2003-27

Final Report

Development of Dynamic Route Clearance Strategies for Emergency Vehicle Operations, Phase I

MINNESOTA DEPARTMENT OF TRANSPORTATION

Research

Technical Report Documentation Page

1. Report No. MN/RC – 2003-27	2.	3. Recipients Accession No.

4. Title and Subtitle DEVELOPMENT OF DYNAMIC ROUTE CLEARANCE STRATEGIES FOR EMERGENCY VEHICLE OPERATIONS, PHASE I	5. Report Date June 2003
	6.
7. Author(s) Eil Kwon, Sangho Kim	8. Performing Organization Report No.
9. Performing Organization Name and Address Center for Transportation Studies University of Minnesota 511 Washington Ave. Minneapolis, MN 55455	10. Project/Task/Work Unit No.
	11. Contract (C) or Grant (G) No. (c) 81655 (wo) 3
12. Sponsoring Organization Name and Address Minnesota Department of Transportation Office of Research Services 395 John Ireland Boulevard Mail Stop 330 St. Paul, Minnesota 55155	13. Type of Report and Period Covered Final Report, 2003
	14. Sponsoring Agency Code

15. Supplementary Notes

http://www.lrrb.gen.mn.us/PDF/200327.pdf

16. Abstract (Limit: 200 words)

A route-based signal preemption strategy is developed to provide the most efficient and safe route for an emergency vehicle under a given network and traffic conditions. It combines an on-line route selection procedure and a dynamic sequential preemption method. The on-line route selection module first quantifies the level of congestion for each link on a given network using a congestion index and finds the least congested route for a given origin/destination pair using the well-known Dijkstra's algorithm. Further it also selects the safest signal phase for each intersection for a given travel direction of an EV. Once an emergency route is selected, the dynamic preemption module starts the preemption of the signals on the emergency route sequentially considering the location of the EV and the state of signal phase for each intersection. By sequentially preempting the traffic signals on a route with advance activation, the proposed strategy tries to clear the traffic queue for an EV approaching each intersection. The evaluation results with pre-specified emergency routes show 10 – 16% reduction of the emergency vehicle travel time for relatively long and/or complicated routes compared with the existing intersection-by-intersection preemption method. Further, the network-wide performance measures with the proposed dynamic preemption method were very compatible with those from the existing intersection-by-intersection clearance method.

17. Document Analysis/Descriptors Preemption Emergency Vehicle	Route-based	18.Availability Statement No restrictions. Document available from: National Technical Information Services, Springfield, Virginia 22161

19. Security Class (this report) Unclassified	20. Security Class (this page) Unclassified	21. No. of Pages 59	22. Price

DEVELOPMENT OF DYNAMIC ROUTE CLEARANCE STRATEGIES FOR EMERGENCY VEHICLE OPERATIONS, PHASE I

Final Report

Eil Kwon, Ph.D., P.E.
Center for Transportation Studies
University of Minnesota

Sangho Kim
Department of Computer Science
University of Minnesota

June 2003

Published by
Minnesota Department of Transportation
Office of Research Services
Mail Stop 330
395 John Ireland Boulevard
St. Paul, Minnesota 55155-1899

This report represents the results of research conducted by the authors and does not necessarily represent the view or policy of the Minnesota Department of Transportation and/or the Center for Transportation Studies. This report does not contain a standard or specified technique.

Table of Contents

List of Tables

List of Figures

Executive Summary

Most preemption systems developed to date operate on a single-intersection basis and require local detection of an emergency vehicle (EV) to activate a signal preemption sequence at each intersection. Such a local detection-based, intersection-by-intersection clearance strategy results in inherent time-delay at intersections, since the signal preemption procedure can start only after an emergency vehicle is detected. Further, in a heavily congested network, e.g., during peak-periods or after athletic events, the requirements of local detection for activating signal preemption present substantial problems in clearing the intersections quickly enough for an emergency vehicle to pass them without stopping or reducing its speed.

In this research a route-based signal preemption strategy is developed to provide the most efficient and safe route for an emergency vehicle under a given network and traffic conditions. It combines an on-line route selection procedure and a dynamic sequential preemption method. The on-line route selection module first quantifies the level of congestion for each link on a given network using a congestion index and finds the least congested route for a given origin/destination pair using the well-known Dijkstra's algorithm. Further it also selects the safest signal phase for each intersection for a given travel direction of an EV. Once an emergency route is selected, the dynamic preemption module starts the preemption of the signals on the emergency route sequentially considering the location of the EV and the state of signal phase for each intersection. By sequentially preempting the traffic signals on a route with advance activation, the proposed strategy tries to clear the traffic queue for an EV approaching each intersection. As soon as an EV clears an intersection, the signal recovery module starts the process to recover the original offset and cycle length of that intersection by adjusting the "Walk" time interval in each phase.

The route-based preemption method was evaluated using a microscopic simulation model using the University of Minnesota campus as the example network. For this study, the travel time data of emergency vehicles at the selected routes were collected in cooperation with the University Police Department. Due to the limitations of the simulation software, the on-line route-selection method developed in this study could not be tested in the current phase. The evaluation results with pre-specified emergency routes show 10 – 16% reduction of the emergency vehicle travel time for relatively long and/or complicated routes compared with the existing intersection-by-intersection preemption method. Further, the network-wide performance measures with the proposed dynamic preemption method were very compatible with those from the existing intersection-by-intersection

clearance method. The preliminary field-testing with off-the-shelf wireless communication devices with an in-vehicle Global Positioning System (GPS) unit shows the promising possibility of implementing the operator-selected route preemption strategy. Continuation of field-testing with an enhanced GPS and communication devices needs to be conducted.

1. INTRODUCTION

1.1 Problem Statement

Providing safe and fast driving environment for emergency vehicles, so that they can reach their destinations at the earliest possible time, is of critical importance in saving lives and reducing property loss. While substantial progress has been made in the areas of vehicle detection and communication technologies, current state-of-the-art in signal preemption in the U.S. has not reached the point where route-based signal clearance strategies can be automatically generated and implemented in real time. To be sure, most preemption systems developed to date operate on a single-intersection basis and require local detection of an emergency vehicle to activate a signal preemption sequence at each intersection. Such a local detection-based, intersection-by-intersection clearance strategy results in inherent time-delay at intersections, since the signal preemption procedure can start only after an emergency vehicle is detected. Further, in a heavily congested network, e.g., during peak-periods or after athletic events, the requirements of local detection for activating signal preemption present substantial problems in clearing the intersections quickly enough for an emergency vehicle to pass them without stopping or reducing its speed. Developing a route-based signal preemption method that can identify an optimal route for an emergency vehicle in real time and clear only those intersections on a given emergency route is of critical importance in emergency vehicle operations.

This report summarizes the results from the first phase of the current research effort to develop a route-based signal preemption strategy, which combines an on-line route selection procedure and a dynamic sequential preemption method to provide the most efficient and safe route for an emergency vehicle under a given network and traffic conditions. The proposed strategy was evaluated at the University of Minnesota campus network in Minneapolis, Minnesota, using a microscopic network simulation model by comparing its performance with that of the existing intersection-by-intersection preemption method.

1.2 Background

Existing signal preemption methods are in general classified into several categories depending on the technologies used for detecting emergency vehicles, i.e., optical, infrared light, acoustic, special types of loop detection, and GPS-based systems (1, 2). The optical systems, developed in the 1960's and the most commonly used ones in the field, use a strobe-lamp on the vehicle and an optical sensor per approach to an intersection requiring a clear line-of-sight path between the vehicle and the intersection (3). The sound-based systems use the directional microphones installed at an intersection to detect the siren of vehicles approaching a given intersection, therefore, no special equipment is required for the emergency vehicles (4). In radio-based systems, the directional signal for preemption can be transmitted from a vehicle to an intersection via one-way radio. The GPS approach uses the satellite-based Global Positioning System and determines the signal preemption time depending on the position, speed, and travel direction of the emergency vehicle approaching an intersection. In a GPS-based system being operated in Peoria, Illinois, both an emergency vehicle and an intersection are equipped with a GPS receiver and a radio transceiver for two-way communication (5). In an ongoing study by the City of Los Angeles, the feasibility of using a special loop sensor with transponders for emergency vehicle preemption is being tested (6). The above preemption systems adopt the intersection-by-intersection clearance approach based on local detection, and the impacts of such preemption strategy on the signal coordination and corridor-wide travel times of normal vehicles was first studied by Bullock, et. al. (1), who developed and applied a hardware-in-loop simulation-based evaluation procedure for the Route 7 Virginia corridor. A more recent study by Bullock et. al. (2), who used three controllers in a hardware-in-loop simulation system, found out that the second and third preempts in a peak period had significant impacts on queuing and delay for normal vehicles on a given network.

As indicated above, while there has been substantial progress in developing local preemption technologies, very few research results on the route-based dynamic preemption have been found in the literature. In fact, the only route-based signal clearance research found from the literature was the Fast Emergency Vehicle Preemption System (FAST) developed by a group of Japanese researchers (7); the detailed algorithm of FAST and its effectiveness over existing preemption strategies have not been published. Developing a route-based dynamic preemption strategy that can provide an efficient and safe traveling environment for emergency vehicles with minimum disruption on network traffic is of critical importance in managing urban traffic. In this paper, a route-based dynamic strategy for signal preemption is proposed and its effectiveness over the existing local-detection-based method is evaluated.

1.3 Research Objectives

The ultimate goal of the proposed research is to develop a dynamic route clearance system for efficient and safe operations of emergency vehicles. The specific objectives of the current project, Phase 1, include:

- Installation of a microscopic network simulation model for evaluating emergency vehicle signal preemption strategies.
- Modeling University campus network and calibration of the simulation model with real data to be collected in cooperation with University Police Department.
- Formulation of alternative route clearance strategies with different types of networking and preemption strategies.
- Evaluation of alternative strategies using simulation analysis.

Further, a preliminary effort to conduct a field-testing of a route-based signal preemption method was also conducted in this study.

1.4 Report Organization

Chapter 2 describes the simulation environment developed in this study to evaluate different types of signal preemption methods under realistic traffic conditions. It includes the qualitative description of the microscopic simulation model selected for this study and the summaries of different modules developed for simulating preemption strategies. The example network and the calibration results of the simulation model parameters are explained in Chapter 3. Chapter 4 summarizes the development and evaluation results of the route-based signal preemption in a simulated traffic environment. The framework for the preliminary field-testing system and initial test results with the operator-based route-preemption strategy are summarized in Chapter 5. Finally Chapter 6 includes conclusions and future research needs.

2. Development of Traffic Network Simulation Environment for Testing EVP Strategies

2.1 Current Status of Traffic Network Simulation Models

Evaluating emergency vehicle preemption (EVP) strategies under a realistic traffic environment prior to field implementation is of critical importance in developing robust EVP strategies. Such an off-line evaluation requires a traffic simulator that can realistically model the dynamic interaction between drivers, vehicles, and control systems in a given network geometry. In particular, a simulator needs to have the capability to model various types of signal preemption strategies as well as different types of sensing technologies that are needed to detect emergency vehicles. Microscopic simulation, which models the behavior of individual vehicles, in general provides the most detailed level of resolution in terms of estimating the effects of operational policies on network traffic performance, such as speeds, delays and queuing. In microscopic simulation, vehicles are generated following a pre-defined statistical distribution and moved according to a set of rules governing their behavior, e.g., car-following, lane-changing and gap acceptance, etc. While various types of microscopic models can be found in the literature, the currently operational models, which are commercially available and equipped with user-interfaces, include CORSIM, PARAMICS, VISSIM and AIMSUN. In this section, the major features of those models found from the literature are summarized.

CORSIM, developed and maintained under the sponsorship of FHWA, is still the most-widely used network simulation model in U.S. It is a part of Traffic Software Integrated Systems (TSIS), which also include TRAFED for input development and TRAFVU for 2-dimensional animation. According to the literature and the FHWA website regarding CORSIM (8, 9), the latest version, v5.0, can model a network that has up to 1000 nodes (500 on freeways and/or 500 on arterials) and 2000 links (1000 on freeways and/or 1000 on arterials). Further, the TSIS/CORSIM 5.0 provides a translator that converts a graphically edited network into a CORSIM input file. Its built-in control module can simulate different types of control devices such as stop or yield sign control, fixed-timing or actuated control, ramp metering and High Occupancy Vehicle (HOV)lane operations. It also provides a special interface to communicate with external control logic or programs.

PARAMICS, developed by Quadstone Ltd. in Scotland, is a suit of software tools for microscopic simulation. According to its website, the major features of the current version, v4.0, can be summarized as follows (10):

- The Modeler allows simultaneous network editing and simulating with 3-D visualization.
- The functionality of the current version of Modeler includes right-hand and left-hand drive capabilities, roundabouts, public transportation, car parking, incidents, truck-lanes and high occupancy vehicle lanes.
- There is virtually no limitations in network size and simulation time periods.
- A sub-network of an existing network can be selected and saved as a separate network.
- Its 3-D modeling function allows the specification of node elevation.
- The output analyzer provides both link-based and network-wide statistics.
- The application programming interface (API) allows user to model various types of traffic control strategies and devices as external modules.
- Traffic demand needs to be defined by a single or multiple matrices of origin-destination flow.

VISSIM is developed by Planung Transport Verkehr (PTV) in Germany. According to Fellendorf and Vortisch (11), it is based on a psycho-physical car-following model, which assumes that a driver can be in one of the four driving modes, i.e., free-driving, approaching, following and braking. It also adopts a rule-based algorithm for lane-changing. The major features of the current version, v3.7, found from its website are as follows (12):

- VISSIM can model integrated roadway networks found in a typical corridor as well as various modes consisting of general-purpose traffic, buses, HOV, light rail, heavy rail, trucks, pedestrians and bicyclists.
- ITS components and strategies that can be modeled include variable message signs, ramp metering, incident diversion, adaptive signal control, transit signal priority, dynamic lane control signs, etc.
- Its 3-D capability allows 3-D visualization of a network and 3-D vehicle animation.
- The output module provides link-based statistics with travel time data for pre-specified pairs.

- A network is modeled with links and connectors. There is virtually no limitation in network size and geometry types.
- Traffic volumes can be specified for pre-defined paths or by turning movements.
- It provides an application programming interface (API), which enables user to model various types of control devices and strategies as external modules.

AIMSUN, originally developed at the Universitat Politecrica de Catalunya in Spain, is currently being distributed by Transport Simulation Systems (TSS). According to Barcelo (13), AIMSUN uses an enhanced version of the Gipps car-following model. Specific enhancements include improved calculation of desired speed, accounting for grade effects in car following and taking into consideration the effects of adjacent vehicles. The major features of the current version, v4.1, found from its website (14) are summarized as follows:

- There is virtually no limitation in network size and shapes.
- Simulation can be either based on input traffic flows and turning proportions or based on O-D matrices and route selection models.
- A refined definition of parameters includes the category of local parameters to distinguish local properties from global ones. This facilitates the calibration process.
- It provides various options in terms of headway models including user-defined ones through the use of GETRAM Extensions.
- Its public transport function allows modeling of bus operations following pre-defined routes and time schedules.
- Its application programming interface allows user to model and simulate various types of control plans.
- Variable message signs and their influence on traffic behavior, e.g., re-routing or speed control, can be modeled.

2.2 Installation of a Network Simulator and Qualitative Testing

In this research, the VISSIM microscopic simulation model, version 3.6 (15), was selected and installed at the ITS Laboratory, University of Minnesota, as the platform for developing and evaluating the route-based signal preemption strategies for emergency vehicles. The major reason that VISSIM is selected for this study is its flexible structure, which allows the development of an external module that can control the state of every signal light for a given network through continuous interaction with the main simulation module of VISSIM. Therefore, the existing signal control and preemption procedures as well as any new strategies can be separately coded into external modules that continuously interact with VISSIM, which acts as the substitute of a real traffic environment and provides simulated detector data to external modules. The interaction between the main simulator and an external module can happen every 0.1 second with the current version.

First, a qualitative testing of VISSIM was performed to find out the capabilities and limitations of the simulator in terms of modeling and evaluating different types of emergency preemption strategies. The major findings from this qualitative testing include the following:

- The size of a network that can be modeled with VISSIM is practically limited by available memory in a computer.
- It is possible to place vehicle detectors at any location along a roadway and collect counts and occupancy/presence data for a pre-specified time interval.
- A special type of detector that detects only pre-specified types of vehicles can be installed at any location.
- A special type of vehicle can be generated from a pre-determined source at a pre-specified time instant, i.e., a vehicle can be designated as an emergency vehicle whose driving characteristics, e.g., maximum desired speed level, can be different from normal vehicles in a given network.
- A certain route can be designated for specific vehicles, i.e., a route can be specified for an emergency vehicle, which would travel only on a pre-specified route.
- However, those routes for special type vehicles need to be defined before simulation starts, i.e., a new emergency route cannot be inserted during a simulation,
- The driving performance of a designated vehicle, e.g., travel time for a certain route, can be generated as part of simulation output,

- While a special type of vehicle, which travels only a designated route with higher maximum desired speed levels, can be specified and generated at a specified time, the lane-changing behavior of those special type vehicles still follow the general behavior rules of normal vehicles in a network, i.e., they do not cross a center-line to pass slow moving vehicles.

As indicated above, while it is possible to generate an emergency vehicle during a simulation and to install a set of detectors that only detect emergency vehicles in a network, specifying a certain route for special type of vehicles must be done before a simulation starts with the current version of VISSIM. This indicates "generating an emergency route" randomly during a simulation is not possible with the current version. Therefore, in this research, only the performance of dynamic preemption strategies with pre-specified routes can be simulated and evaluated in a VISSIM network, i.e., the testing of the on-line route-selection module depending on network traffic conditions can not be conducted in an integrated manner.

2.3 Review of Advanced Transportation Controller Functionalities

In this study the functionalities of the Advanced Traffic Controller (ATC), whose standard specifications are still being developed, were reviewed as a potential implementation tool of the dynamic signal preemption strategies to be developed in this research. The development of ATC was initiated by the California Department of Transportation and the City of Los Angeles Department of Transportation in 1992. The main goal of this initial effort was to develop multitasking and multipurpose control equipment addressing the shortcomings of conventional signal controllers that were mainly single purpose control devices. In 1997, a steering committee, consisting of multiple agencies under the auspices of FHWA, finalized the hardware procurement specification, which became a part of the CalTrans specifications for the type 2070 controllers. According to Ghaman (16), the hardware architecture of the 2070 controller adopted a "shared hardware manager" concept to assure interchangeability and interoperability between controllers from various sources. The major components of a 2070 controller hardware included multiprocessor-design Central Processing Unit (CPU) with a Motorola 68360 processor, standard VME bus, asynchronous serial communication module, I/O module, and modular plug-in power supply. The OS-9 was selected as the operating system for the 2070 controller, whose software architecture includes hardware interface, standard/shared data modules, and application programs.

In July 1999, a formal agreement was reached among NEMA, Institute of Transportation Engineers (ITE), and AASHTO to jointly develop, approve and maintain an all-new ATC standard that will embody the best aspects of all controller technologies. The new standard would form a basis for a new controller family, which includes Type 2070 as one member. The first work of their effort produced 'ATC Standard Specification for the Type 2070 controller (17), which was a generic version of the CalTrans specifications for the 2070 controller. Currently a multi-agency cooperative effort, consisting of a steering committee and three working groups, is ongoing to develop a set of standards for the three main subsystems of ATC, i.e., Cabinet, Controller Unit, and Applications Programming Interface. The results from their initial work have been published at the website of Institute of Transportation Engineers (ITE). The rest of this section summarizes the major functionalities of the ATC extracted from the working documents found in the ITE website (18, 19):

- ATC is designed to provide an open architecture hardware and software platform for embedded applications that can implement multiple tasks simultaneously.
- Single ATC can control multiple intersections and the use of serial communication to multiple cabinet monitoring units allows response to a conflict at a single intersection without affecting other intersections, ramps, etc.
- In particular, the detectors can communicate to the controller in the form of either conventional contact closure or serial data string, which makes it possible to use 'smart' detectors that can pass additional information, such as vehicle classification and speed data.
- The CPU of the controller unit will be modular and interchangeable between manufacturers. It would allow substantial flexibilities in terms of updating or expanding system capability depending on required complexity of an application.
- The software system for ATC will consist of three distinctive parts: the Application, the Application Programming Interface (API), and the Operating System (OS). As shown graphically in Figure 2.1, the key component of the software architecture is the API, which can be considered as a software buffer between the application program and

CPU/OS combination. With the API that separates application software from particular CPU or OS, an independent and portable application can be developed.

- API is a source-code level interface defined by "C" function descriptions and header information. Application software developers compile and link their application with the appropriate API library, provided by manufacturers, for a target controller.
- The API architecture consists of two layers: Platform Interface and Application Environment Interface. The Platform Interface, Layer 1, defines the functional interface between application software and the discrete I/O device driver on an ATC. It encapsulates platform-dependent services that are specific to each ATC platform and provides the lowest common denominator to support common application software across a variety of ATC platforms. Layer 2, Application Environment Interface, defines the functional interface between application software and the discrete I/O device manager. It runs on top of Layer 1 and makes the software code directly portable to any ATC platform providing a compliant Layer 1.

It can be noted that the open software architecture of ATC with API allows development and implementation of various types of traffic control strategies that are mainly restricted by the number of inputs and outputs, i.e., available detection and devices to be controlled. The emergency preemption algorithm to be developed in this research takes advantages of the flexible features of the ATC software architecture.

2.4 Development of Virtual Controller and Signal Preemption Modules

In this section, a virtual controller is developed to implement and evaluate the route-based emergency vehicle preemption (EVP) strategies under a simulated environment. Figure 2.1 shows the structure of the virtual controller module, whose architecture is based on the advanced transportation controller reviewed in the previous section. The virtual platform interface consists of the function library provided by the VISSIM simulator. Using the VISSIM function library, virtual detector and traffic signal modules were developed to facilitate the implementation of a control application, e.g., the EVP algorithm to be developed in this study.

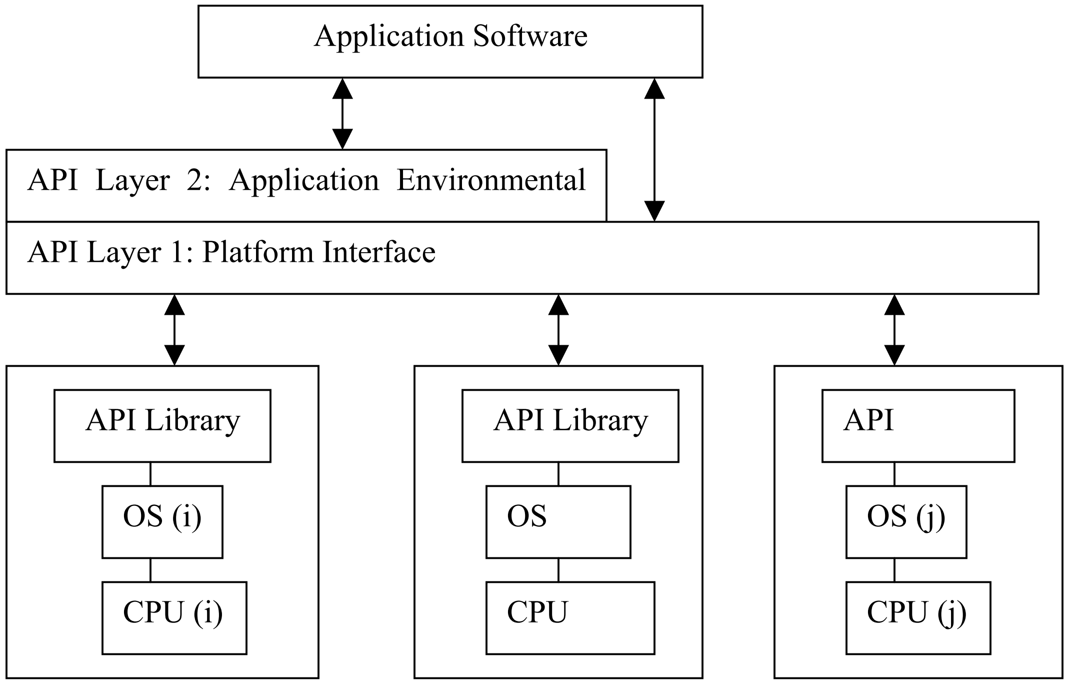


Figure 2. 1 Software Architecture of Advanced Transportation Controller

The interaction between the virtual controller and the network simulator is performed through an interface, which is a part of the virtual platform interface. The modular structure of the virtual controller with reusable virtual detectors/signals makes it possible to separate application development from the basic simulation environment, thereby allowing flexible and efficient development and implementation of different types of traffic control strategies without redundancy. In this study, the network simulator and the virtual controller reside in the same Windows-based personal computer.

Virtual platform interface consists of the following functions provided by the network simulator:

float Simulationssekunde (void);

Returns the current simulation second.

void VonLSA (int kommando, int nr, BOOLEAN uebergang);

Sends new signal state to signal group No. <nr>, depending on the code No.

unsigned char zaehlwert (int nr);

Returns the number of detected vehicle front ends since the last pass through the signal control.

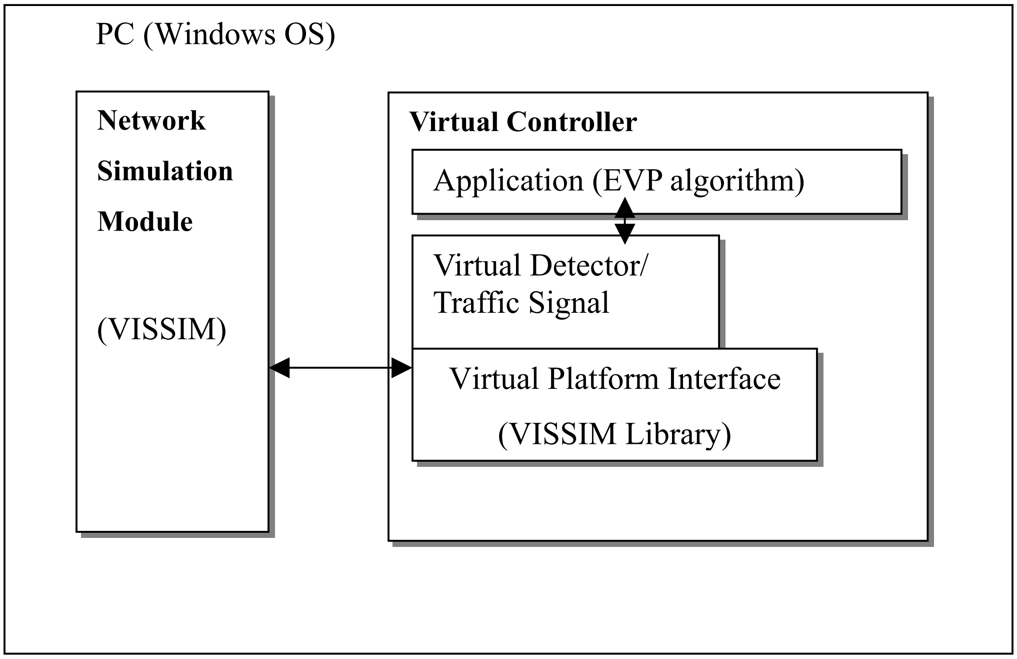

PC (Windows OS)
Network Simulation Module
(VISSIM)
Virtual Controller
Application (EVP algorithm)
Virtual Detector/ Traffic Signal
Virtual Platform Interface
(VISSIM Library)

Figure 2.2 Structure of Virtual Controller and Interaction with Network Simulator

Signal Preemption Module

The module to handle the sequence of signal preemption developed in this study follows the current signal preemption procedure being implemented in the City of Minneapolis after a traffic signal receives a preemption call from an emergency vehicle. Figure 2.3 shows the procedure whose main purpose is to provide safety to the pedestrians crossing an intersection. From the point of a pedestrian, a traffic signal consists of five operating steps: Walk, Flashing Don't Walk, Yellow, All Red and Red. As shown in the figure, the activation of signal preemption depends on the state of a signal when an emergency call is received as follows:

- Case 1: Complete "Minimum Walk" interval and go to "Flashing Don't Walk", continue to Yellow and "All Red" before switching to Green phase for Main Street, i.e., an emergency route for a given situation.
- Case 2: Go to "Flashing Don't Walk" immediately. Complete Yellow and "All Red" before switching to Green for Main Street.
- Case 3: Complete the current signal sequence for cross traffic before switching.

In this research, the above process is coded in C and incorporated into the virtual traffic signal module. Therefore, the route-based preemption module to be developed in this study determines the activation time for preemption at each intersection on an emergency route.

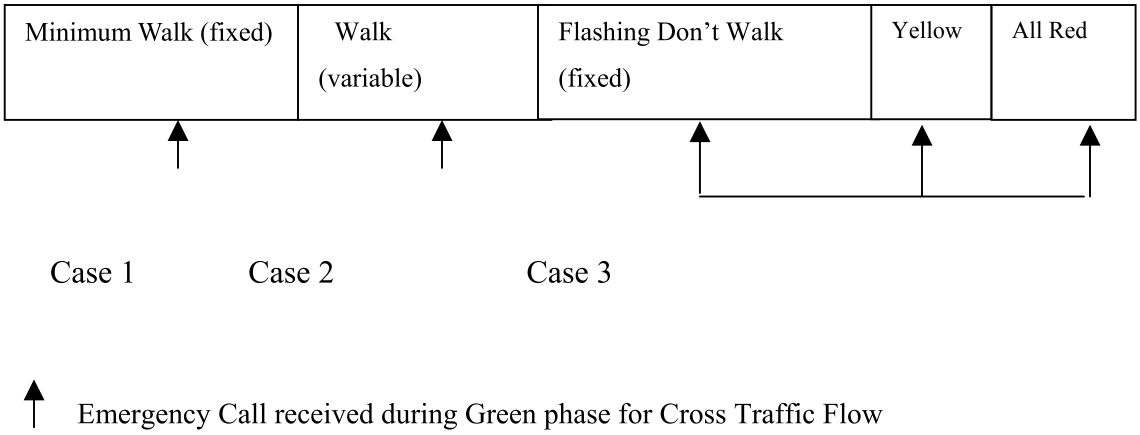


Figure 2.3 Different Cases Activating Preemption Sequence

2.5 Development of an Interface Between Virtual Controller and Network Simulator

The interface module between the network simulator and the virtual controller was developed using the built-in functions offered by the VISSIM developer. The current version provided by the developer for this study allows the interaction between the main simulation module and the external virtual controller every 1/10 seconds. Therefore the interface module continuously reads detector data from the simulator and sends them to the EVP module that determines appropriate signal phases and preemption activation times for all the intersections on an emergency route. The interface module also collects the signal phase data and sends them back to the main simulator every 1/10 seconds. The following functions of VISSIM are used to develop the interface module:

To receive detector information from the network simulator,

```
unsigned char  zaehlwert (int nr);
```

The return value is the number of vehicle front-ends passed over a detector since the last pass. (If a vehicle moves onto a detector while another one is still there, no new front end will be detected!)

To send traffic signal state from the Virtual Controller Module to signal group ‘nr’ in VISSIM,

```
void  VonLSA (int kommando, int nr, BOOLEAN uebergang);
```

where, “kommando” has the following values depending on signal state:

1: signal group green, 2: signal group red, 3: solid state signal on (green),
4: solid state signal or signal group off (black), 5: flashing signal on (amber),
6: flashing signal off (black), 7: signal group red/amber, 8: signal group amber,
9: signal group amber flashing, 10: signal group red flashing, 11: signal group green flashing,
12: signal group red-green flashing, 13: signal group green/amber,

If <uebergang> == TRUE, a transition state will be requested.

If signal group No. <nr> does not exist in the VISSIM network or if the code No. <kommando> is not in [1.8], a runtime error occurs.

3. Modeling Example Network and Calibration of Simulation Module

3.1 Sample Network and Data Collection

In this study, the University of Minnesota campus in Minneapolis, Minnesota, is used as the sample network, where the proposed preemption method was implemented and evaluated using microscopic simulation. Figure 3.1 shows the sample network with 33 signalized intersections that are currently operated in a pre-timed, offset-based coordinated mode. The geometric data for the campus network was collected from the aerial photos purchased from the Metropolitan Council, Minnesota, while the detailed traffic data for the intersections in the network, including traffic volume, signal timing plans, preemption sequence, and the parameters for existing preemption systems, were provided by the Traffic Operations Center (TOC), City of Minneapolis. Further, to calibrate the simulation model, the travel time data of emergency vehicles in the sample network with existing preemption systems were also collected during two afternoon periods on two weekdays in cooperation with the University of Minnesota Police Department. Figure 3.1 also includes the three routes, i.e., Route 1-3, where the emergency vehicle travel time data were collected for this study. Most of the signalized intersections on Route 1 and 2 have the optical preemption systems, while none of Route 3 has any preemption device. For each route, a total of four test-runs were made with a police car equipped with the light-beam emitter. Table 3.1 shows the emergency travel time data for different routes collected from the example network.

3.2 Modeling Intersection-based Signal Preemption Procedure with Virtual Controller

In this research, the existing intersection-by-intersection preemption strategy was also modeled and coded into an external controller module. To simulate existing light-beam or sound-based preemption systems with different detection ranges, each link in the sample network is equipped with the loop detectors installed every 15 meters. Therefore, depending on the detection range of a given detection system, a particular detector set can be programmed to detect only emergency vehicles. The distance between the detector set for emergency vehicles and an intersection can be adjusted to reflect the detection range of a given preemption system. If an emergency vehicle passes over one of those detectors designated for preemption detection for an intersection, then the intersection activates the preemption sequence described in the previous chapter.

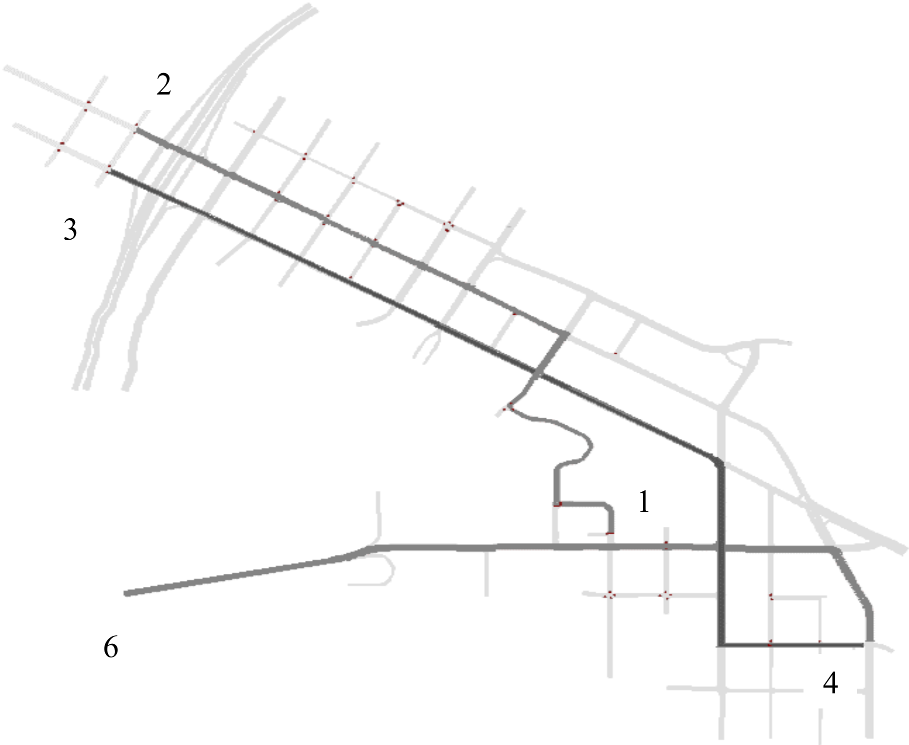


Figure 3.1 Example Traffic Network for Evaluating EVP Strategies

Table 3.1 Emergency Vehicle Travel Time Data for Selected Routes

.

Route	1st round	2nd round	3rd round	4th round	Speed range of EV
□→□	2 min 14 sec	1 min 57 sec	2 min 00 sec	1 min 47 sec	50 ~ 80 km/hr
□→□	2 min 07 sec	2 min 11 sec	2 min 15 sec	2 min 01 sec	50 ~ 100 km/hr
□→□	1 min 41 sec	1 min 35 sec	1 min 32 sec	1 min 33 sec	50 ~ 130 km/hr

3.3 Calibration of Simulation Model for Intersection-based Signal Preemption

Using the travel time data of emergency vehicles collected in the sample network, the simulation software was first calibrated to make the simulation model reflect the real traffic environment as much as possible. For this calibration, the existing intersection-by-intersection preemption strategy being implemented in the sample network was simulated for the three routes where the emergency vehicle travel time data were collected. The resulting simulated travel time data of the emergency vehicles were compared with the real data and the difference was minimized by adjusting the speed range of an emergency vehicle during simulation and the detection distance of the existing preemption system in the sample network. Specifically, the speed profile of an emergency vehicle was set to 60 – 130 km/hr, while the detection range was determined as 200 – 300 meters. Table 3.2 shows the simulated and actual travel time data of an emergency vehicle for the three routes in the sample network. To reflect the effects of stochastic simulation, five random seeds were used for each case and their results were averaged.

Table 3.2 Calibration Results of the Simulation Model

Route	Before calibration	After calibration	Averaged actual travel time of EV
□→□	2 min 42 sec	2 min 01 sec	2 min 00 sec
□→□	3 min 37 sec	2 min 00 sec	2 min 09 sec
□→□	2 min 29 sec	1 min 49 sec	1 min 35 sec

4. Development and Evaluation of Route-based Emergency Vehicle Preemption Strategies

4.1 Overview of Route-based EVP Strategy

Figure 4.1 shows the simplified structure of the route-based dynamic signal preemption strategy developed in this study. The network-monitoring module continuously collects traffic data from field detectors and quantifies travel cost of each link, i.e., a section of roadway between two intersections, by combining its length and current congestion level. When an emergency call is received at the control center and after the current location of an available emergency vehicle (EV) is identified, the route-selection module determines the quickest route between the current location of the EV and a given destination. In this research, the well-known Dijkstra's algorithm (20) is adopted to find the optimal route that has the minimum travel cost under current traffic conditions for a given origin/destination pair. Alternatively, an emergency-vehicle operator, e.g., police officer, can determine an emergency route based on his/her preference and local knowledge. Once an emergency route is selected, the dynamic preemption module starts the preemption of the signals on the emergency route sequentially considering the location of the EV and the state of signal phase for each intersection. As soon as an EV clears an intersection, the signal recovery module starts the process to recover the original offset and cycle length of that intersection by adjusting the "Walk" time interval in each phase. The proposed strategy assumes that the two-way communication between an emergency vehicle and the control center is available and the location of the emergency vehicle can be detected at the control center with a GPS or loop-transponder-based systems. Further, the preemption sequence at each intersection can be activated at the control center. Therefore, the focus of this study is to evaluate the potential effectiveness of a route-based dynamic preemption method over existing intersection-by-intersection preemption strategies. The rest of this section describes the major features of each module.

4.2 On-line Route Selection Module

Once the current location of an emergency vehicle and its destination is determined, the route-selection module determines the best route that has the minimum travel cost for a given origin/destination pair under current traffic conditions. Figure 4.1 shows the major sub-modules in the on-line route selection module including link travel cost estimation, selection of minimum cost route for a given origin/destination of an emergency vehicle and finally selection

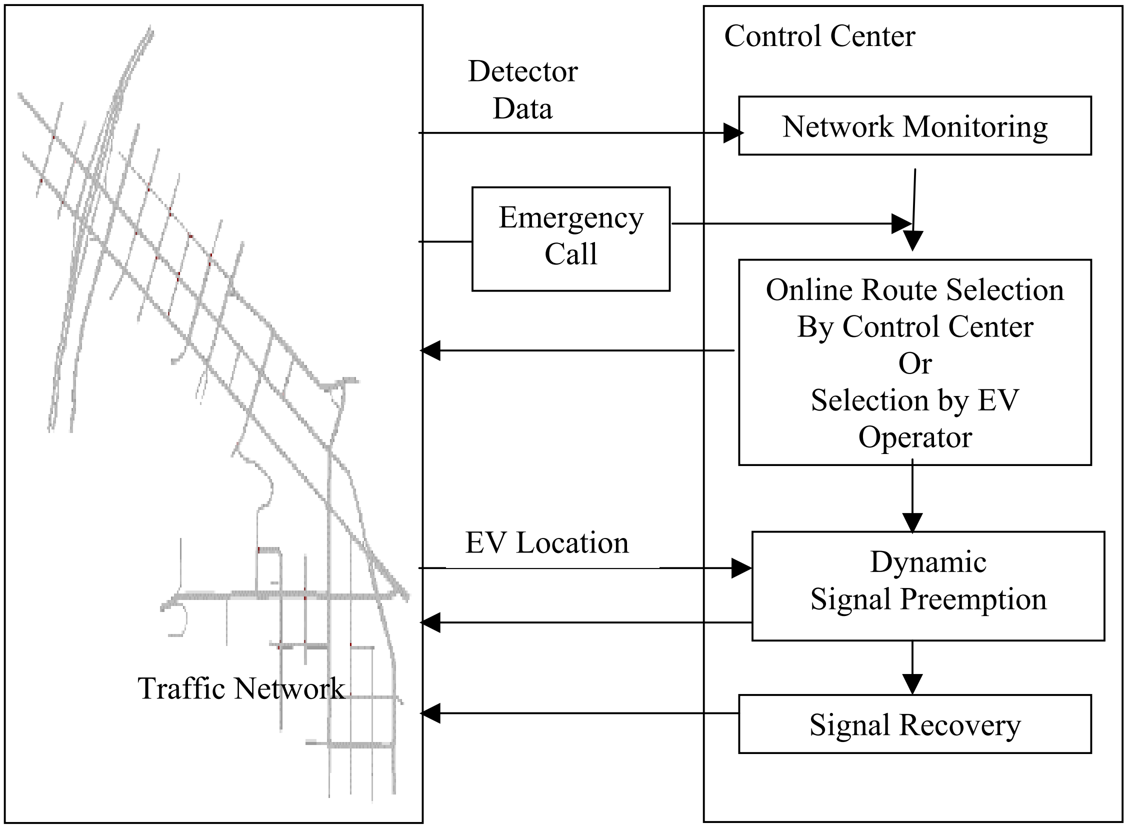


Figure 4.1 Simplified Structure of Route-based Emergency Vehicle Preemption Strategy

of a safe phase at each intersection on an emergency route. In the proposed method, a network is represented as a set of links/nodes and the well-known shortest-path algorithm developed by Dijkstra (20) is adopted to find the quickest route. The Dijkstra's algorithm has been proven to result in the shortest-path from a single source on a weighted directed graph, where all edge weights have nonnegative values (21). In this research, a given network is modeled as a set of directional links with nonnegative travel costs and the Dijkstra's algorithm is applied to find the minimum travel cost route from the current location of an emergency vehicle to its given destination. Further, to ensure the safest traveling environment for an emergency vehicle during the preemption period, the proposed method also selects a specific phase combination for each intersection when multiple options exist in terms of available phases. Therefore, the output from the on-line route selection module includes both the minimum travel-cost route and the specific signal phase combination at each intersection during preemption.

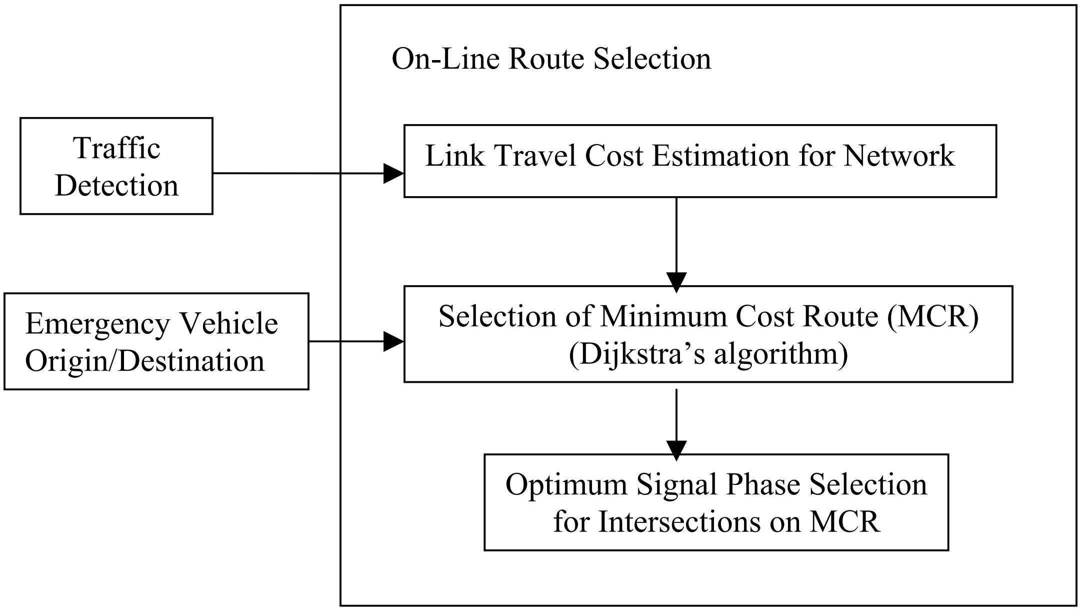

On-Line Route Selection
Traffic Detection
Link Travel Cost Estimation for Network
Emergency Vehicle Origin/Destination
Selection of Minimum Cost Route (MCR) (Dijkstra's algorithm)
Optimum Signal Phase Selection for Intersections on MCR

Figure 4.2 Structure of On-Line Route Selection Module

Quantification of Link Travel Cost through Time

The travel cost of each link is quantified with the volume and presence data collected from the loop detectors at each link by the network-monitoring module through time. In this research, the travel cost of link i during time interval k, $T_{i,k}$, is modeled as a function of the length of link i and its congestion level during k, i.e.,

$T_{i,\ k} = L_i\ (1 + C_{i,k})$

where, L_i = length of link i

$C_{i,\ k} = \ \Sigma\ \beta_j\ (P_{j,k} + V_{j,k})/(1 + Vj_{,k})$.

$V_{j,k}$ = number of vehicles passed detector j in link i during k,

$P_{j,k}$ = 1.0 if detector j is occupied by a vehicle at the end of k,

0.0 else.

β_j = weight for detector j in link i, $\Sigma\ \beta_j = 1.0$

In the above formula, $C_{i,k}$ represents the congestion level of link i during k on a 0 to 1.0 scale using only volume and presence detection commonly available from loop detectors. It has been successfully used as an index that quantifies level of congestion at each link for intersection signal control (22, 23). Further, it is also possible to add certain physical characteristics of a route, such as number of turns, in the above travel cost function to reflect safety concerns of emergency-vehicle operators.

Modeling Dijkstra's Algorithm

Dijkstra's algorithm solves the single-source, minimum-cost-path problem on a weighted, directed graph G = (V, E) for the case in which all edge weights, i.e., link travel costs, are nonnegative. Dijkstra's algorithm maintains a set S of vertices whose final minimum-cost-path weights from the source s have already been determined. The algorithm repeatedly selects the vertex $u \in V - S$ with the minimum-cost-path estimate, adds u to S, and relaxes all edges leaving u. In the following implementation, we use a min-priority queue Q of vertices, keyed by their d values. The pseudo code for Dijkstra's algorithm can be written as follows:

```
DIJKSTRA(G, w, s)
INITIALIZE-SINGLE-SOURCE(G, s)
S ← 0
Q ( V[G]
4 while Q ( 0
5       do u ( EXTRACT-MIN(Q)
6             S ( S ( {u}
7             for each vertex v ( Adj[u]
8                   do RELAX(u, v, w)
```

Line 1 performs the usual initialization of d and (values, and line 2 initializes the set S to the empty set. The algorithm maintains the invariant that Q = V – S at the start of each iteration of the while loop of lines 4-8. Line 3 initializes the min-priority queue Q to contain all the vertices in V; since S = 0 at that time, the invariant is true after line 3. Each time through the while loop of line 4-8, a vertex u is extracted from Q = V – S and added to set S, thereby maintaining the invariant. (The first time through this loop, u = s.) Vertex u, therefore, has the smallest shortest path estimate of any vertex in V – S. Then lines 7-8 relax each edge (u, v) leaving u, thus updating the estimate d[v] and the predecessor ([v] if the minimum-cost-path to u can be improved by going through u. Observe that vertices are never inserted into Q after line 3 and that each vertex is extracted from Q and added to S exactly once, so that the while loop of lines 4-8 iterates exactly |V| times.

The sequential step-by-step process to find a minimum-cost-path is illustrated in Figure 4.3 for a simple network. The source s is the left lower corner vertex. The minimum estimates are shown within the vertices, and thick edges indicate predecessor values. Black vertices are in the set S, and white vertices are in the min-priority queue Q = V – S.

(a): The situation just before the first iteration of the **while** loop of lines 4-8. The shaded vertex has the minimum d value and is chosen as vertex u in line 5.

(b)-(f): The situation after each successive iteration of the **while** loop. The shaded vertex in each part is chosen as vertex u in line 5 of the next iteration. The d and π values shown in part (f) are the final values.

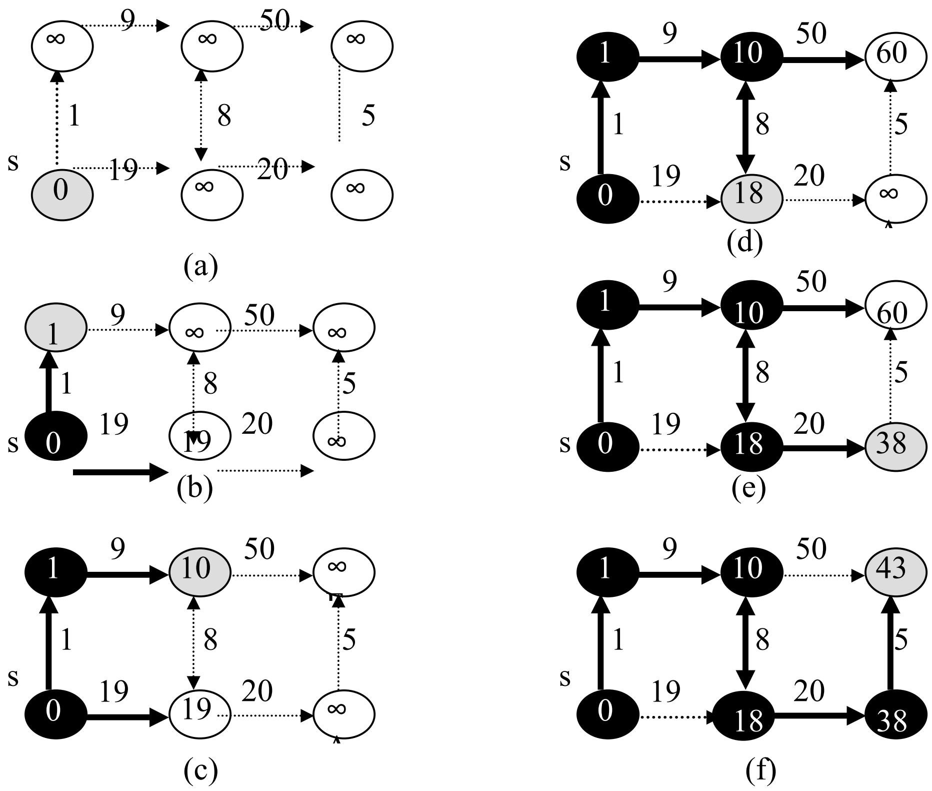


Figure 4.3 An Example of Dijkstra's Algorithm

Modeling Route-Selection Algorithm with VDM-SL

The on-line route selection algorithm described above is modeled using an abstract modeling language called Vienna Development Method (VDM-SL), which has been recently standardized by the International Organization for Standardization. The VDM-SL enables a system developer to directly implement the core concept of a system model into a set of executable code that maintains the structure of a system in question. This allows quick testing of the validity of the core algorithm of a system model without going through detailed design of a system model, thus saving substantial amount of programming time. The on-line route selection algorithm implemented in VDM-SL has a hierarchical data structure containing traffic elements. The highest level of record type is Network which represents a traffic network. The Network contains a sequence of intersections. Record type of each Intersection contains a sequence of outgoingLinks and a sequence of incomingLinks. Record type of each Link has nonnegative weight. These are the skeletal structure of the route-selection model. Here is a part of source code for above data structure.

```
Network ::       intersections : seq of Intersection
                 Q : seq of IntersectionId
                 current : [IntersectionId];

Intersection ::  id : IntersectionId
                 d : nat
                   pi : IntersectionId
                   phase : [Phase]
                   outgoingLinks : seq of Link
                   incomingLinks : seq of Link
                   phases : set of Phase;

Link ::          weight : nat
         fromI : IntersectionId
         toI : IntersectionId
         linkDirection : LinkDirection;
```

In the definition of a traffic network, sequence is used instead of set to use the iterative property of sequence for the implementation even though the order of elements is conceptually not important. Intersections in our traffic network model are applicable to the vertices in the Dijkstra’s algorithm, while links are also applicable to the edges. Thus a traffic network itself is equivalent to the weighted and directed graph of Dijkstra’s algorithm.

We implemented INITIALIZE-SINGLE-SOURCE routine by manipulating the case file data. Source intersection is defined by setting the d value of an intersection 0 in the case file. All other intersections are given value 999 which is equal to infinitive value in Dijkstra's algorithm. All the intersections have pi value as '-' in the initial case file. The min-priority queue in Dijkstra's algorithm is equal to Q in the record type Network. The case file initializes the Q with all the intersections' id.

There are two important functions for route selection. One is ExtractMin and the other is Relax. The underlying principles of these two functions are same as described in section 2.3. The ExtractMin extracts an intersection whose d value is minimum from min-priority queue Q in the Network. An auxiliary function FindMin finds the intersection in min-priority queue Q of intersections whose d value is minimum. Then the ExtractMin function extracts the result from the FindMin from Q in the Network. Here is a part of source code for ExtractMin function.

```
ExtractMin : Network -> Network
ExtractMin(network) ==
        let min = FindMin(network) in
                mk_Network(
                        network.intersections,
                    [network.Q(i) | i in set inds network.Q & network.Q(i) <> min],
                    min)
pre pre_FindMin(network);
```

The Relax function reduces d value of intersection. This function receives a link and traffic network as input parameters. A link is directed and related with two intersections; starting intersection and ending intersection. This function compares ending intersection's d value with the sum of starting intersection's d value and the link's weight. Only when the sum value is less than or equal to the ending intersection's d value the function updates the ending intersection's d value with the less value. This Relax function is applied for all the outgoing links of an intersection by ApplyRelax function. We only converted for loop in the pseudocode to recursive function because VDM-SL did not support loop control logic. Here is a part of source code for Relax function.

```
Relax : Link * Network -> Network
Relax(link, network) ==
        if GetIntsn(link.fromI, network).d + link.weight
                <= GetIntsn(link.toI, network).d
        then
                let new_i = mk_Intersection(
                        link.toI,
                        GetIntsn(link.fromI, network).d + link.weight,
                        link.fromI,
                        nil,
                        GetIntsn(link.toI, network).outgoingLinks,
                        GetIntsn(link.toI, network).incomingLinks,
                        GetIntsn(link.toI, network).phases) in
                mk_Network( network.intersections
                        ++ {GetIntsnIndex(link.toI, 1, network) |-> new_i},
                        network.Q,
                        network.current)
        else
                mk_Network( network.intersections,
                        network.Q,
                        network.current);
```

Signal Phase Selection Algorithm

Providing the safest signal phase for an emergency vehicle on an emergency route is of critical importance for efficient emergency operations. To simplify modeling, eight basic signal phases are used in our model. Figure 4.4 shows those eight phases. The signal phase selection algorithm is based on the concept that there exist priorities between signal phases in terms of safe passage of an emergency vehicle. For example, if an emergency vehicle receives a signal phase which has left turn and thru movements at the same time, it can be considered as the safest phase for the emergency vehicle regardless of its direction, since no other vehicles can pass the intersection except emergency vehicle during the preemption period. Phase 4+7, phase 3+8, phase 2+5 and phase 1+6 are applicable to this type of phase. Second type of phase has left turn only signals in both directions. This phase is safe for a left turning emergency vehicle. In addition, the left turning vehicles from the opposite link can pass the intersection during the preemption. Phase 3+7 and phase 1+5 are applicable to this type of phase. The remaining signal phases have an unprotected left turn. The unprotected left turn may cause dangerous situation because a careless driver who does not recognize the emergency vehicle coming in the opposite direction may enter the intersection to make a left turn. In the other case the left turning emergency vehicle may wait until the opposite thru vehicles pass the intersection. Thus, they are classified as unsafe phases. Phase 4+8 and phase 2+6 are applicable to this type of phase.

The priority of signal phase for safe operations of an emergency vehicle can be defined as follows depending on the direction of an emergency vehicle (> means safer phase):

From north to south thru:	phase 4+7 > phase 4+8
From north to east left turn:	phase 4+7 > phase 3+7 > phase 4+8
From south to north thru:	phase 3+8 > phase 4+8
From south to west left turn:	phase 3+8 > phase 3+7 > phase 4+8
From west to east thru:	phase 2+5 > phase 2+6
From west to north left turn:	phase 2+5 > phase 1+5 > phase 2+6
From east to west thru:	phase 1+6 > phase 2+6
From east to south left turn:	phase 1+6 > phase 1+5 > phase 2+6

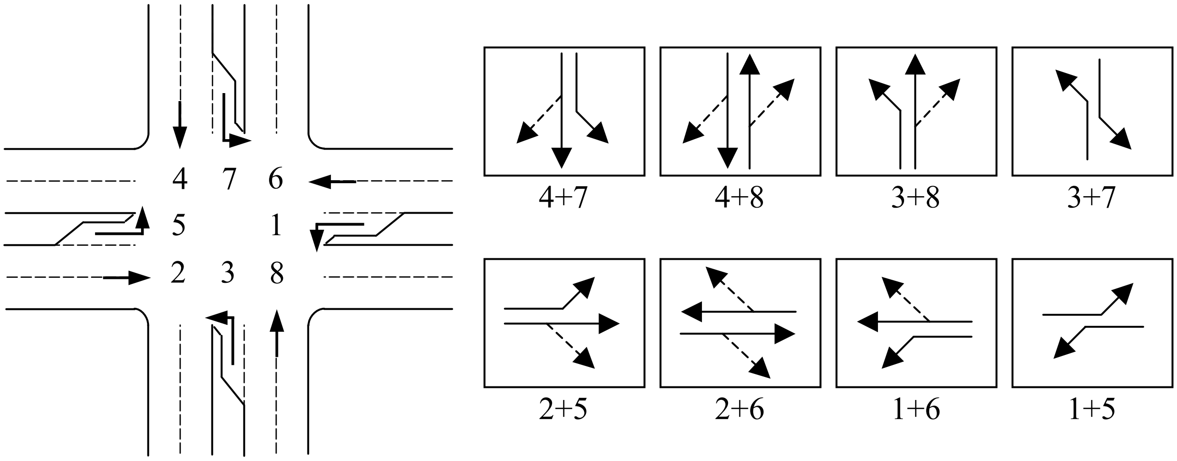


Figure 4.2.3 Definition of Traffic Signal Phases

Modeling Phase Selection Algorithm with VDM-SL

All the phases described above are defined in the source code as a type Phase. The record type Intersection has a set of phases whose type is Phase. There is one important function for signal preemption. It is DecidePhase. The function receives incoming link, outgoing link and intersection as input parameters. Then it applies signal preemption rules mentioned in the above section and returns the optimized phase for an intersection. This function is applied to all the intersections on the quickest route by ApplyDecidePhase function.

```
DecidePhase : Link * Link * Intersection -> [Phase]
DecidePhase(incomingLink, outgoingLink, intsn) ==
        if incomingLink.linkDirection = <d2> and
```

```
outgoingLink.linkDirection = <d2> and
        <p25> in set intsn.phases then
        <p25>
else if incomingLink.linkDirection = <d2> and
outgoingLink.linkDirection = <d2> and
        <p26> in set intsn.phases then
        <p26>
else
…
else nil;
```

Integration of Route Selection and Phase Selection Modules

Figure 4.5 shows the combined structure of the route-selection module implemented in VDM-SL. The function Dijkstra, consisting of ExtractMin and ApplyRelax, is one iteration of 'while loop' in the pseudo-code of the Dijkstra's algorithm. The function ExtractMin extracts an intersection whose d value is minimum from min-priority queue Q in the Network. The function ApplyRelax is equivalent to the for loop in the Dijkstra's algorithm executing the function Relax for outgoing links of the intersection from ExtractMin. The function ApplyDijkstra is equivalent to the while loop in Dijkstra's algorithm executing the function Dijkstra until the min-priority queue Q is empty. After executing the ApplyDijkstra, we can get the processed network containing intersections with final d values, the minimum travel cost from the source intersection.

The GetQuickestRoute extracts a sequence of intersections resulting from the ApplyDijkstra. The input to GetQuickestRoute includes origin/destination intersections and a processed network, and its output is the minimum cost route. ApplyDecidePhase applies signal phase selection algorithm to each intersection on the minimum cost route in a recursive way. DecidePhase selects the safest signal phase for an intersection with the input information on incoming and outgoing links.

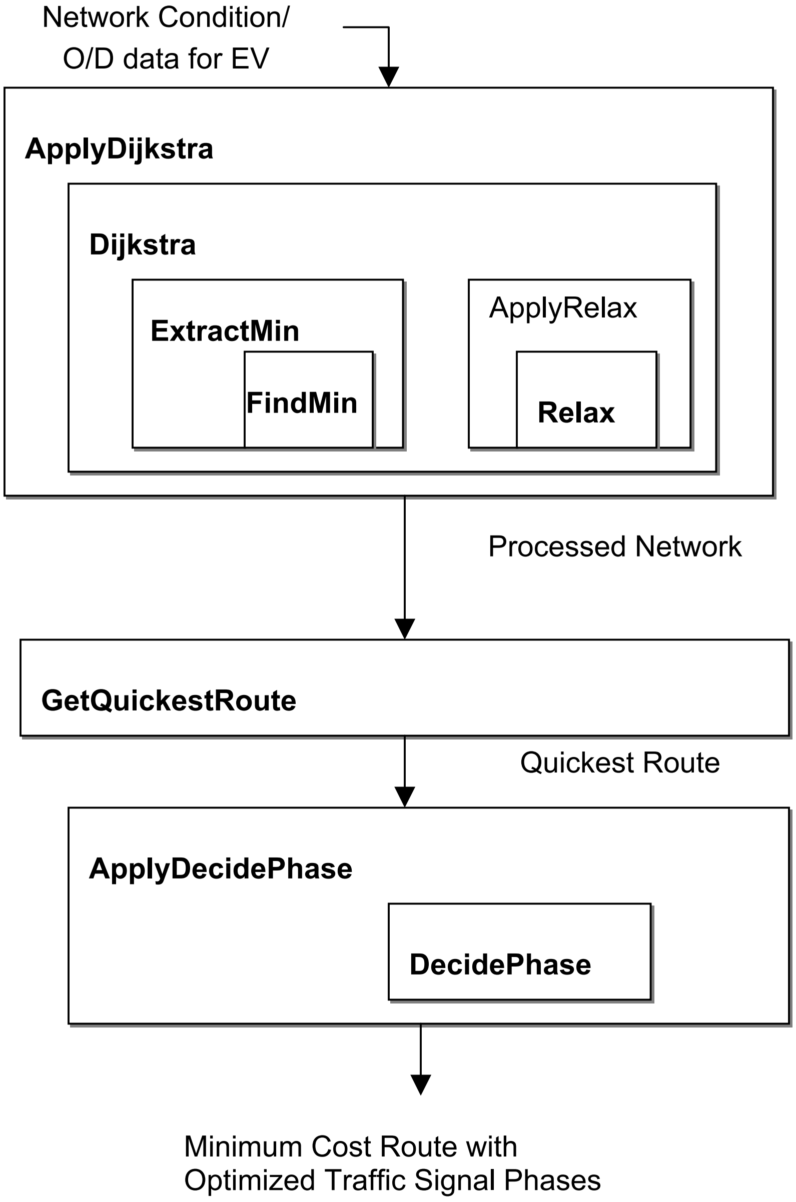


Figure 4.5 Combined Structure of Route-Selection Module

Testing Route-Selection Module

To test the route-selection module developed in VDM-SL, a hypothetical network with non-negative link weights is constructed as shown in Figure 4.6.

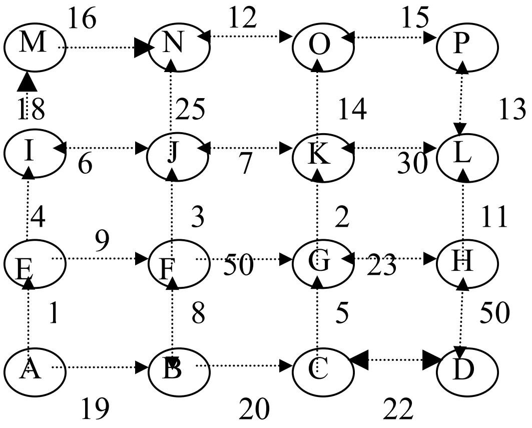


Figure 4.6 Test Network

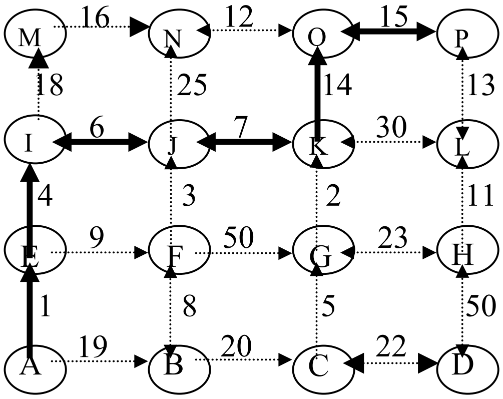
To run a sequence of functions for route-selection, we made a function Run. A signature of the function is as follows.

```
Run : Network * IntersectionId * IntersectionId -> seq of char
```

This function receives network, source intersection id and destination intersection id as input parameters. Then it applies all the functions mentioned in the above sections and prints the result. The results from Run function to find the quickest route with optimized signal phases from 'A' to 'P' are as follows:

```
vdm> print Run(network, 'A', 'P')
"[A-nil] -> [E-p48] -> [I-p48] -> [J-p25] -> [K-p25] -> [O-p38] -> [P-nil]"
```

Figure 4.7 shows the above results in a graphic mode with the selected quickest route for an emergency vehicle and the signal phases for each intersection. For example, intersection K has <p15>, <p25>, <p26>, <p48> phases. Possible phases are <p15>, <p25>, <p26> for an emergency vehicle to make a left turn. Among them the selected phase is <p25>, which is the safest phase for a left turn.



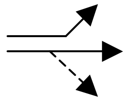
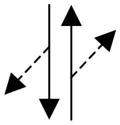
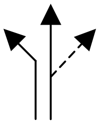
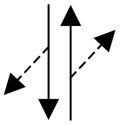
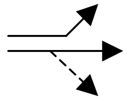
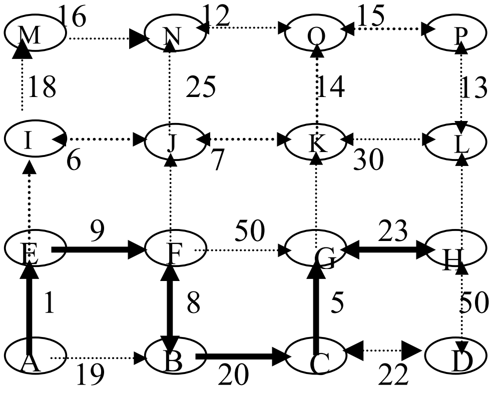
A	nil	K	
E		O	
I		P	nil
J			

Figure 4.7 Test Results for ‘A’ to ‘P’

The following example shows another result of Run function to find the quickest route with optimized signal phases from ‘A’ to ‘H’. The results are also represented in Figure 4.8.

```
vdm> print Run(network, 'A', 'H')
"[A-nil] -> [E-p48] -> [F-p26] -> [B-p47] -> [C-p26] -> [G-p48] -> [H-nil]"
```



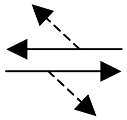
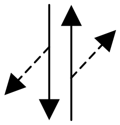
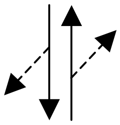
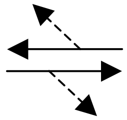
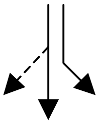
A	nil	C	
E		G	
F		H	nil
B			

Figure 4.8 Test Results for ‘A’ to ‘H’

4.3 Dynamic Signal Preemption Strategies

Determining the right time to activate the signal preemption sequence for the intersections along the emergency route is of critical importance in reducing travel time of emergency vehicles and minimizing negative effects of signal preemption on normal traffic flow. Once the best route is determined for a given emergency situation, the dynamic preemption module sequentially activates the preemption procedure for the intersections on the route depending on the direction and location of an emergency vehicle. The amount of time for the preemption procedure to be activated for an intersection, $T_{act,l}$, varies depending on the specific state of a signal phase for a cross street when an emergency call is received, i.e.,

$$0 <= T_{act,l} <= P_{max,l}$$

where, $P_{max,l}$ is the maximum total time to complete the preemption sequence for intersection l after an emergency call is received at the beginning point of the green phase for its cross street, i.e., Minimum Walk, Flashing Don't Walk, Yellow and All Red. In Minneapolis, P_{max} generally equals to 20 seconds for most intersections. Therefore, the optimal amount of time needed to activate the preemption sequence at an intersection l to provide the "best" traveling environment for an emergency vehicle to clear the intersection can be considered as

$$P_{max,l} \pm w_{t,l}$$

where, $w_{t,l}$ can vary through time depending on several factors including the speed of an emergency vehicle, the status of the signal phase and traffic condition at intersection l at time t.

Single/Variable Point Activation

Since it would be extremely difficult to calculate the optimum value of $w_{t,l}$ in real time for each intersection, in this research a simplified approach is developed to determine the activation point for each intersection by assuming a fixed value for $w_{t,l}$, which can be selected considering the average speed level of an emergency vehicle, u_{EV}, and the value of P_{max} for a given network. The location of two potential activation points from the stop line of an intersection l can be formulated as follows:

$$u_{EV} * (P_{max,l} \pm w_l), \text{ where, } w_l >= 0.$$

In this study, two types of activation strategy were evaluated using microscopic simulation: single and variable point activation. Figure 4.9 shows the location of those two potential activation points on an emergency route. The single point method activates the preemption sequence when an emergency vehicle arrives at a pre-specified point, either A or B, while in the case of the variable point activation, the activation is determined depending on the state of the signal phase at the intersection when an emergency vehicle arrives at point A as follows:

```
when an emergency vehicle arrives at the first potential activation point A for the intersection l,
 if the signal phase of intersection l is Green for the travel direction of the emergency vehicle,
   then Hold the current phase,
   else if the signal phase is in the "Walk" stage for the cross street traffic,
     then Start Preemption sequence,
     else, Activate Preemption when the emergency vehicle passes the 2nd activation point B.
```

The above procedure is sequentially applied to the intersections on the emergency route in the traveling direction of an emergency vehicle, and, depending on the distance between intersections, it is possible for more than two intersections on a given emergency route to be preempted at the same time. The variable point method tries to minimize the unnecessary preemption while providing green signals for an emergency vehicle with sufficient lead-time, so that the traffic at each intersection can be cleared enough for the safe and efficient passage of the emergency vehicle. In this research, both single and variable point activation methods were simulated and their performance was evaluated with different types of emergency routes.

4.4 Signal Recovery Procedure

As soon as an emergency vehicle clears an intersection, the signal recovery procedure kicks in to restore the original cycle and/or coordinated timing settings of the intersection. The current signal operational policy in Minneapolis, Minnesota, allows only the "Walk" interval in each signal phase to be adjustable during the transition period, while other intervals such as "Flash Do Not Walk" and "Yellow/All Red" must be fixed for safety reasons. Therefore, the signal recovery procedure developed in this study restores the original timing settings by adjusting the amount of "Walk" interval of the cross street, i.e., blocked roadway during preemption.

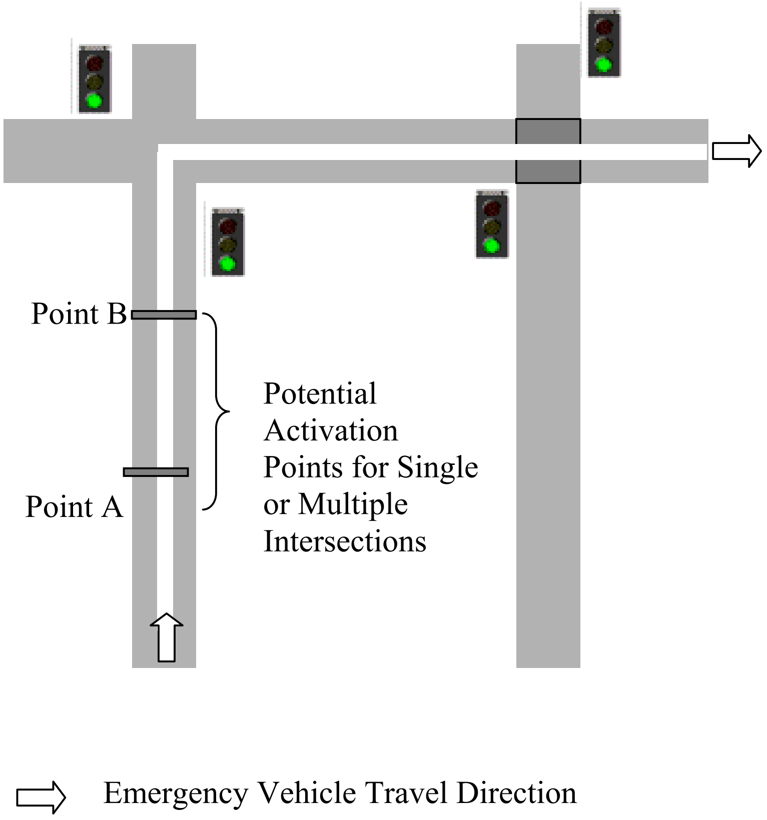

Point B
Point A
Potential Activation Points for Single or Multiple Intersections
Emergency Vehicle Travel Direction

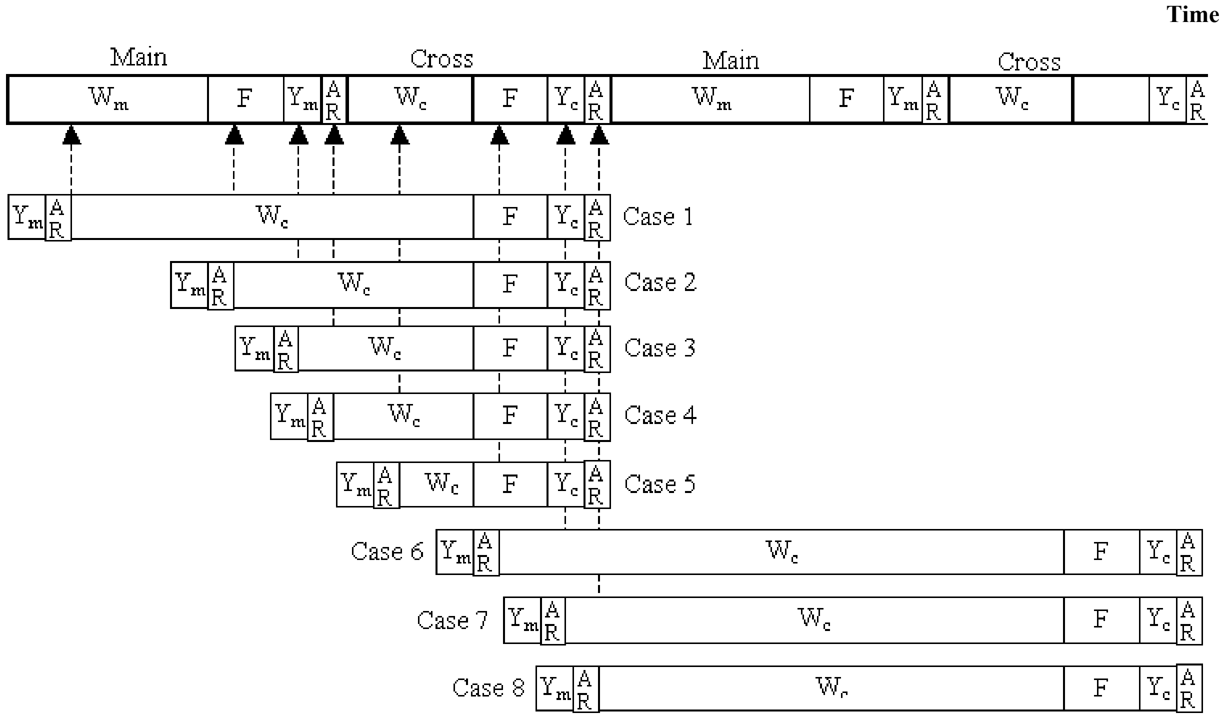
Figure 4.9 Example Activation Points for Dynamic Preemption

Figure 4.10 illustrates the adjustment process for the “Walk” time of cross street to recover the original signal settings of a pre-timed intersection. When preemption is activated for an intersection, the proposed method keeps tracking the original timing plan of the intersection in the background mode on a global time scale. As the preemption terminates, the signal phase starts to change to provide Green time for the cross street, i.e., the blocked roadway during preemption, by changing the Green light of the main street to Yellow and then All Red. Depending on the location where the All Red interval of the main street ends on the original timing plan being tracked on a global time scale in the background mode, the proposed procedure either extends or shortens the Walk time of the cross street, so that the resulting timing settings can catch up to the original timing plan at the end of the current signal phase for the cross street. For example, in the case 2 shown in Figure 4.10, the All Red interval of the main street ends at the Flash Do Not Walk interval of the main street on the original timing plan. In this situation, the Walk interval of the cross street is extended to the originally scheduled Walk time for the next phase, so that the regular timing schedule can be restored at the end of the next phase. While the procedure shown in Figure 4.10 indicates a direct recovery approach within one cycle for a pre-timed control intersection, the proposed method can be extended to a multiple-cycle transition period for incremental adjustment of timing settings.

4.5 Evaluation of Route-based Dynamic Preemption Strategies

The proposed route-based dynamic preemption strategy is evaluated with a microscopic simulation software, VISSIM, whose main simulation function can interact with an external module that can set the state of each signal light in a given network every 1/10th second. One key feature of the simulation software used in this research is its capability to install a set of detectors that can only detect pre-designated emergency vehicles that are generated at pre-specified times during simulation. However, the current version only allows the simulation of emergency vehicles following pre-specified routes, i.e., an emergency route cannot be either generated or changed during simulation. Because of this limitation, the on-line route-selection module developed in this study could not be linked to the simulation software in this work. Therefore, the evaluation conducted in this research focused on the effectiveness of the proposed dynamic preemption strategy over the existing intersection-by-intersection preemption method for a given set of pre-determined routes.

Original Signal Timing Schedule



W_m : Main Street Walk Interval
W_c : Cross Street Walk Interval
F : Flash Do Not Walk
Y_m : Main Street Yellow
Y_c : Cross Street Yellow
AR : All Red

Figure 4.10 Signal Recovery Process

Figure 4.11 shows the example network with four different types of routes, i.e., Route 1-4, where the proposed strategy was simulated and its performance was compared with that of the existing intersection-by-intersection preemption method. Among the routes tried in this study, Route 1 and 3 are relatively simple straight routes with few turns, while Route 2 is the longest route with the most traffic signals. Route 4 has the most left-turns in a relatively short distance.

For evaluating the dynamic preemption with single-point activation, three different activation points upstream of each intersection, i.e., 330m, 400m and 470m from the intersection stop line, were simulated with different random seeds. Those three points were located within 15-20 second range of an emergency vehicle traveling at 80 km/hr. It can be noted that the maximum amount of time needed for preemption at the intersections in the sample network is 20 seconds. The variable point preemption method used two potential activation points, i.e., 15 and 25 second driving distance at 80 km/hr from each intersection, i.e., 330m and 560m from the stop line. Each case was simulated for a 45-minute period with the same peak-hour demand for the sample network, where the emergency vehicle was generated at 15 minutes into the simulation. Once generated, the emergency vehicle traveled the pre-determined route and the signals on the route were dynamically preempted following pre-specified preemption strategy for each case. For a fair comparison, a common set of 6 different random seeds was used for the simulation of each preemption strategy and their results were averaged. Table 4.1 includes the simulation results from each preemption strategy in terms of the network-wide traffic performance including total vehicle-hours and average delay per vehicle after preemption started until the end of the simulation period. Figure 4.12 shows the simulated travel time of an emergency vehicle on each route with different types of preemption strategies evaluated in this study.

As indicated in Figure 4.12 the simulation results clearly show the advantage of the proposed route-based dynamic preemption strategy over the existing method in terms of the emergency vehicle travel time. As indicated in Figure 4.12, all four routes evaluated in this study exhibit consistently reduced travel times of an emergency vehicle with the dynamic preemption strategies compared with those of the intersection-by-intersection preemption. Further, the relatively long and complicated routes, Route 2 and 4, showed significantly larger reduction of the emergency vehicle travel time than the simpler routes, i.e., Route 1 and 3, over the existing intersection-by-intersection clearance method. Route 2, the longest with the

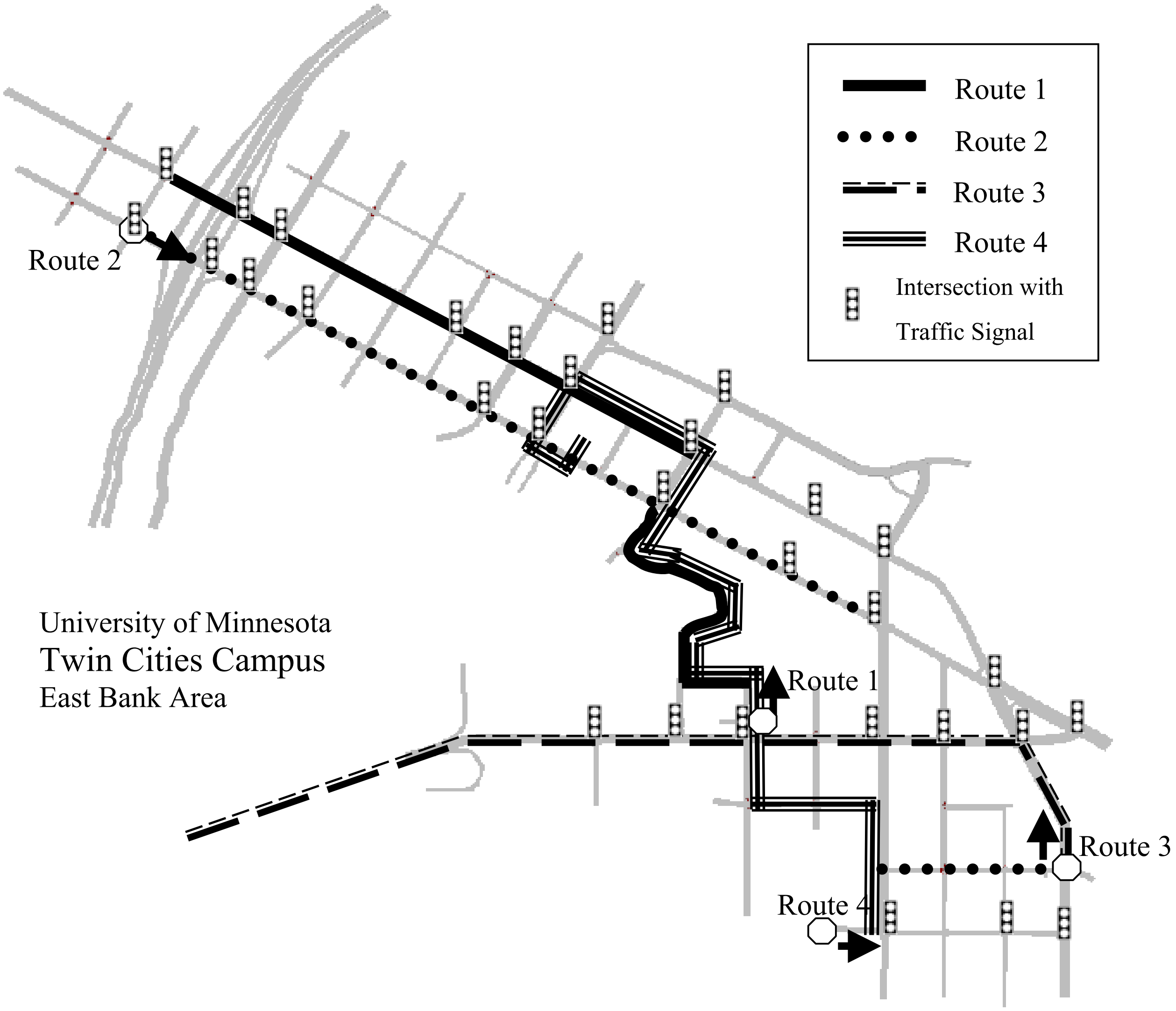


Figure 4.11 Sample Network and Example Emergency Routes

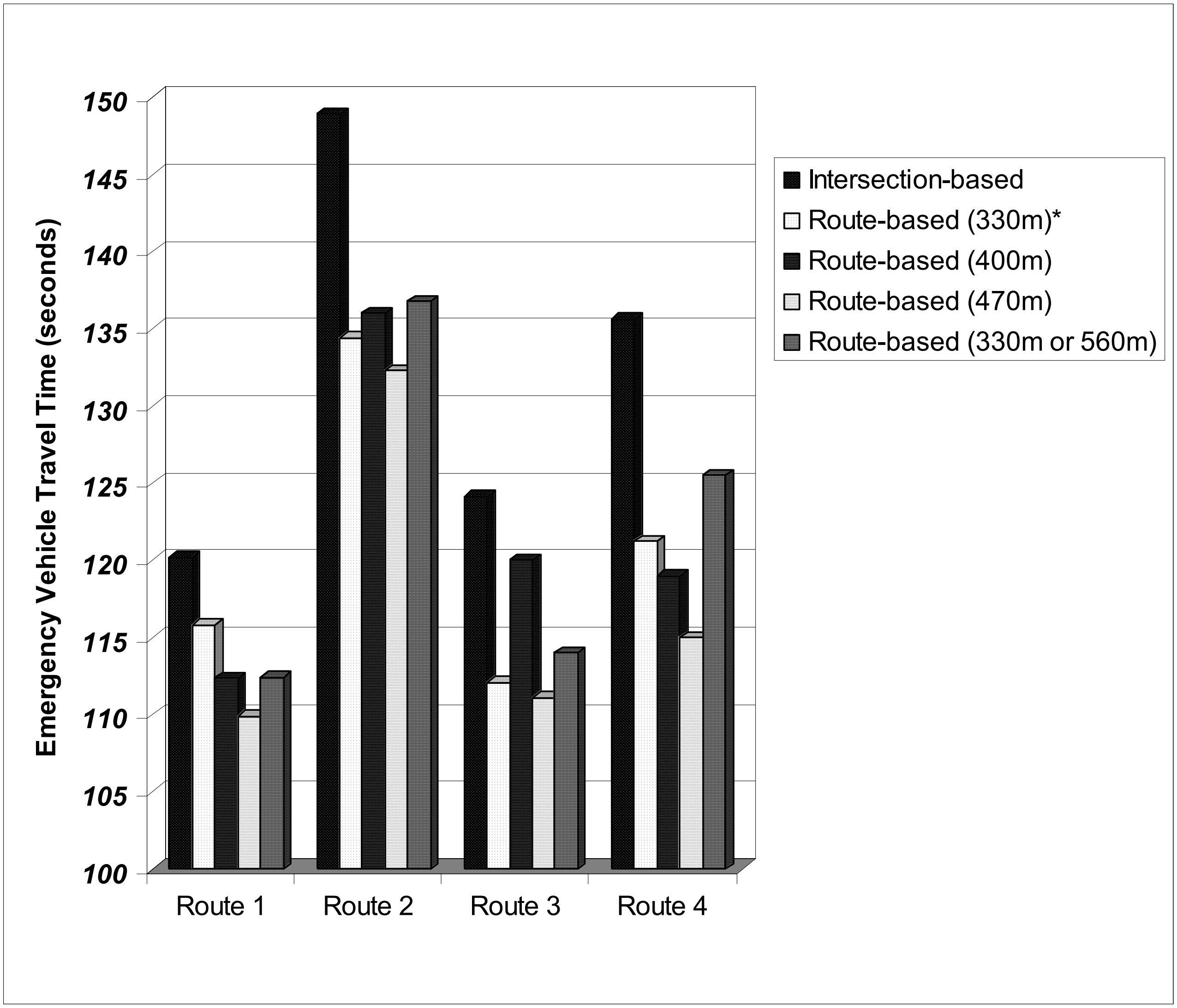


Figure 4.12 Simulated Travel Time of Emergency Vehicle for Different Routes

Table 4.1 Simulation Results from Different Signal Preemption Strategies

		Intersection -based Preemption	Route-based Preemption			
			SPA* (330 m)	SPA (406m)	SPA (470m)	Variable** Point Activation
Route 1	EV travel Time (sec)	120.2	115.8	112.4	109.9	112.4
	Total Vehicle Hours	13.5	12.9	12.7	12.8	13.5
	Average Delay (sec)	17.4	16.1	15.4	16.3	17.5
Route 2	EV travel Time	148.9	134.4	136.0	132.3	136.8
	Total Vehicle Hours	14.2	14.0	13.7	13.6	13.8
	Average Delay	16.1	17.6	16.8	16.7	16.8
Route 3	EV travel Time	124.1	112.1	120.0	111.1	114.0
	Total Vehicle Hours	14.0	13.2	13.1	13.1	13.9
	Average Delay	17.4	16.7	16.9	16.3	17.7
Route 4	EV travel Time	135.6	121.2	119.0	115.0	125.5
	Total Vehicle Hours	10.0	10.2	10.4	10.5	10.5
	Average Delay	15.0	16.1	15.5	15.8	15.7

* SPA: Single Point Activation, e.g., 330 m from the intersection stop line.
** Activate either 330 m or 560 m from the stop line depending on signal status.

most traffic signals, shows 9-11% improvement, while the results with Route 4, the most complicated route, indicate 10-16% reduction of travel time over the existing method. While the results with Route 1 and 3, which are relatively short and straight routes, do not show significant reduction of the emergency vehicle travel time, i.e., 3-11% reduction, they still exhibit consistently lower travel time patterns with the proposed dynamic preemption methods. It can also be noted that, in the case of Route 2, the location of preemption activation points did not make significant differences in terms of the travel time for an emergency vehicle among different dynamic preemption methods. This can be partially due to the current limitation of the simulation software in modeling the behavior of the vehicles reacting to an emergency situation, i.e., an emergency vehicle needs to maneuver among normal vehicles to pass normal vehicles. It can be observed that, with the single-point activation method, while no clear pattern can be found between the location of the activation point and the emergency vehicle travel time, the cases with the furthest activation point, i.e., 470m, consistently produced the most efficient signal preemption in terms of the emergency vehicle travel time. It is also interesting to note that the variable-point activation strategy does not show significant advantage over the single-point activation method in terms of reducing the emergency vehicle travel time.

Table 4.1 includes the network-wide traffic performance after an emergency vehicle entered the network until the end of the simulation period. As shown in this table, the total vehicle-hours and average delay per vehicle show an interesting pattern, i.e., the results with Route 1 and 3, the simpler and shorter routes than Routes 2 and 4, indicate slightly, but consistently improved network-wide performance with the dynamic preemption, while the cases with the longer and more complicated routes, Route 2 and 4, exhibit a clear pattern of degraded performance in terms of average delay. However, the magnitude of the degradation is relatively small ranging from 5-9% compared with the reduced travel time of emergency vehicles. This indicates the efficiency of the proposed strategy by reducing unnecessary preemption in a given network, thus minimizing the delay because of preemption.

5. Development of a Framework for Preliminary Field Testing

5.1 Overview of Operator-based Route Preemption Strategy

In this section, a preliminary field testing of the dynamic signal preemption strategy was conducted with a route selected by an emergency vehicle (EV) operator in real time. Figure 5.1 shows the simplified structure of the field testing system, where an EV unit is connected to multiple intersection units through a wireless communication network. First, an EV operator identifies an emergency route by selecting a set of intersections using the EV unit, which also continuously receives its location data through an in-vehicle GPS device. As the EV travels following the pre-defined emergency route, the software in the EV unit analyzes its location data and determines if one or multiple intersections need to be preempted at the current location of the EV. The identification (ID) numbers of the intersections to be preempted are then broadcasted by the EV unit and all the intersections within the communication range of the EV unit would receive the same information. Upon receiving the information, each intersection unit compares its own ID with the broadcasted numbers and decides if it should start the preemption sequence. When the EV unit passes a preempted intersection, it also sends out a signal to close preemption.

5.2 Development of a Field Testing System

In this study, a small-scale virtual intersection network was developed with off-the-shelf communication products to test the feasibility of the above operator-based wireless preemption methodology. Figure 5.1 also shows the wireless communication devices used for this testing. To receive the GPS data from a GPS device, we used LabIML, RS232 serial instrumentation software made by Windmill Software Ltd. LabIML is a universal driver for instruments that send or accept ASCII messages over RS232. It automatically passes data from the instruments to Windows application software.

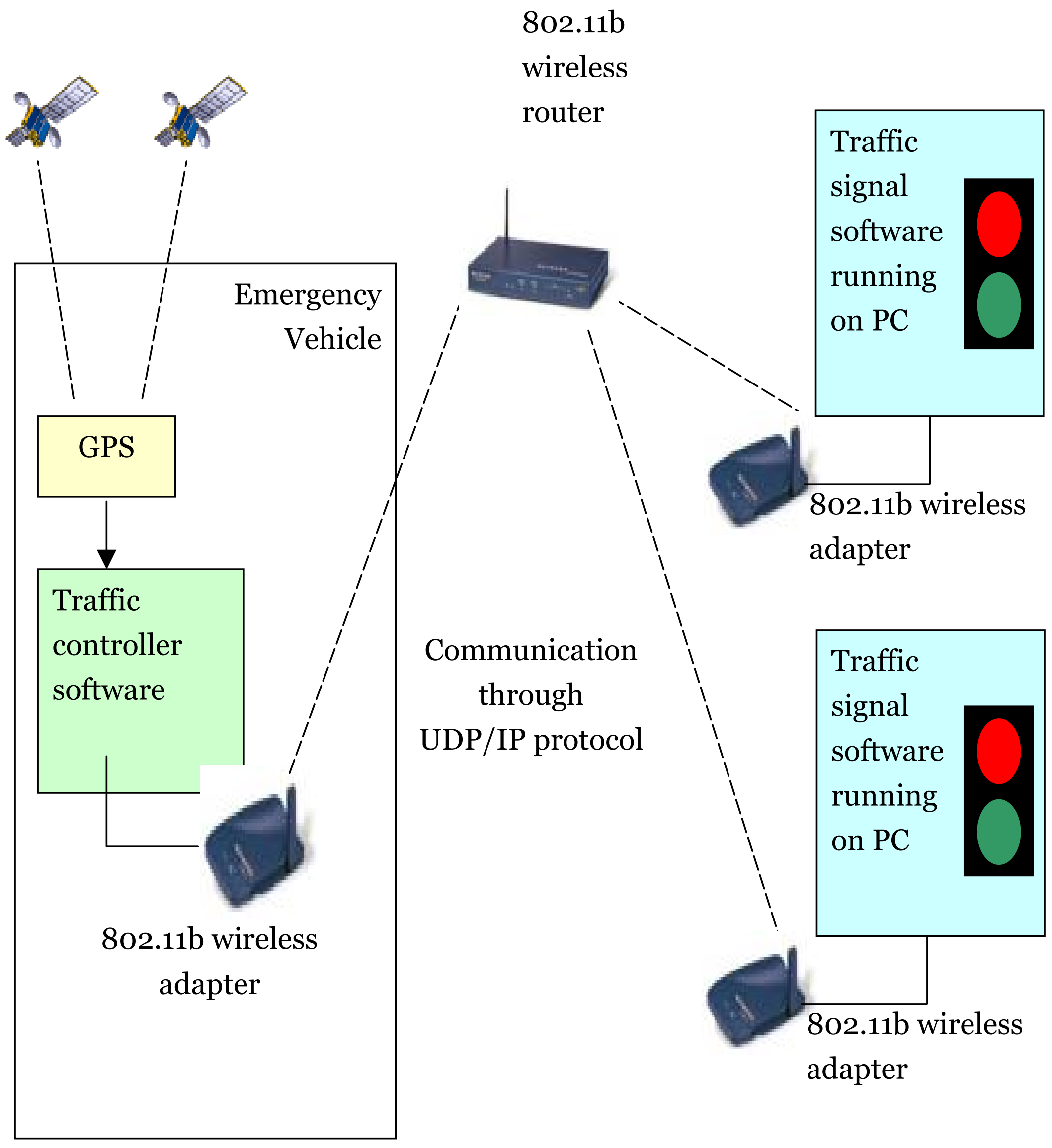


Figure 5.1 Structure of Field Testing Network

LabIML is a DDE server application sending GPS data. The intersection unit emulating a traffic controller is a DDE client application receiving GPS data. This receiving part is built with Delphi’s existing components: TDdeClientConv and TDDEClientItem. The EV unit software analyzes the GPS data and extracts the EV location in terms of its latitude and longitude. Once the location information of the EV and the intersections in a surrounding network are given, the EV unit software determines if the distance between the EV and a certain intersection along the pre-defined emergency route is within the pre-defined distance for activating preemption. The testing network consists of the following devices and software:

EV unit: PC running on Windows operating system, Windmill software,
GPS communicating with PC through COM port,
802.11b wireless adapter
Intersection unit: PC running on Windows operating system,
802.11b wireless adapter
Software Development: Delphi7 (Indy components for UDP network),
Windmill software

To extract information from the GPS's string of data, the Windmill software is configured as follows:

```
$GPGLL,5330.12,N,00215.31,W,134531,A<CR><LF>
```

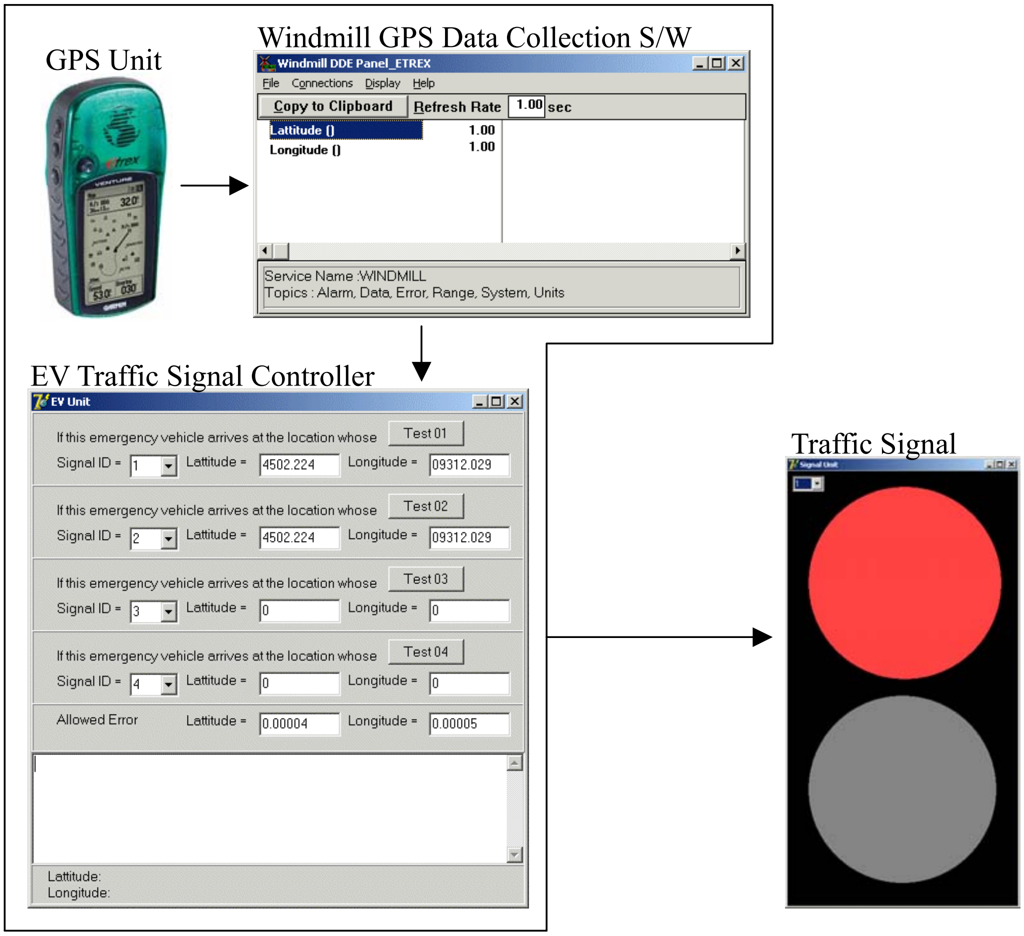
The above data string consists of a NMEA code ($GPGLL), the latitude, North or South, Longitude, East or West, Time (hhmmss), Data Valid (A), Carriage Return <CR> and Line Feed <LF>. Out of this data string, it might be helpful to record just the latitude and longitude. Windmill will collect this information as 2 "channels" of data, but it must be told how to recognize the desired information. In this example, for the longitude channel, you might tell Windmill to search for 'GLL,' and extract up to the next comma. For the latitude channel, search for 'N,' and extract until ',W'. The result will be a parse string looking like this for Channel 0 (latitude): \S"GLL,"\E"," and this for Channel 1 (longitude): \S"N,"\E",W". In this way, a piece of necessary data can be extracted for the desired purpose. Further, to implement wireless communication features of the EVP system, UDP/IP communication protocol was used. UDP/IP protocol offers minimal datagram delivery service, as if we sent letters through the post office. Even though TCP/IP protocol provides a connection-oriented, reliable byte stream service, it causes considerable overhead for connection. In the described test, UDP/IP is preferable in that a fast connection is an important factor for real time applications. Figure 5.2 shows the input data screens of the utility programs developed for this testing.

5.3 Initial Field Testing Results

The above testing network was implemented with three laptop PCs connected with the 802.11b wireless router and adapters. The EV unit software was installed onto one laptop which was used as the moving EV unit, while the other two laptops were used as intersection units. The preliminary testing with the three PC-network found these results:

- The concept of wireless communication-based signal preemption for multiple intersections was feasible within a restricted area for the test network. However, to cover a large area, more enhanced wireless communication devices would be needed.
- The wireless communication becomes unstable when the EV unit is blocked by objects such as a building or garage.
- While the wireless UDP/IP protocol response time is acceptable, the accuracy of the GPS data used in this testing, i.e., 50 feet, and 1 second updating frequency was not enough for emergency vehicle operations. More reliable GPS with frequent

refresh rate would be needed for future testing.



Refresh Rate 1.00 sec

Lattitude () 1.00
Longitude () 1.00

Service Name :WINDMILL
Topics : Alarm, Data, Error, Range, System, Units

If this emergency vehicle arrives at the location whose
Signal ID = 1 Lattitude = 4502.224 Longitude = 09312.029

If this emergency vehicle arrives at the location whose
Signal ID = 2 Lattitude = 4502.224 Longitude = 09312.029

If this emergency vehicle arrives at the location whose
Signal ID = 3 Lattitude = 0 Longitude = 0

If this emergency vehicle arrives at the location whose
Signal ID = 4 Lattitude = 0 Longitude = 0

Allowed Error Lattitude = 0.00004 Longitude = 0.00005

Lattitude:
Longitude:

* GPS unit produced by Garmin eTrex Venture

Figure 5.2 Input Data Screens for Preliminary Field Testing

6. Conclusions

This report presented a route-based dynamic preemption method for emergency vehicles and its evaluation results in an example network using a microscopic simulation model. The proposed strategy sequentially preempts the traffic signals at the intersections along the optimal route with advance activation, so that the traffic queue at each intersection can be cleared for the approaching emergency vehicle. Due to the limitations of the simulation software, the on-line route-selection method developed in this study could not be tested in the current phase. The evaluation results with pre-specified emergency routes show substantial reduction of the emergency vehicle travel time for relatively long and/or complicated routes compared with the existing intersection-by-intersection preemption method, while the magnitude of the benefit can vary significantly depending on the length and type of the route. Further, the network-wide performance measures with the proposed dynamic preemption method were very compatible with those from the existing intersection-by-intersection clearance method. The performance comparison between single and variable-point activation indicates no significant advantage with the more complicated variable point method, which implies the practical applicability of the proposed simple activation strategy.

Future research includes the development of an efficient clearance method for the user-specified route with a decentralized approach. The preliminary field testing network developed in this study showed the possibility of developing a wireless-network-based preemption system for a predefined route. Continuation of field testing with an enhanced GPS and communication devices needs to be conducted. Simulation models also need to be improved to realistically reflect the behavior of vehicles responding to an emergency vehicle.

REFERENCES

1. Bullock, D., Morales, J., and Sanderson, B., “Evaluation of emergency vehicle signal preemption on the Route 7 Virginia corridor,” FHWA-RD-99-70, Federal Highway Administration, Virginia, 1999.
2. Bullock, D. and Nelson, E. “Impact Evaluation of Emergency Vehicle Preemption on Signalized Corridor Operation,” Presented at 2000 TRB Annual Meeting, Transportation Research Board, Washington, D.C., January 2000.
3. Hunter-Zaworski, K. and Danaher, A. “NE Multnomah street Opticom bus signal priority pilot study,” Final report, TNW97-03, Seattle, Washington, 1995.
4. Traffic Technologies, LLC, “Sonem 2000 Digital Siren Detector,” Technical Document for Installation, Traffic Technologies, LLC, 2001.
5. Webber, T. “Priority 1 GPS-based Fire Preemption System,” Internal memo, City of Peoria, Illinois, September 2001.
6. Benerjee, F. T. “Transit Priority System for Metro Rapid Bus,” Evaluation Report, Department of Transportation, City of Los Angeles, June 2001.
7. Shibuya, S., Yoshida, T., Yamashiro, Z., and Miyawaki, M., “Fast emergency vehicle preemption systems (FAST),” ITS America, 79th Annual Meeting of Transportation Research Board, Washington, D.C., 2000.
8. www.fhwa-tsis.com
9. Owen, L., Zhang, Y., Rao L. and Michael, G. “Traffic Flow Simulation using CORSIM,” Proceedings of the 2000 Winter Simulation Conference, 2000.
10. www.paramics-online.com
11. Fellendorf, M. and Vortisch, P. “Validation of the Microscopic Traffic Flow Model VISSIM in Different Real-World Situations”
12. www.itc-world.com
13. Barcelo, J. “Microscopic Traffic Simulation: A Tool for the Analysis and Assessment of ITS Systems,” Transport Simulation Systems.
14. www.aimsun.com
15. Planung Transport Verkehr AG, “Vissim User Manual, V3.61,” Germany, 2001.
16. Ghaman, R., “Advanced Transportation Controller,” web document: stargate.ornl.gov/trb/signal_systems/ATCI
17. Joint Committee on ATC, “ATC Standard Specification for the Type 2070 controller,” 2001.
18. Joint Committee on ATC, “Advanced Transportation Controller Standards Overview,” 2000.
19. Joint Committee on ATC, “Application Programming Interface Standard for the Advanced Transportation Controller”, 1999.
20. Dijkstra, E., “A Note on Two Problems in Connexion with Graphs”, Numerische Mathematik, pp. 269-271, 1959.
21. Cormen, T., Leiserson, C., Rivest, R. and Stein, C. “Introduction to Algorithms, 2nd Edition”, MIT Press, Cambridge, Massachusetts, 2001.
22. Kwon, E., and Stephanedes, Y., “Development of an adaptive control strategy in a live intersection laboratory”, Transportation Research Record 1634, National Research Council, pp. 123-129, Washington, D.C., 1998.
23. Kwon, E., Kim, S. and Kwon, T. “Pseudo real-time evaluation of adaptive traffic control strategies using Hardware-in-Loop simulation,” Proceedings. IEEE Industrial Electronics Society 27th Annual Conference, IECON 2001, Denver, Colorado, November 2001.