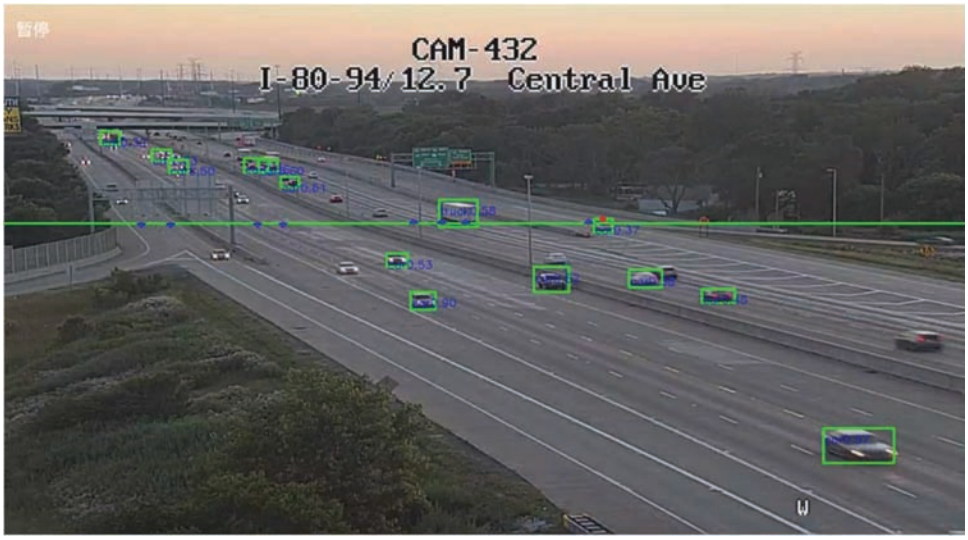

Figure 3.18 Results of YOLO detection and following lane detection.

Figure 3.19 Detection results at nighttime (lit by light poles).

recognized vehicle when it is passing the yellow horizontal line and counting the number of vehicles on each lane over time continuously. Compared with our previous image processing method (Section 3.2.4), YOLOv3 does not rely on the vehicle motion and is not affected by sun shadows. Since it is more robust, we intend to use YOLO for vehicle detection and may use the previous results for verifications.

Since YOLOv3 is the state-of-the-art object detection model, we tested TensorFlow and Keras implementation of YOLOv3 on two computers, which both installed Windows 10 but have different GPU/CPU configurations.

- One computer has an Intel 8 core i7-9800x and the other computer has an Intel 4 core i7-7700HQ. The bench mark speed of i7-9800x is about 30% faster than that of i7-7700HQ. The cost of i7-9800x is more than double of that of i7-7700HQ
- One computer has a Quadro RTX 5000 with 16G Memory and the other computer has a GeForce GTX 1060 GPU with 6G Memory. The cost of Quadro RTX 5000 with 16G Memory is 3.5 times of the cost of GeForce GTX 1060 GPU with 6G Memory

3.2.5.2.1 Speed evaluation. Table 3.1 shows the execution time of TensorFlow and Keras for the INDOT video frame on these two computers. From the results, we found that the TensorFlow version is faster and we can use less advanced GPU/CPU like GeForce GTX 1060 and i7-7700HQ.

Based on Table 3.1, we can conclude the following:

- GPU is necessary
- TensorFlow is faster than Keras
- For processing one video stream, powerful GPU is not helpful

We further tested if one computer can process videos from multiple cameras simultaneously. We tried two implementations. The first used a simple strategy, which is running multiple copies of the program (one for each camera video). For this strategy, we ran two projects together in real-time with TensorFlow based YOLOv3 under GeForce GTX 1060 with 6G Memory and i7-7700HQ. The second approach explored the parallel computing (Python multiprocessing library). But we found the processing time increased when we ran more videos at the same time. The issue could be that we need to call multiple YOLOv3 for each video, and then the resources of O/I and CPU/GPU would be run out. We plan to try more parallel computing techniques to make one machine run more videos at the same time and real time.

3.2.5.2.2 Object detection accuracy evaluation. The performance of the vehicle detection at an easy vehicle detection horizontal line by YOLOv3 is checked with 21 videos at different light conditions. The result is shown in Table 3.2. The result is as follows:

- Daytime detection rate is high (>90%)
- Detection rate at daytime decreases to 20% range as the images are not stable (vibrating). Need to figure out the way to eliminate the video vibration
- Evening time (before totally dark) detection rate is in 60%–98%
- Detection rate is in 40%–70% range in dark lit condition
- Detection rate is extremely low in totally dark condition

3.2.5.3 Remaining work for YOLO-based vehicle detection. Since the computation speed and detection accuracy of YOLOv3 are important to the applicability of this system to be developed, we still need to study several issues.

1. Accuracy improvement—There are still many situations that YOLOv3 does not work well, such as:
   - Inconsistent vehicle detection in a sequence of video frames. Figure 3.20 shows a vehicle in red circle undetected in the current frame but detected in previous frames
   - Unlit road—Since the vehicles cannot be seen on the video images and only the light halo can be seen on the image, YOLO cannot classify the light halo as vehicles. Other algorithms need to be developed
   - Camera with raindrops—We will not work on this problem until other issues are solved
2. Computation cost improvement

Although it seems that the computation speed can achieve real-time operation, it requires expensive computer and GPU. There are two ways to reduce the cost,

TABLE 3.1
The execution time of TensorFlow and Keras for the INDOT video frame on two computers

YOLO v3 Implementation	GPU	CPU	Time Per Frame
TensorFlow	Quadro RTX 5000 Memory: 16G	i7-9800x	≈0.04s
TensorFlow	GeForce GTX 1060 Memory: 6G	i7-7700HQ	≈0.04s
TensorFlow	N/A	i7-9800x	≈0.30s
TensorFlow	N/A	i7-7700HQ	≈0.35s
Keras	Quadro RTX 5000 Memory: 16G	i7-9800x	≈0.09s
Keras	GeForce GTX 1060 Memory: 6G	i7-7700HQ	≈0.10s

TABLE 3.2
The vehicle detection of YOLO_v3 using video at different light conditions

Video Name	Light Condition	Detection				Detection Rate (%)
		True	False	No	Total	
64_1234+2018-11-04+16.1	day	300	2	8	308	97.40
65_90_2+2018-11-03+13.54	day	160	8	17	177	90.40
65_2526+2018-11-08+10.53	day	184	23	14	198	92.93
US-31_1269+2018-11-06+9.37	day	220	81	20	240	91.67
64_1234+2018-11-02+21.43	day	109		51	160	68.13
80.94 _9+2018-10-23+9.00	day	158	4	440	598	26.42
I-70_848+2018-10-29+15.9	day	156	160	205	361	43.21
94_170+2018-10-26+7.55	evening (before dark)	272		4	276	98.6
I-70_833+2018-11-06+7.26 (right side)	evening (before dark)	296		16	312	94.9
I-69_2037_2+2018-10-27+8.3	evening (before dark)	201		54	255	78.8
I-70_833+2018-11-06+7.26 (left side)	evening (before dark)	52		511	563	9.2
65_56+2018-11-04+17.35	evening (before dark)	154		86	240	64.2
I-65_1084+2018-11-05+6.33	evening (before dark)	190		122	312	60.9
I-69_2091+2018-11-05+17.54	evening (before dark)	282		38	320	88.1
265_94+2018-11-05+23.48	dark lit	0		16	16	0.00
80.94 _9+2018-10-25+20.16 (right side)	dark lit	185		64	329	56.2
80.94 _9+2018-10-25+20.16 (left side)	dark lit	144		120	184	78.3
I-80.94_150+2018-11-04+5.52	dark lit	19		31	50	38.0
65_2573+2018-11-04+5.32	dark	2		35	37	5.4
US-31_1252_2+2018-11-08+4.58	dark	4		32	36	11.1
I-74_990+2018-11-06+5.26	dark	N/A		N/A	N/A	#VALUE!
65_56+2018-11-07+18.11		N/A		N/A	N/A	#VALUE!
I-65_583+2018-11-02+21.22		N/A		N/A	N/A	#VALUE!
I-465_397+2018-11-03+5.48		N/A		N/A	N/A	#VALUE!

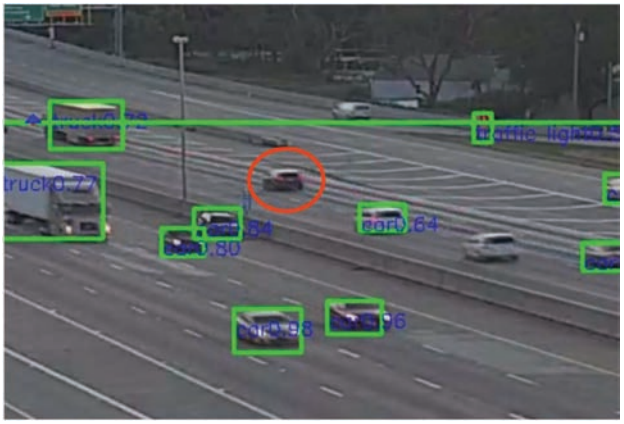

Figure 3.20 YOLOv3 missed vehicle detection.

the first is to try various less computational cost YOLOv3 implementation to reduce the hardware requirement. The second is to see if we can process multiple videos on one computer.

3.2.6 Traffic Information Obtained From a Camera

3.2.6.1 Find vehicle density. We have not worked on this yet in this project and will address this issue in the follow up project.

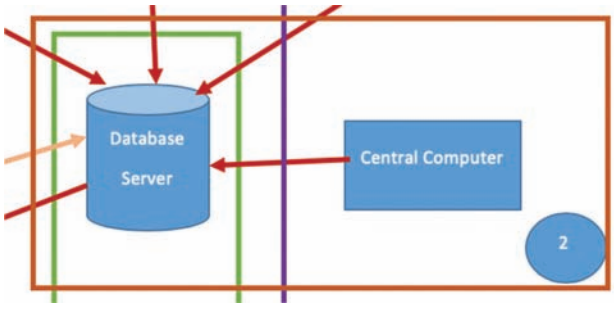

Figure 3.21 Central computer and database server.

3.2.6.2 Find the flow rate of each lane

1. The inputs are all vehicles passing through given horizontal reference line for each lane
2. The output is the flow rate per minute of vehicles passing through given horizontal reference line for each lane
3. The algorithm is:
    a. Maintain the vertical y coordinate at which the vehicles are going to be sampled (the horizontal line). A segment of the line Y = y inside of a lane
    b. We then consider the upper edge of each vehicle box in the current image. If it coincides with the line segment then the number of vehicles at this x value is incremented by 1, and this count is maintained as a running count per minute
    c. Find the moving average of the flow rate: The number of vehicles in each subsequent frame with time stamp is stored in a Flowrate object to calculate the flow rate

3.3 Traffic Incident Detection on the Central Computer

The decision-making part of the system is executed on Central Computer. This cyclical process runs indefinitely. Figure 3.21 shows the component related to the traffic incident detection.

3.3.1 Scope of the Decision-Making Software

- Determines the traffic incidents based on the traffic flow data reported by each camera
- Monitors the operation related to each traffic incident
- Updates the incident table in the database
- Determines the end of traffic incidences and remove them from the system
- Stores all traffic incidences related information in the database. The information includes the type of traffic incident, start time and duration of the incident, the video of the incident, the operation history of the incident, and effectiveness of the operation. The accumulated information of all incidences will be valuable data for improving incident prevention and handling

3.3.2 Decision-Making Process

1. Gather the adjacency relationship among all cameras in the database
2. Read traffic data periodically from the database
    a. As all video processing units in the field uploads the traffic information to the database every 5 seconds or as soon as an incident is identified, the Central Computer continuously check the database
    b. Analyze the data from the database and report any problems on the camera, or the field computer
    c. Calculate the flow rate difference between adjacent cameras and determine the incidences. A significant change in flow rate is defined as a>x% differences between two adjacent cameras (x will be determined)
3. What happens after Central Computer decides there is an incident?
    a. If the change in Flow Rate is >x%, it is suggestive of an incident. In this case, the incident table will be updated
    b. Whenever the incident table is updated, the activity listener on the Graphical User Interface will be triggered
4. Once a new entry (a new input to the Incident Table) pops up, the activity listener on the Graphical User Interface will catch this, and reflect the update on the Graphical User Interface's interactive map
5. Through the pop-up on the interactive map, the users are then alerted of the new occurrence of incidents, and they can act accordingly

4. DATABASE DEVELOPMENT

The automatic traffic incidents detection needs the information of location of the cameras, the adjacency graph of all cameras in the road network, number of lanes of each road, traffic directions of each lane, traffic flow rates for each road/lane, past incidents, and so on. This project uses a database as a central place to gather and distribute the information generated from camera images, the incident detection results derived from sensory information in the database, and to provide user interface information. Figure 4.1 shows the tables

Figure 4.1 ER-diagram of the designed database.

and relationship design of the system being developed. Some tables describe the property of the cameras and their installation information. Some tables store the real-time traffic flow information generated by the image processing computers on the field. Some tables store the incidents information generated from the traffic information. The database has been implemented in MySQL https://www.mysql.com/. We can convert to PostgreSQL https://www.postgresql.org/ if needed. The design of the database is still being updated during the project development.

This database has been hosted on the IU network. Unfortunately, the database server and web server were not well maintained and down frequently. TASI is in the process of setting up an account from a commercially supported cloud service to host the developed database and web servers.

5. WEB-BASED GRAPHICAL USER INTERFACE

A web-based Graphical User Interface (GUI) was developed with the input from INDOT. The web-based GUI reads data from the database as described in Section 4 and generate the user-interested information in an easy way to read output display. Figure 5.1 to Figure 5.3 show the finalized GUI. Figure 5.1 has four major components. The upper-left component shows all installed cameras on the Google map. The cameras at locations with incidents are shown in Red color. The component on the right side shows the camera location information. The component on the left-bottom is the real-time video of a focused incident. The middle bottom component is the location and traffic information of the focused incident. This window supports the following operations:

- By clicking camera on the camera symbol on the Google map, we can change the focus to a different camera location
- When a user clicks Incidents tab in the right-side component, the UGI switches to a screen as shown in Figure 5.3. The dynamic incident information is pulled from the database
- When a user clicks the real-time video component on the bottom left, the UGI switches to a screen as shown in Figure 5.2, which provides similar information as shown in Figure 5.1 but with larger sized video window
- When a user clicks the real-time video component in Figure 5.2, the UGI switches to a screen back to the window as shown in Figure 5.1

GUI has been successfully implemented and tested with small set of artificial data. First test was to retrieve incidents from the database for information from just one camera. Then we retrieved information by adding more cameras to the database. For the single camera test, we created an incident in the database to check if this incident is both reflected on an overlaid camera icon on the google maps API and reflected on the explorer list. We change the severity of the incident and check to see if both the respective camera icon and its entry in the explorer list change colors and the text info box reflects this incident and changes dynamically as well. When double clicking the camera icon, we also check to make sure that the full screen window of the camera feed for that camera enlarges and it is up to date. If the user tries to open a camera feed while another camera feed is open, check to make sure that the first camera feed is closed out or overwritten.

We repeated these tests for more cameras until all 350 cameras are linked. We also did the following tests:

- Checked to see that the website handles 3–8 Mb/s with just one camera streaming in conjunction with the constantly refreshing google map API
- Checked to see that this happens simultaneously with dynamically colored camera icons and explored incidents that are constantly changing

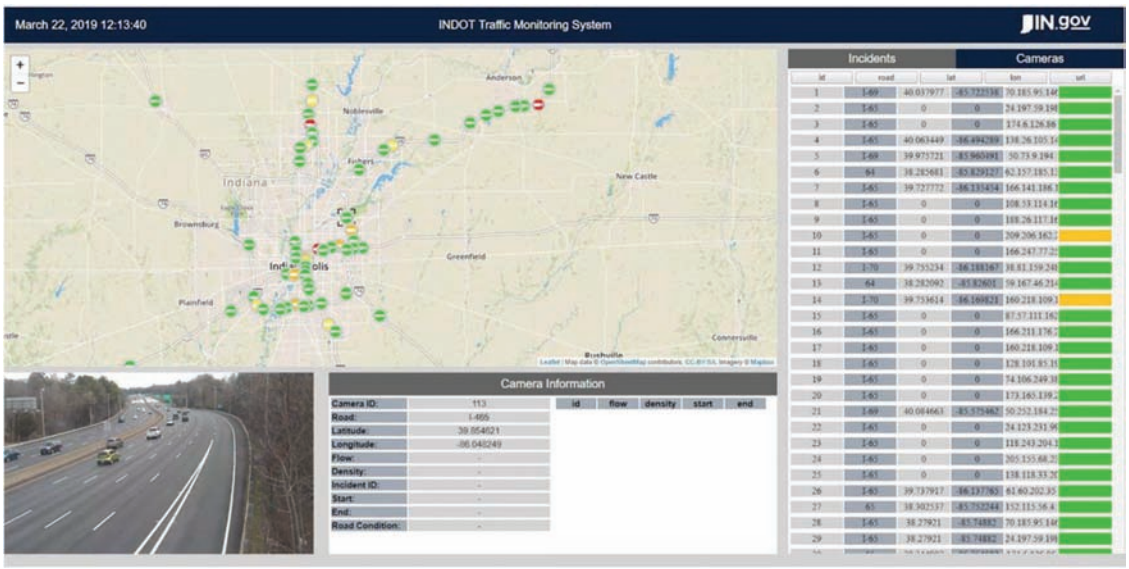

Figure 5.1 Focus on the camera list.

Figure 5.2 Enlarged video feed for specific camera.

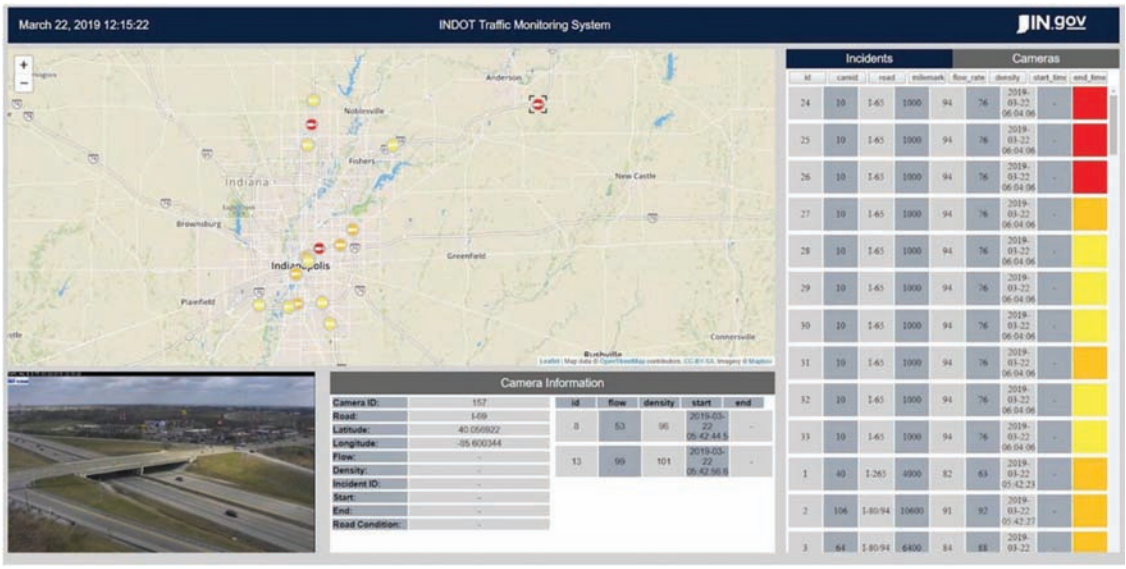

Figure 5.3 Focus on the incident list.

- Checked to make sure that the website doesn't crash if the user tries to load every incident since the beginning of time either in the main explorer or in the text info box's specific camera history explorer

The GUI and database passed the tests described above under the current circumstances.

6. CONCLUSIONS AND FUTURE WORK

In this project, we have developed the system architecture and implementation strategy for an automatic real-time system to monitor traffic conditions using the INDOT CCTV video feeds. More specifically, an AI based deep learning algorithm provided in YOLO3 was used for vehicle detection which generates good results during the daytime and well-lit area at nighttime. The tracking information of moving vehicles has been used to determine the locations of roads and lanes. A database was designed as the center place to gather and distribute the information generated from all camera videos. The database with a web-based Graphical User Interface (GUI) provides all information for the traffic incident detection. The algorithms for obtaining traffic flow information need to be further improved to provide more accurate results for future implementation in the field applications.

The team is currently in the process of integrating the prototypes of all components of the system together to establish a complete system prototype. In a follow up research project that has been approved by INDOT, the research team will continue to enhance the proposed methods and algorithms for extracting all relevant information and parameters from CCTV video data which are needed for the successful system implementation.

REFERENCES

Redmon, J., & Farhadi, A. (2017). YOLO9000: Better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, Honolulu, HI (pp. 7263–7271). https://doi.org/10.1109/CVPR.2017.690

Redmon, J., & Farhadi, A. (2018, April 8). *YoLOv3: An incremental improvement*. arXiv preprint arXiv:1804.02767. https://arxiv.org/abs/1804.02767

About the Joint Transportation Research Program (JTRP)

On March 11, 1937, the Indiana Legislature passed an act which authorized the Indiana State Highway Commission to cooperate with and assist Purdue University in developing the best methods of improving and maintaining the highways of the state and the respective counties thereof. That collaborative effort was called the Joint Highway Research Project (JHRP). In 1997 the collaborative venture was renamed as the Joint Transportation Research Program (JTRP) to reflect the state and national efforts to integrate the management and operation of various transportation modes.

The first studies of JHRP were concerned with Test Road No. 1—evaluation of the weathering characteristics of stabilized materials. After World War II, the JHRP program grew substantially and was regularly producing technical reports. Over 1,600 technical reports are now available, published as part of the JHRP and subsequently JTRP collaborative venture between Purdue University and what is now the Indiana Department of Transportation.

Free online access to all reports is provided through a unique collaboration between JTRP and Purdue Libraries. These are available at http://docs.lib.purdue.edu/jtrp.

Further information about JTRP and its current research program is available at http://www.purdue.edu/jtrp.

About This Report

An open access version of this publication is available online. See the URL in the citation below.

Chien, S., Chen, Y., Yi, Q., & Ding, Z. (2019). *Development of automated incident detection system using existing ATMS CCTV* (Joint Transportation Research Program Publication No. FHWA/IN/JTRP-2019/23). West Lafayette, IN: Purdue University. https://doi.org/10.5703/1288284317101