Evaluation of Coordinated and Local Ramp Metering Algorithms using Microscopic Traffic Simulation

by

Joseph R. Scariza

B.S. in Civil Engineering, University of Rhode Island (2001)

Submitted to the Department of Civil and Environmental Engineering
in partial fulfillment of the requirements for the degree of

Master of Science in Transportation

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2003

Signature of Author ....
Department of Civil and Environmental Engineering
May 9, 2003

Certified by ....
Moshe E. Ben-Akiva
Edmund K. Turner Professor of Civil and Environmental Engineering
Thesis Supervisor

Certified by ....
Tomer Toledo
Research Associate
Thesis Supervisor

Accepted by ....
Oral Buyukozturk
Chairman, Departmental Committee on Graduate Studies

Evaluation of Coordinated and Local Ramp Metering Algorithms using Microscopic Traffic Simulation

by

Joseph R. Scariza

Submitted to the Department of Civil and Environmental Engineering
on May 9, 2003 in partial fulfillment of the requirements
for the degree of Master of Science in Transportation

Abstract

Ramp meters are special traffic signals at the end of a freeway on-ramp that regulate the flow of traffic onto the mainline. The main purpose of ramp meters is to keep the mainline of the freeway from becoming overly congested, and to maximize the efficient use of freeway capacity. The first use of ramp metering was in Chicago, in 1963, and today ramp meters are becoming more popular in both the US and in Europe.

Although the original ramp metering controllers used pre-timed ramp meters, nearly all modern ramp metering algorithms are traffic responsive. Traffic responsive ramp meters can be divided into two categories: local or coordinated. Local ramp metering algorithms only take into account traffic conditions near a single ramp, while coordinated algorithms try to optimize traffic over an area. Four algorithms are evaluated in this thesis. ALINEA is a local ramp metering algorithm. ALINEA / Q is a local algorithm based on ALINEA, but handles ramp queues in a more efficient manner. FLOW is a coordinated algorithm that tries to keep the traffic at a predefined bottleneck below capacity. The Linked Algorithm is a coordinated algorithm that seeks to optimize a linear-quadratic objective function.

Each of these four algorithms was tested on the M27 Motorway near Southampton, UK. Because none of the algorithms showed any significant benefits, different scenarios were tested, both on the M27 network, and on a generic network. The effect of four variables was studied: total demand, ramp spacing, proportion of traffic using ramps, and traffic distribution among ramps. A regression analysis was performed on each algorithm to determine the sensitivity to each variable. The most significant result was that ramp metering, especially the coordinated algorithms, was only effective when the ramps are spaced closely together. It was also observed that ramp metering was only effective at relatively high demand levels, and that ALINEA / Q and the coordinated algorithms were more effective than regular ALINEA when the volume was extremely high.

Thesis Supervisor: **Moshe E. Ben-Akiva**
Edmund K. Turner Professor of Civil and Environmental Engineering

Thesis Supervisor: **Tomer Toledo**
Research Associate, Center for Transportation Studies

Acknowledgements

I would like to thank my advisors, Prof. Moshe Ben-Akiva, and Dr. Tomer Toledo for all their help and support during my time at MIT. I consider it to be a real privilege to have had the opportunity to work with them during the past two years.

I would also like to thank Lancaster University and the UK Highway Agency for their financial support, which made this research possible. I would especially like to thank Dr. Paul McKenna, of Lancaster University, for his help with the Linked ramp metering algorithm.

I also give thanks to Prof. Harris Koutsopolos, as well as all of the students and researchers in the ITS Lab for their help and for making the ITS place a pleasant place to work. I'd like to give special thanks to Rama Balakrishna and Costas Antoniou for their help with the lab computers and for acting as network administrators. I'd also like to thank Leanne Russell, as well as all the other staff members who have helped me during my time at MIT. I also thank all of the transportation faculty members for all that I learned in my classes, and I felt it was a great honor to have had the opportunity to study under them. I would also like to thank Bob Gordon for all of his help with various ramp metering algorithms.

Last, but not least, I would like to thank my family, especially my parents, Joseph G. and Rosalie and my brother Tom, as well as my friends, especially Tara Pellegrino, for all of their encouragement and emotional support during my time at MIT.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Ramp metering is one of the most frequently used methods of freeway control intended to reduce congestion. Ramp meters are special traffic signals on a freeway on-ramp that allow one vehicle or a platoon of vehicles to enter the freeway. The first use of ramp control was on the Eisenhower Expressway (I-290) in Chicago, Illinois, in 1963, where a police officer directed the traffic to allow one vehicle to enter at a time, at a predetermined rate. Today, ramp metering has evolved and expanded, and is used throughout the US, with notable applications in Minnesota, California, New York, and Washington state. Ramp metering is also becoming popular in Europe, with applications including Amsterdam, Paris, and Glasgow.

1.1 Freeway Congestion

When traffic on surface streets became too congested, freeways were constructed, in an effort to relieve the congestion. Freeways were intended to allow long distance travel between cities at high speed without interference of local traffic. However, in the later half of the 20th century, as Americans became increasingly dependent on automobiles, and as commuters moved to the suburbs, freeways began mixing intercity travel with commuter traffic. The freeways then became very congested, and the problems that existed on the surface streets not exist on the freeways too.

In order to understand what causes freeway congestion, it is important to understand the theory of traffic flow. The following traffic flow parameters are defined:

- Flow (q) = Number of vehicles passing a certain point during a given time period. Given as vehicles per hour (veh / hr)
- Speed (s) = The rate at which vehicles travel (mph)
- Density (k) = Number of vehicles occupying a certain space. Given as veh / mi. k = q / s

The fundamental diagram relates flow and density. Although the exact shape of the curve can vary based on the situation, the typical form of the fundamental diagram is shown in figure 1.1:

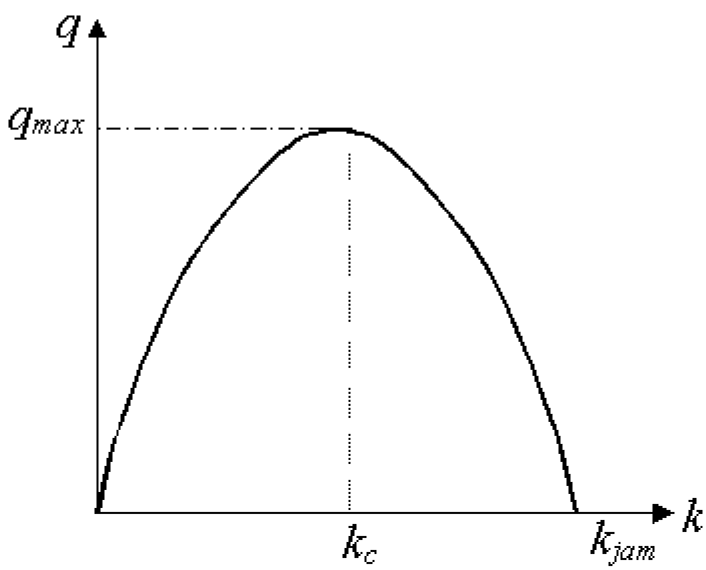


Figure 1.1: Fundamental Diagram of Traffic Flow

When the density reaches a certain point, the critical density (k_c), the freeway reaches its maximum flow (q_{max}). When the density increases beyond the critical density, the flow actually decreases, until the density reaches the jam density (k_{jam}), where the flow becomes zero, and all traffic is stopped. When the density is below the critical density, the flow is said to be stable, or uncongested. When the density exceeds the critical density, the flow is said to be congested, or unstable, and the freeway capacity decreases. Because more vehicles are processed when the flow is stable, it is best for the density to be as close as possible but below to the critical value so the freeway can operate at its full capacity. Ramp metering regulates the flow so that the density remains below the critical value, and the full capacity can be used.

1.2 Application of Ramp Metering

In the past, when roads became congested, either the existing roads were widened, or new roads were built to handle the demand. Unfortunately, road building is expensive, usually requires expensive and disruptive property taking and construction, and can cause negative environmental impacts. Also, the new roads often become just as congested as the old road. Because of these reasons, highway building and improvements are often opposed, and can be politically, economically, and environmentally unfeasible. Fortunately, there are other methods available to manage congestion.

Intelligent transportation systems, or ITS, can be defined as “A joint public-private venture involving the use of integrated communications and electronic technologies in mitigating surface transportation related problems” (Hunter, 2000). ITS involves managing existing facilities more effectively, rather than building new facilities. Ramp metering is an example of ITS.

1.2.1 Ramp Metering Advantages

Ramp metering has many potential advantages (Roess et al, 1998):

- Improvement of freeway mainline flow, due to access control and traffic diverting to other, less congested roads (such as parallel frontage roads)
- Metering smoothes out the traffic flow and breaks up platoons, allowing more efficient merging
- Reduction of accidents, fuel consumption, emissions, and vehicle operating costs
- Network routings may be altered to achieve greater balance and efficiency

1.2.2 Ramp Metering Disadvantages

The main disadvantage to ramp metering is that it can lead to long queues on the ramps, and lead to delays for on-ramp traffic. However, once the vehicles enter the freeway, their speed and travel time should be improved. Another disadvantage is that network rerouting can possibly have negative effects on alternate routes. If no method of controlling ramp queues is used, then ramp traffic can back up onto surface streets. Also, although ramp meters make it difficult to accelerate to high speed on the ramp, this is rarely a problem, since ramp meters should only be used when congestion exists and the highway speed is fairly low.

1.2.3 Types of Ramp Metering Algorithms

The first ramp metering systems used pre-timed controllers. These controllers would choose a metering rate based on the time of day, and would be based on historical data. Because these controllers do not use real-time data, they are unable to respond to changes in traffic patterns. If demand at a particular time is unusually high or low, or if an incident occurs, the metering rate may be ineffective.

Because of this limitation of pre-timed ramp meters, virtually all modern ramp metering systems are traffic responsive. Traffic responsive ramp metering algorithms can fit in one of two categories:

- Local or Isolated control: Local ramp metering control takes into account traffic measurements only near a single ramp, and controls only a single ramp. The metering rate at one ramp does not take into account metering rates at other ramps.
- Area-wide or Coordinated control: Area-wide algorithms take traffic measurements and control several ramp meters as a system in order to optimize traffic over an area, rather than just at a single ramp.

Traffic responsive ramp metering controllers require real-time traffic surveillance. A variety of devices are available for traffic surveillance, although magnetic loop detectors are by far the most common. Loop detectors typically measure traffic flow, speed, and occupancy. Occupancy refers to the proportion of the time that a vehicle is located on the sensor, and is used as a measure of traffic density.

1.3 Objective

The purpose of this thesis is to compare various local and area-wide ramp metering strategies, and to determine under what situations ramp metering, and coordination are useful. Four strategies will be evaluated: a local strategy known as ALINEA (Papageorgiou et al, 1991); an enhanced version of ALINEA that more effectively controls ramp queues, known as ALINEA/Q (Smaragdis and Papageorgiou, 2003); a bottleneck based area-wide algorithm known as FLOW (Jacobson et al, 1989), and an area-wide algorithm based on proportional-integral plus, linear quadratic (PIP-LQ) optimal controllers (Taylor et al, 1998). These algorithms were chosen because they are representative of various classes of ramp metering algorithms that are in use today, as discussed in the literature review. This thesis will be concerned with measuring the effects of demand level, traffic patterns, and ramp spacing on both the corridor travel time and the total travel time for all vehicles. Because field testing is not feasible for this project, a microscopic traffic simulator known as MITSIMLab will be used for the study.

1.4 Organization of Thesis

This thesis is organized into six chapters. Chapter 2 is a literature review, describing existing ramp metering algorithms and prior field studies and simulation studies. Chapter 3 describes the MITSIMLab simulation laboratory, as well as the case study being used, and the MITSIMLab enhancements necessary for the project, and a method of calibration and validation. Chapter 4 shows the results and conclusions from the M27 network in the UK, used as the case study for this project. Chapter 5 shows the design and results from study using a generic freeway network. Finally, Chapter 6 presents conclusions, as well as ideas for future work in the area of ramp metering.

Chapter 2

Literature Review

Ramp metering is a very effective method of freeway traffic control, which uses a traffic signal on an on-ramp, to regulate the flow of traffic. Ramp metering was first used in the 1960s. The original ramp meters were pre-timed, and were not sensitive to real-time traffic conditions. These controllers would choose a program based on the time of day, with cycle lengths based entirely on historical traffic volumes (Blosseville, 1985). Today, however, nearly all ramp meter controllers are based on real-time traffic surveillance.

2.1 Local Ramp Metering Algorithms

Ramp metering controllers can be divided into two broad categories: local or isolated, and area-wide or coordinated. Local ramp meters take into effect only the traffic conditions at a single ramp, whereas coordinated ramp meters take into account traffic conditions over an area.

2.1.2 Demand-Capacity

One traffic responsive local ramp metering algorithm is known as demand-capacity (Masher et al., 1975, Koble et al., 1980). This works by measuring the downstream occupancy (O_{out}), and if it is above the critical occupancy (O_{cr}), congestion is assumed to exist, and the metering rate is set to the minimum rate (r_{min}). Otherwise, the volume is measured upstream of the merge (q_{in}), and the metering rate is set to the difference between the downstream capacity (q_{cap}) and the upstream volume. The equation used is:

$$r = \begin{Bmatrix} Max(q_{cap} - q_{in}, r_{\min}) \;\; if \; O_{out} \le O_{cr} \\ r_{\min} \quad otherwise \end{Bmatrix} \tag{2.1}$$

2.1.3 Percent-Occupancy

A second example of a local ramp metering algorithm is the percent-occupancy strategy (Masher et al., 1975, Koble et al., 1980). The advantage to this strategy is that it does not require calculation of freeway capacity, and thus has potentially lower implementation cost. This strategy uses only upstream sensor occupancy measurements, and uses the occupancy as the sole means to identify and measure congestion. The critical occupancy is measured using historical data (Hadj Salem et al., 1988). This algorithm involves 2 constants: K_1 is the capacity flow, and K_2 is a constant based on slope of a straight line approximation of the uncongested part of the fundamental diagram. The metering rate is determined by:

$$r(k) = K_1 - K_2 o_{in}(k-1) \tag{2.2}$$

2.1.4 ALINEA

Both demand-capacity and percent-occupancy are examples of open loop, or feedforward control. These strategies do not use the system output as input for the next iteration. In contrast, closed loop, or feedback control, the control input is based on the system output. Generally, closed loop systems are more robust than open loop systems.

The most widely used closed-loop local ramp metering algorithm is known as ALINEA (Papageorgiou et al, 1991). ALINEA is based on the linear quadratic (LQ) feedback law. A fairly simple equation is used to calculate the metering rate:

$$r(k) = r(k-1) + K_R[O - O_{out}(k-1)] \tag{2.3}$$

Note that the metering rate, r(k) is a function of the metering rate for the previous iteration, r(k-1), therefore ALINEA is a closed loop algorithm. The metering rate is also a function of the difference between the measured occupancy (O_{out}) and a target set point occupancy (O). K_R is a regulator parameter. ALINEA is one of the most commonly used and one of the most effective algorithms, and will be described in greater detail in Section 2.5.1.

2.1.5 Recent work on local closed loop strategies

Recently, there has been research to enhance the ALINEA algorithm. Oh and Sisiopiku (2001) proposed a modified version, known as MALINEA. MALINEA addresses two main disadvantages to ALINEA. The first is that although ALINEA optimizes the occupancy downstream of the entrance ramp, congestion can still occur upstream of the ramp. The second is that the optimal detector location can be difficult to determine. MALINEA measures the upstream occupancy, O_u, and accepts as parameters a regulator parameter K, the slope of the curve relating the downstream and upstream occupancies (A), and the time lag between the upstream and downstream measurements (n). The equation that MALINEA uses is:

$$Q^r(t+1) = [O_u(t+n+1) - O_u(t)]K / A + Q^r(t-n) \tag{2.4}$$

Smaragdis and Papageorgiou (2003) expanded the applications of ALINEA-based algorithms. The traditional ALINEA algorithm requires occupancy measurements on downstream detectors, which are unfortunately not always available. FL-ALINEA is an algorithm that uses flow measurements from downstream detectors, rather than occupancy measurements. Its formula is identical to the formula used for traditional occupancy-based ALINEA, except that it measures flow, and tries to reach a set point flow rather than a set point occupancy. However, when the occupancy is over the critical occupancy, the metering rate is set to the minimum rate, since the freeway is already over capacity.

$$r(k) = \left\{ \begin{matrix} r(k-1) + K_F[\hat{q} - q_{out}(k-1)] \;\; if \; O_{out} \leq O_{cr} \\ r_{\min} \quad otherwise \end{matrix} \right\} \tag{2.5}$$

UP-ALINEA uses occupancy measurements, but from upstream detectors, and estimates the downstream occupancy. This is useful in cases where a feedfoward algorithm (such as demand-capacity or percent-occupancy) was previously used. This algorithm uses the following equation to estimate the downstream occupancy, $\tilde{O}_{out}$ using the upstream occupancy, and incorporating the effects of the ramp traffic:

$$\tilde{O}_{out}(k) = O_{in}(k)\left[1 + \frac{q_{ramp}(k)}{q_{in}(k)}\right]\frac{l_{in}}{l_{ouy}} \tag{2.6}$$

where ë$_{in}$ is the number of lanes upstream of the ramp, and ë$_{in}$ is the number of lanes downstream of the ramp.

The algorithm is then identical traditional ALINEA in all other ways, using the estimated downstream occupancy from equation 2.6:

$$r(k) = r(k-1) + K_R[\hat{O} - \tilde{O}_{out}(k-1)] \tag{2.7}$$

UF-ALINEA is a variation based on upstream flow measurements. This algorithm simply uses the sum of the upstream flow and the ramp flow to estimate the downstream flow:

$$\tilde{q}_{out}(k) = q_{in} + q_{ramp} \tag{2.8}$$

The equation used for the algorithm is then virtually identical to the equation used for FL-ALINEA, equation 2.5.

$$r(k) = \begin{Bmatrix} r(k-1) + K_F[\hat{q} - \tilde{q}_{out}(k-1)] \ \ if \ \ \tilde{O}_{out} \leq O_{cr} \\ r_{\min} \quad otherwise \end{Bmatrix} \tag{2.9}$$

X-ALINEA/Q is where any of the modified ALINEA algorithms are used with queue control. All of these algorithms, except for X-ALINEA/Q are less efficient than the traditional ALINEA algorithm, but are useful when downstream occupancy measurements are not available. X-ALINEA/Q will be described in greater detail in Section 2.5.3.

2.2 Area-wide Ramp Metering Algorithms

Although the ALINEA local ramp metering algorithm is widely used, the current trend is toward area-wide, or coordinated algorithms. Area-wide algorithms are designed to optimize traffic flow over a section of the freeway rather than just a single ramp, in order to achieve greater efficiency. Area-wide algorithms can be further divided into three classes: incremental or cooperative algorithms; bottleneck or competitive algorithms; and integral algorithms (Kwon et al, 2001; Zhang et al, 2001).

2.2.1 Incremental algorithms

Cooperative, or incremental algorithms work similarly to local algorithms. However, when a ramp is metered very restrictively, the upstream ramps are also metered more restrictively, in order to increase efficiency, and to avoid congesting a single ramp. A cooperative algorithm is used in Denver, Colorado (Lipp et al, 1991).

2.2.2 Bottleneck Algorithms

Bottleneck, or competitive algorithms calculate both a local metering rate and a bottleneck metering rate. The bottleneck metering rate is calculated to keep the flow of traffic at a defined bottleneck below capacity. For each ramp, the more restrictive of the two rates is chosen. A well known and widely used competitive algorithm is known as FLOW, and was formerly used in Seattle, Washington (Jacobson et al, 1989). FLOW will be described in greater detail in Section 2.5.4.

Another example of a competitive algorithm is System Wide Adaptive Ramp Metering, or SWARM, used in Irvine, California (Paesani et al, 1997). Unlike FLOW, SWARM uses predicted volumes, rather than just measured traffic conditions to locate bottlenecks. Empirical results comparing FLOW versus SWARM are inconclusive, and indicate that the performance of SWARM is very sensitive to the accuracy of the predictions (Zhang et al, 2001).

2.2.3 Integral Algorithms

The third class of coordinated ramp metering algorithms is integral algorithms. Integral algorithms have a well-defined objective function to optimize. According to Zhang et al (2001), these algorithms are the most theoretically sound and potentially the most robust, however, they are also the most complex to calibrate and operate.

2.2.3.1 METALINE

Papageorgiou et al (1990) extended the ALINEA algorithm into an integral coordinated algorithm known as METALINE. METALINE was used in Paris, France. The equation for METALINE is basically a vectorization of the ALINEA equation, which uses vectors of occupancy, and 2 control gain matrices to return a vector of metering rates. The equation is basically a vectorization of the ALINEA equation:

$$\mathbf{r}(k) = \mathbf{r}(k-1) - \mathbf{K_1}(\mathbf{O}(k) - \mathbf{O}(k-1)) - \mathbf{K_2}(\mathbf{O}(k) - \mathbf{O_{cr}}) \qquad (2.10)$$

2.2.3.2 Fuzzy Logic Controller (FLC)

Another, more recent example of an integral algorithm is the fuzzy logic algorithm (Taylor and Meldrum, 1998), used today in Seattle, Washington. This algorithm uses qualitative measurements and a method similar to human reasoning to divide measurements into categories. A set of fuzzy rules, with different weighting factors, converts the fuzzified measurements into a metering rate. Empirical tests show that this algorithm performs favorably compared to other algorithms, can prevent congestion before it occurs, and can effectively use imprecise detector data (WSDOT, 2000). According to Zhang et al (2001) this algorithm is theoretically very attractive, but very complicated to configure, and performs very poorly when not configured properly, which limits the practical value of this algorithm in the field.

2.2.3.3 Lancaster University – Linked Algorithm

As part of the Lancaster University coordinated ramp metering project, a linked ramp algorithm, based on nonminimal state space (NMSS) approach to control system design was developed (Taylor et al 1998). This algorithm is based on adaptive proportional-integral-plus, linear quadratic (PIP-LQ) optimal controllers. The objective of this algorithm is to minimize a linear-quadratic cost function. The equation used to determine the metering rates is:

$$\mathbf{u}_t = \mathbf{u}_{t-1} - \mathbf{F}(\mathbf{O}_t - \mathbf{O}_{t-1}) + \mathbf{K}_\mathbf{I}(\mathbf{y}_{\mathbf{d},t} - \mathbf{y}_t) \qquad (2.11)$$

Where:

- $\mathbf{u}_t$, $\mathbf{u}_{t-1}$ = vector of on-ramp flows (metering rate) for current and previous time period

- **F** = feedback matrix
- $\mathbf{O}_t$, $\mathbf{O}_{t-1}$ = vector of occupancy measured at each sensor over entire control area, for current and previous time period
- $\mathbf{K_I}$ = matrix of integral gains
- $\mathbf{y}_{\mathbf{d},t}$ = vector of set point occupancies downstream of each on-ramp
- $\mathbf{y}_t$ = vector of occupancies measured downstream of each on-ramp

This algorithm is described in greater detail in Section 2.5.5.

2.3 Queue Control and Equity Concerns

One potential problem with ramp metering is that, although it can significantly improve corridor travel time, it can cause long queues on entrance ramps, leading to long delays. These delays also may affect local streets when the ramp queues spill on to them.

2.3.1 Queue Adjustment and Queue Override

Many ramp metering algorithms are used in conjunction with either queue adjustment and / or queue override. Queue adjustment increases the metering rate to a less restrictive rate when a ramp queue becomes excessively long. Queue override completely disables ramp metering when a ramp queue becomes too long.

A problem with both queue adjustment and queue override is that most of them are algorithms separate from the main control algorithm, and compete with it. This can lead to oscillation, where the freeway will become congested, causing ramp to be metered very restrictively, leading to a long queue, and activate queue adjustment or override. This will then allow vehicles to flood the freeway, leading to congestion, and causing an even more restrictive metering rate, and the cycle continues. Gordon (1996) proposed an algorithm to take into account both the mainline traffic and the ramp queues in order to avoid having two competing algorithms that lead to oscillation. This algorithm works by calculating a smoothed occupancy for the ramp queue detectors, and raising the metering rate when the smoothed occupancy is above a certain threshold.

2.3.2 BEEX Algorithm

Another side effect of ramp queues is that they can lead to inequity. Often, from a system point of view, the most efficient situation is where a particular ramp is metered very restrictively. This, however, is also the least equitable situation, since the users of one on-ramp will experience long delays while everyone else experiences very little delay. The most equitable situation would be where every ramp, and the mainline, has the same delay. Unfortunately, this is rarely the most efficient strategy.

Although most ramp metering algorithms have focused on reaching optimal efficiency, Zhang and Levinson (2003) developed a series of coordinated algorithms that seek to balance efficiency and equity. These BEEX (Balanced Efficiency and Equity) algorithms seek to minimize the total weighted travel time, which involves weighting both the freeway mainline travel time and the ramp delays. The weighted travel times take into account human perception of delay by using a non-linear function to increase the weight when the delay increases.

2.4 Evaluations of Ramp Metering

2.4.1 Field Testing

Several recent field evaluations of ramp metering centered around the Minneapolis-St. Paul metro area, in Minnesota. Minnesota has one of the largest ramp metering systems in the world. MnDOT started using ramp meters in 1970, and currently uses a coordinated ramp metering algorithm that divides the freeway into zones. Cambridge Systematics (2001) estimates that ramp metering saves the motoring public $40 Million, increasing mainline mean freeway speeds from 46 mph to 53 mph, and significantly reducing accidents.

2.4.1.1 Minnesota Evaluation

Unfortunately, despite the fact that ramp metering can have great benefits, the public often opposes the use of ramp metering. For this reason, MnDOT needed evidence that ramp metering is beneficial. Two highways were selected for evaluation: Trunk Highway 169, a circumferential highway; and I-394, a downtown highway. Traffic was studied in March, 2000. The ramp meters were shown to reduce total travel time between 6% and 16%, with speeds increasing between 13% and 26%. Because of the improved traffic flow, traffic stops on ramps were cut to one third when ramp metering were implemented. Ramp metering was also shown to reduce both fuel consumption and pollutant emissions by between 2% and 47% (Hourdakis and Michaelopoulos, 2002).

2.4.1.2 Minnesota Shut-off Experiment

The Minnesota state legislature passed a bill requiring ramp meters to be shut off for evaluation for eight weeks in the fall of 2000. This experiment showed that shutting off the ramp meters increased congestion and increased accidents, and changed travel patterns. However, on certain test sites, the ramp meters were also shown to significantly increase travel time for short trips, despite improving travel time for longer trips. Because of this, MnDOT will now focus on equity, rather than simply improving mainline efficiency (Levinson et al, 2002).

2.4.1.3 European Evaluations

Ramp metering field evaluations have also occurred in Europe. ALINEA and several other local ramp metering algorithms were tested on the Boulevard Peripherique in Paris, France, as well as the A10 Motorway in Amsterdam, Netherlands. These tests showed ALINEA to be the superior local ramp metering algorithm. ALINEA was also compared to METALINE, a coordinated algorithm. ALINEA and METALINE showed similar results, although METALINE is by far more difficult to set up and calibrate (Papageorgiou et al, 1997).

2.4.2 Simulation Testing

Although field testing can be a useful method of evaluating ramp metering algorithms, it has many limitations. Field testing can be expensive, difficult, and time consuming. The impossibility of changing detector locations in real time can limit flexibility. Also, it is difficult to isolate the effect of ramp metering from other uncontrollable factors, such as weather, incidents, construction, or changes in traffic patterns. For these reasons, traffic simulation has become a valuable tool used as an alternative to field evaluation.

2.4.2.1 Paris, France

A macroscopic traffic simulator known as METANET was used to study the Boulevard Peripherique, in Paris, France. The test was to compare ALINEA and METALINE. The results showed that the two algorithms performed similarly for recurring congestion, although METALINE performed slightly better for non-recurring congestion (Papageorgiou et al, 1991).

2.4.2.2 Twin Cities, Minnesota

Kwon et al (2001) used a macroscopic simulator at the University of Minnesota in order to compare a coordinated algorithm from each of the three classes: the incremental algorithm used in Colorado; the zone algorithm used in Minnesota, and the fuzzy logic algorithm used in Seattle, Washington. Because the Minnesota algorithm did not use queue control, it resulted in the most restrictive metering rates, the lowest amount of mainline congestion, but the longest ramp queues. In contrast, the Denver algorithm and the Seattle fuzzy logic algorithms both showed that queue control can reduce the mainline efficiency. Furthermore, this test showed that the fuzzy logic algorithm is very sensitive to the weights used for each rule.

2.4.2.3 Orange County, California

The PATH program at the University of California (Zhang et al., 2001) used Paramics to compare four algorithms: ALINEA, Bottleneck, Zone, and SWARM. The tests showed that all of the algorithms tested improve traffic flow. Also, there was very little difference in the performance of each algorithm, perhaps due to the difficulty in calibrating the more complex coordinated algorithms. Also, the performance of SWARM was very sensitive to the accuracy of the predictions that it makes.

2.4.2.4 Central Artery / Tunnel, Boston, MA

Hasan (1999) used MITSIMLab to study ramp metering on the Central Artery / Tunnel (Big Dig) network, and compared the local strategy ALINEA with the coordinated strategy FLOW. The results showed that ramp metering deteriorated system performance at low demands, and that coordination was only effective at very high demand levels. However, ramp metering almost always improved the mainline traffic flow. He also showed that queue control always improved system performance, and that coordination significantly improved performance when a bottleneck existed downstream of the on-ramp.

2.5 Ramp Metering Algorithms Studied

Of the ramp metering algorithms described in this chapter, four will be studied in greater detail: ALINEA, ALINEA/Q, FLOW, and the Linked Algorithm. Additionally, the local Ramp Metering Pilot Scheme (RMPS) currently used in Southampton, which is loosely based on ALINEA, will also be discussed.

2.5.1 ALINEA

ALINEA (Papageorgiou et al, 1991) is a local, closed loop ramp metering algorithm. It is a closed loop algorithm because the metering rate is a function of the metering rate that was used in the previous time interval. ALINEA works by measuring the occupancy at a loop detector downstream of the ramp, and measuring the difference between the measured occupancy, and the optimal set point occupancy. The set point occupancy is generally set slightly lower than the critical occupancy, in order to ensure that the freeway is always operating below capacity. The equation used to calculate the metering rate for time interval k is:

$$r(k) = r(k-1) + K_R[O - O_{out}(k-1)] \qquad (2.12)$$

The parameters are as follows:

- r(k), r(k-1): metering rate for the current, and previous interval
- K_R: regulator parameter
- O: set point occupancy
- O_{out}: measured occupancy

Papageorgiou et al (1991) recommend that the regulator parameter, K_R, be set to 70 veh/hr. Experimentation shows that the operation of the algorithm is not very sensitive to the value for K_R, so 70 veh/hr is almost always used.

If the controller measures that the occupancy is lower than the set point occupancy, then the metering rate is increased, and more vehicles are allowed to enter the freeway. If the controller measures that the occupancy is higher than the set point occupancy, then the metering rate is decreased, so fewer vehicles can enter the freeway, so that it can be less congested.

When the calculated metering rate is greater than the maximum rate (1320 veh / hr in this implementation), the ramp meters shut off. When the calculated metering rate is less than the minimum rate (240 veh / hr in this implementation), the metering rate is set to the minimum rate. Also, in order to prevent excessively long queues, queue override is used. When the ramp's queue detector exceeds a set occupancy, the ramp meter shuts off.

2.5.2 ALINEA/Q

ALINEA/Q (Smargdis and Papageorgiou, 2003) is an enhancement to the traditional ALINEA algorithm, using a more sophisticated queue control strategy. This algorithm uses video detectors, rather than loop detectors, to measure the length of the ramp queue. The advantage of video detectors is that they are area detectors, rather than point detectors, and can measure the length of the queue over an area.

This algorithm calculates two metering rates. The first rate is calculated exactly the same as in the traditional ALINEA algorithm. The second rate that is calculated is the minimum rate needed to keep the ramp queue at or below the maximum allowable queue length. This rate is calculated as:

$$r'(k) = -\frac{1}{T}[\hat{w} - w(k)] + d(k-1) \qquad (2.13)$$

where:

- r'(k): minimum metering rate to prevent queue buildup
- $\hat{w}$: maximum allowable queue length
- w(k): number of vehicles in ramp queue
- T: time period over which measurements are taken
- d(k-1): number of vehicles entering ramp

The final calculated rate is the greater of either the ALINEA rate or the queue control rate:

$$R(k) = \max\{r(k), r'(k)\} \quad (2.14)$$

Using this algorithm has several advantages over using a simple on / off queue override method. The metering rate is adjusted more smoothly, and the oscillation of the simple method is avoided. Also, by calculating the queue length at each interval, the algorithm can potentially keep queues from forming by always having a sufficiently high metering rate, rather than waiting for a long queue to develop before taking any action.

2.5.3 Ramp Metering Pilot Scheme (RMPS)

Currently, on the M3 and M27 freeways in the UK, a local ramp metering strategy, loosely based on ALINEA is used (Gould et al, 2002). This algorithm has the following components:

- Calculation of smoothed flows, speeds, and occupancies
- Switch on / off algorithm
- Calculating cycle lengths

- Queue control
- Queue override

This algorithm calculates smoothed flows, speeds, an occupancies. The equation for smoothed flow is:

$$F_S(t) = [1 - K_F]F_S(t-1) + K_F F(t) \qquad (2.15)$$

The formulas for smoothed speed and smoothed occupancy are the same, substituting speed or occupancy for flow.

The switch on / off algorithm uses mainline smoothed flows and speeds downstream of the ramp. During the times of day in which the controller is operation, the ramp meter switches on only when the smoothed flow is above the rising flow, and the smoothed speed is below the falling speed. When the ramp meter is switched up, a start up sequence is invoked. This cycle has a pre-timed green phase (usually 30 seconds), a 3 second stopping amber phase, followed by a red phase. After the red phase, the normal cycle begins. The ramp meter shuts off when the smoothed flow falls below the falling flow, and the smoothed speed rises above the rising speed.

Figure 2.1, below, shows a flowchart showing when the ramp meters switch on or off. Note that TMO refers to the loop detectors:

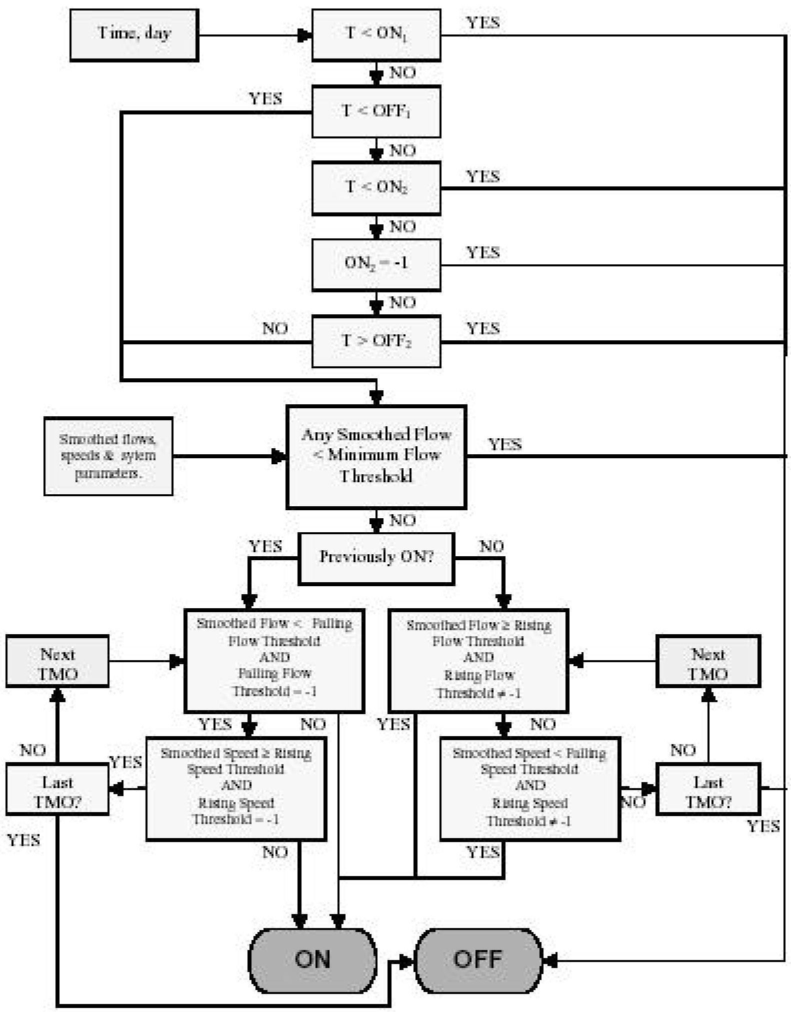


Figure 2.1: RMPS Switch On / Off Algorithm

During the normal operation of the algorithm, a cycle length is calculated once each minute, based on the smoothed occupancy of the downstream mainline loop detectors. A lookup table of up to 5 predetermined cycle lengths is used. Each cycle length has a rising threshold occupancy and a falling threshold occupancy associated with it. When the occupancy rises above the rising threshold occupancy, the next highest cycle length is selected. When the occupancy falls below the falling threshold occupancy, the next lowest cycle length is selected. This lookup table is designed to approximate the

ALINEA algorithm. Figure 2.1, below, shows a flowchart showing how the cycle lengths are selected:

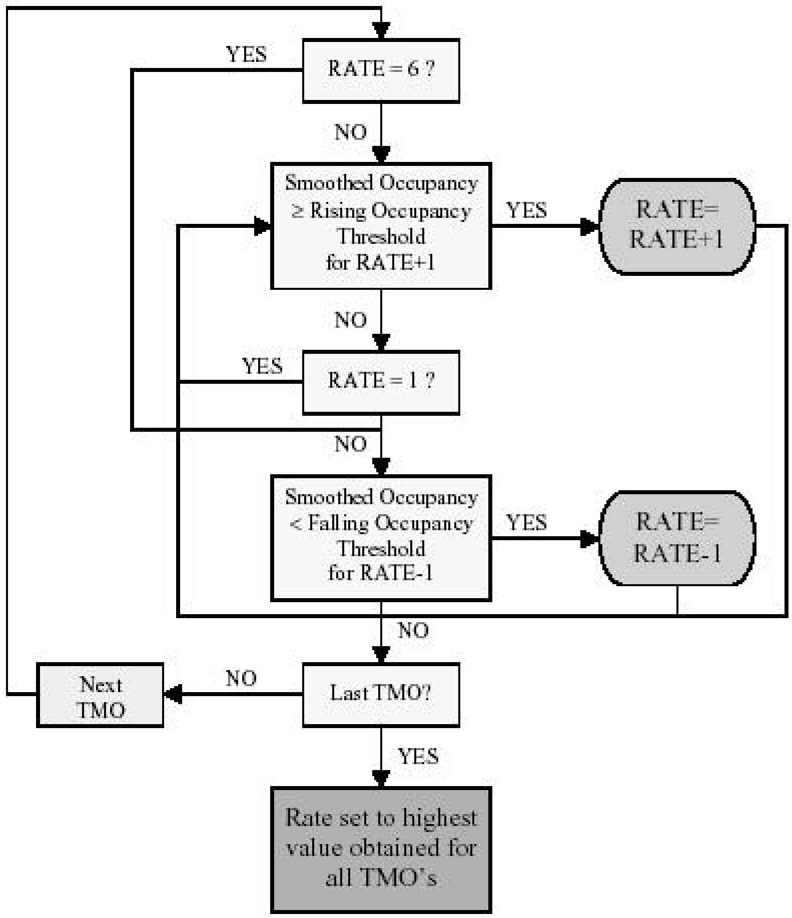


Figure 2.2: RMPS Cycle Length Selection

An English traffic signal cycle is slightly different than an American traffic signal cycle. In England, after the green phase, a 2 second stopping amber phase (equivalent to the American yellow phase) is invoked, followed by the red phase. After the red phase, a 2 second starting amber phase (which has no equivalent in the US) is invoked, followed by the green phase.

The timing of the green phase is determined by the use of a release detector, located slightly beyond the stop line. When a predetermined number of vehicles cross the release detector, the green phase ends and the stopping amber phase begins. The time of the red phase is calculated as whatever time is left of the cycle length that is not taken up by the green, stopping amber, or starting amber phases. Also, there is a pre-set minimum red time. The minimum red time and the cycle length are used to calculate a maximum green time, and the green phase will end once the maximum green time is exhausted, regardless of whether or not enough vehicles crossed the release detector.

This algorithm a queue adjustment algorithm to prevent queues. The occupancy at several queue detectors on the ramps is measured and weighted to calculate an adjustment score. Depending on the adjustment score and the thresholds that are set, the cycle length may be increased by as many as 3 cycle levels from the lookup table. Figure 2.3, below, shows how the queue adjustment algorithm works:

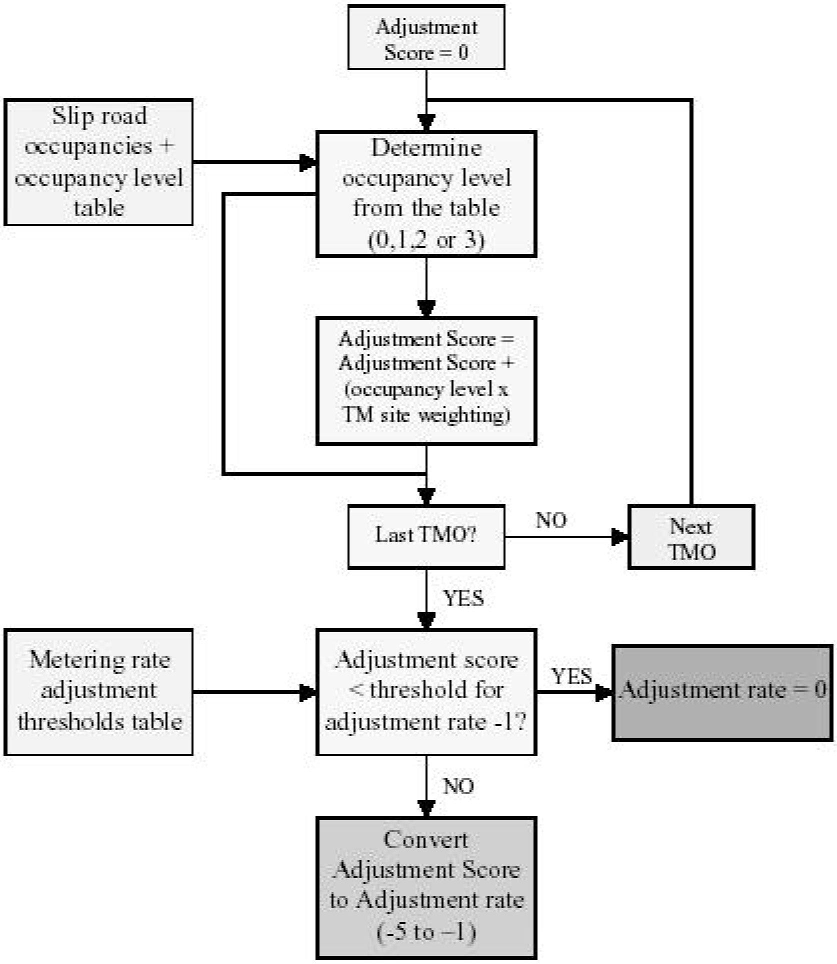


Figure 2.3: RMPS Queue Adjustment Algorithm

2.5.4 FLOW

FLOW (Jacobson et al, 1989) is a competitive, bottleneck-based, area-wide ramp metering algorithm. For each ramp, FLOW calculates both a local metering rate and a bottleneck metering rate, and selects the more restrictive of the two rates.

The local metering rate uses a percent occupancy algorithm. A lookup table is used to relate upstream occupancies with metering rates. The lookup table is determined using historical volume-occupancy relationships. The metering rate associated with each upstream occupancy is the difference between the capacity and volume associated with the occupancy on the fundamental diagram.

For the bottleneck metering rate, bottleneck locations on the freeway must be determined. Each bottleneck must have an influence zone with one or more on-ramps associated with it. Each metered on-ramp has a weighting factor associated with it, determined both by the distance of the ramp from the bottleneck, and the historical volume on the ramp. Loop detectors must be located both upstream and downstream of the influence zone, as well as on all on-ramps (both metered and un-metered) and off-ramps. In order for the bottleneck algorithm to be invoked, two conditions must be met. The first condition is that the downstream occupancy must be greater than a threshold occupancy, indicating that the freeway section is operating above capacity. The second condition is that the freeway section must be storing vehicles, meaning that the sum of the vehicles entering the section and entering via on-ramps must be greater than the sum of the vehicles exiting the section and leaving via off-ramps. If both conditions are met, the metering rate reduction for section i for time interval t+1 is determined as follows:

$$U_{i(t+1)} = (q_{IN_{it}} + q_{ON_{it}}) - (q_{OUT_{it}} + q_{OFF_{it}}) \quad (2.16)$$

The volume reduction calculated using equation 2.16, and the weighting factors, are then used to calculate the bottleneck metering rate reduction for each ramp within the influence zone:

$$BMRR_{ji(t+1)} = U_{i(t+1)} \frac{WF_j}{\sum_j^n (WF_j)_i} \tag{2.17}$$

The bottleneck metering rate for each ramp is then calculated by subtracting the bottleneck metering rate reduction from the measured on-ramp flow during the previous interval:

$$BMR_{ji(t+1)} = q_{ON_{jt}} - BMRR_{ji(t+1)} \tag{2.18}$$

Because influence zones may overlap, each ramp may have more than one bottleneck metering rate associated with it. In that case, the most restrictive bottleneck metering rate is chosen. Finally, either the most restrictive bottleneck metering rate, or the local metering rate, which ever is more restrictive, is selected.

FLOW uses a two step queue control process. The first part is queue adjustment. When the queue for a ramp reaches a certain length, the metering rate for that ramp is increased slightly. The second step is advance queue override. When the queue reaches its maximum permissible length, the ramp meter is shut off.

2.5.5 Linked Algorithm

The Linked algorithm (Taylor et al, 1998) is based on Proportional-Intergral-Plus (PIP) control theory (McKenna, 2003). PIP control theory is an example of state variable feedback (SVF) controllers. The basis of the control design is the non-minimal state space (NMSS) description of the system to be controlled. The NMSS is formulated using the states, past value of outputs, past value of inputs, and additional integral-of-error states. A description follows, and is from McKenna (2003).

For application to traffic, a special form of the NMSS description is formulated based on the local linear model (LLM) for each point in the network. With LLM, each point in the network where measurements are obtained is modeled using the previously sampled measurements at the current location, as well as the upstream and downstream locations. This allows the model to handle both congested and uncongested traffic conditions. For the on-ramps, the on-ramp flow is used as an additional variable. The LLM for a point with an onramp is:

$$o_{j,k} = ao_{j,k-1} + bo_{j-1,k-1} + co_{j+1,k-1} + dq_{on,k-1} \quad (2.19)$$

Where:

- $o_{j,k}$ = occupancy measured at location j, time k
- $o_{j,k-1}$ = occupancy measured in previous time interval, location j
- $o_{j-1,k-1}$ = occupancy measured in previous time interval, downstream of location j
- $o_{j+1,k-1}$ = occupancy measured in previous time interval, upstream of location j
- $q_{on,k-1}$ = on-ramp flow in previous time interval
- a, b, c, d: parameters that are estimated

$$\boldsymbol{x}_k = \boldsymbol{F}\boldsymbol{x}_{k-1} + \boldsymbol{G}\boldsymbol{u}_{k-1} + \boldsymbol{D}\boldsymbol{y}_{\boldsymbol{d}\,k-1} + \boldsymbol{g_2} o_{up,k-1} + \boldsymbol{g_3} o_{down,k-1} \quad (2.20)$$

$$\boldsymbol{y}_k = \boldsymbol{H}\boldsymbol{x}_k \quad (2.21)$$

Where:

- $\mathbf{x_{k-1}}$ = state vector at time k

- $\mathbf{u_{k-1}}$ = vector of controlled inputs (on-ramp flows) at time k-1
- $\mathbf{y_{dk-1}}$ = vector of set-point occupancies
- $o_{up,k}$ = boundary condition of upstream occupancy
- $o_{down,k}$ = boundary condition of downstream occupancy
- $\mathbf{y_k}$ = vector of controlled outputs

The matrices **F, G, D,** and **H**, as well as the vectors $\mathbf{g_2}$ and $\mathbf{g_3}$ are all defined based on the parameters obtained for each LLM at each location, using equation 2.19. Suppose, for example, there is a network that has 11 sensors, numbered 0 through 11, and 4 on-ramps, which are upstream of sensors 3, 5, 7, and 9. Sensors 0 and 10 will be used for the upstream and downstream boundary conditions. The state vector for this example would be:

$$\boldsymbol{x}_k^{\mathrm{T}} = \begin{bmatrix} o_{1,k} & o_{2,k} & o_{3,k} & o_{4,k} & o_{5,k} & o_{6,k} & o_{7,k} & o_{8,k} & o_{9,k} & z_{3,k} & z_{5,k} & z_{7,k} & z_{9,k} \end{bmatrix}$$

The last four elements of this vector are the integrals of error. The integrals of error are calculated as:

$$z_{j,k} = z_{j,k-1} + \left(y_{d_{j,k}} - o_{j,k} \right) \tag{2.22}$$

The other inputs for this example would be defined as:

$$\boldsymbol{u}_k^{\mathrm{T}} = \begin{bmatrix} q_{on3,k} & q_{on5,k} & q_{on7,k} & q_{on9,k} \end{bmatrix}$$
$$\boldsymbol{y}_{\boldsymbol{d}k}^{\mathrm{T}} = \begin{bmatrix} o_{d3,k} & o_{d5,k} & o_{d7,k} & o_{d9,k} \end{bmatrix}$$
$$o_{up,k} = o_{0,k}$$
$$o_{down,k} = o_{10,k}$$

By using the parameters from equation 2.19, the matrices defining the NMSS are defined as:

$$\boldsymbol{F} = \begin{bmatrix} a_1 & c_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ b_2 & a_2 & c_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & b_3 & a_3 & c_3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & b_4 & a_4 & c_4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & b_5 & a_5 & c_5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & b_6 & a_6 & c_6 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & b_7 & a_7 & c_7 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & b_8 & a_8 & c_8 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & b_9 & a_9 & 0 & 0 & 0 & 0 \\ 0 & -b_3 & -a_3 & -c_3 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -b_5 & -a_5 & -c_5 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -b_7 & -a_7 & -c_7 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -b_9 & -a_9 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\boldsymbol{G} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ d_3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & d_5 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & d_7 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & d_9 \\ -d_3 & 0 & 0 & 0 \\ 0 & -d_5 & 0 & 0 \\ 0 & 0 & -d_7 & 0 \\ 0 & 0 & 0 & -d_9 \end{bmatrix} \quad \boldsymbol{D} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \boldsymbol{g_2} = \begin{bmatrix} b_1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad \boldsymbol{g_3} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ c_9 \\ 0 \\ 0 \\ 0 \\ -c_9 \end{bmatrix}$$

$$\boldsymbol{H} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

The vectors for $\mathbf{x_k}$ and $\mathbf{y_k}$ are then calculated using equations 2.20 and 2.21. The SVF control law is then defined as:

$$\boldsymbol{u}_k = -\boldsymbol{K}\boldsymbol{x}_k \tag{2.23}$$

Where **K** is the control gain matrix.

The control gain matrix can be obtained using a number of different SVF control approaches. One method is to use Linear Quadratic (LQ) control, where the matrix is chosen to minimize the following cost function:

$$J = \sum_{k=1}^{\infty} \left\{ \boldsymbol{x}_k{}^T \boldsymbol{Q} \boldsymbol{x}_k + \boldsymbol{u}_k{}^T \boldsymbol{R} \boldsymbol{u}_k \right\} \tag{2.24}$$

Q and **R** are symmetric positive semi-definite and symmetric positive definite matrices respectively (Taylor et al, 1998). Often, the identity matrix may be used as a starting point for the **Q** and **R** matrices.

This algorithm is a theoretical design which does not explicitly take into account queue lengths. For the UK implementation of this algorithm, the queue adjustment algorithm in the RMPS strategy was modified to be included with this algorithm.

2.6 Summary

A large number of ramp metering algorithms have been proposed. Limited field and simulation studies have been performed, mostly focusing on testing existing implantations, or various algorithms at a single location. Very little research has been done to identify conditions under which ramp metering is effective. Furthermore, little work has been performed to determine under which conditions coordination is especially useful.

Chapter 3

Simulation Study

Because of the difficulties and limitations of field study, traffic simulation was chosen as the method of study for this thesis. A simulation package known as MITSIMLab developed at MIT (Yang and Koutsopolos, 1996; Yang, 1997; Ben-Akiva et al, 1997) was used for the study.

3.1 Case Studies

The major case study for this thesis is the project “Coordination of Ramp Metering Sites” (Taylor et al, 2002). The project involves a section of the eastbound Motorway M27, near Southampton, UK. This case study involved both calibrating MITSIMLab to accurately model the traffic in Southampton, as well as enhancements to MITSIMLab to handle the ramp metering algorithms used in the study.

3.1.1 M27 Network

The M27 network currently has two on-ramps that are metered. The current algorithm is the RMPS algorithm, a local ramp metering algorithm, which is loosely based on ALINEA (Gould et al, 2002). The purpose of this project is to determine under what circumstances the linked algorithm (Taylor et al, 1998) would be beneficial. Figure 3.1, below, shows a diagram of the network:

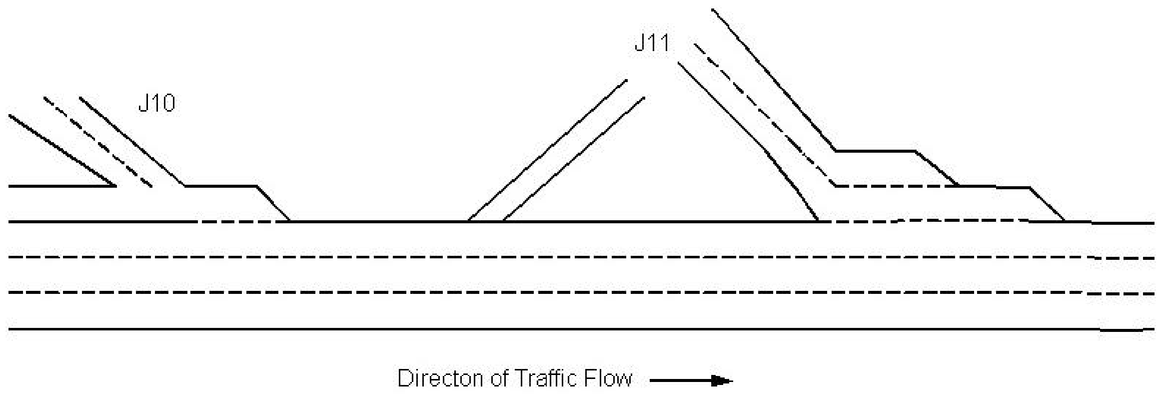


Figure 3.1: M27 Network

3.1.2 Generic Network

Because the M27 network only has two metered ramps that are spaced relatively far apart, the tests that could be run on it are somewhat limited. In order to test the effects of geometry and ramp traffic distribution, a generic network was created. Figure 3.2, below, shows a diagram of the generic network:

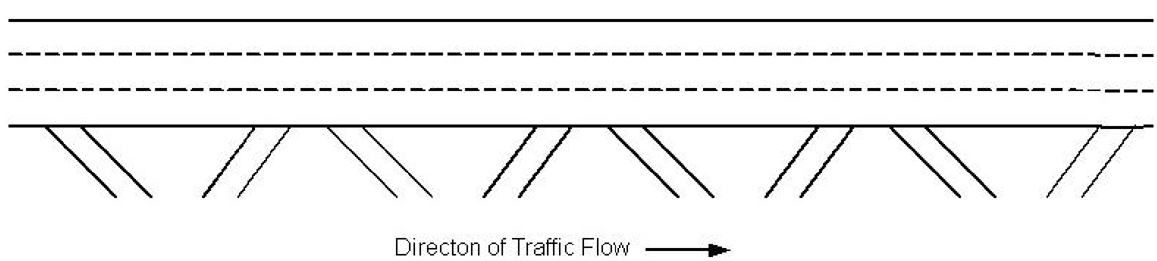


Figure 3.2: Generic Network

3.1.3 Measures of Effectiveness

In order to test the effectiveness of ramp metering, three measures of effectiveness (MOEs) were chosen. Because the primary purpose of ramp metering is to improve the mainline traffic flow, the first MOE that was used was mainline travel time. Because ramp metering can sometimes improve mainline travel time while hurting the ramp traffic, the other MOEs that were used are ramp travel time, and total travel time for all vehicles. Travel times were chosen for all three MOEs because they are a factor that drivers are sensitive to.

3.2 MITSIMLab

MITSIMLab is a powerful simulation tool that was developed at MIT by Yang (1997), and uses a variety of models to simulate various traffic networks. MITSIMLab is organized into three modules:

- Microscopic Traffic Simulator (MITSIM)
- Traffic Management Simulator (TMS)
- Graphical User Interface (GUI)

A description of each of these modules, and how they communicate with each other follows.

3.2.1 Microscopic Traffic Simulator (MITSIM)

MITSIM is the module that creates the network, and controls the flow of vehicles through the network. MITSIM is a microscopic traffic simulator: this means that individual vehicles are modeled, and individual driver behavior, such as lane changing and car-following are modeled. The main components of MITSIM are:

- **Network Components:** The road network is built using nodes, links, segments, and lanes. The network also consists of sensors, which measure and collect data such as flow, speed, and occupancy. Traffic control devices (such as ramp meters or traffic signals) are also represented, although their operation is controlled by TMS.

- **Travel Demand and Route Choice:** MITSIM accepts as input time-dependent origin-destination (OD) trip tables. MITSIM then uses a probabilistic route choice model to assign vehicles to different paths.
- **Vehicle Movement and Driving Behavior:** The OD matrix is translated into individual vehicles, which enter the network at a specific time. Each vehicle is assigned behavior characteristics (such as desired speed, aggressiveness, critical gaps for lane changing, compliance rate to control devices, etc), and vehicle characteristics (such as size, and acceleration and deceleration capabilities). The simulator then moves vehicles according to the car-following and lane-changing models. The car-following model captures the effects of the conditions ahead of the vehicle. The lane-changing model distinguishes between mandatory and discretionary lane changes.

3.2.2 Traffic Management Simulator (TMS)

TMS is the component that operates the traffic control devices in the network. MITSIMLab allows MITSIM and TMS to communicate with each other, so TMS can use the traffic surveillance data generated by MITSIM, to simulate traffic responsive control. The traffic control systems that TMS can simulate include:

- Ramp meters
- Intersection Traffic Signals
- Lane Control Signs (LCS)

- Variable Speed Limit Signs (VSLS)
- Portal Signals (PS) at tunnel entrances
- Variable Message Signs (VMS)
- In-vehicle route guidance

3.2.3 Graphical User Interface (GUI)

The GUI provides a graphical display to allow the user to watch the traffic flow. It is useful for testing, debugging, and demonstration purposes.

3.2.4 Communication between MITSIM, TMS, and GUI

Figure 3.3 shows a diagram of how MITSIM, TMS, and GUI communicate with each other.

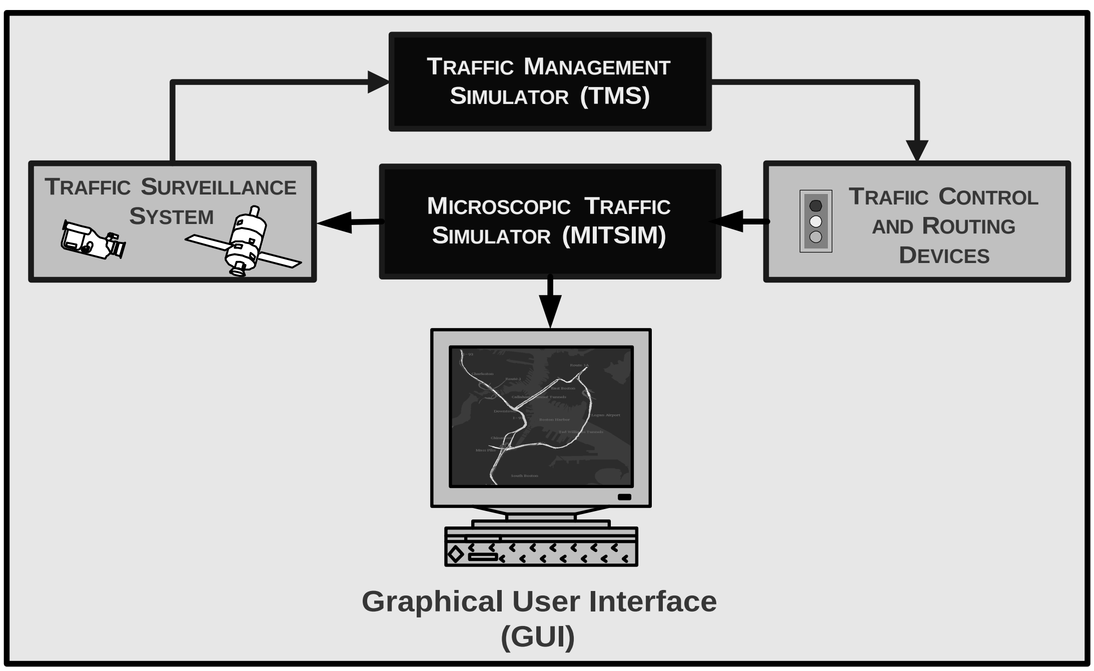


Figure 3.3: Communication Between MITSIMLab Components

Both MITSIM and TMS can either be run independently, or they can be run together. For the purposes of traffic responsive ramp metering, it is nessesary to run MITSIM and TMS together, to allow them to communicate. Also, MITSIM and TMS can be run either with or without the GUI.

3.3 Enhancements to MITSIMLab

In order to simulate the UK implementation of ALINEA, the ALINEA/Q algorithm, and the linked algorithm, it was nessesary to add new modules to MITSIMLab. All of these enhancements were made to the TMS component of MITSIMLab.

3.3.1 Ramp Metering Pilot Scheme (RMPS) Algorithm

Although this algorithm is based on ALINEA, it's actual implementation is quite different than the traditional algorithm. This algorithm is explained in Section 2.5.2, and in Gould et al (2002). The heart of the module is the function that cycles through each of the conditions, and if the condition is met, the appropriate action is performed. For more information, see the flowcharts in section 2.53. The conditions that are checked are:

- Determine if ramp metering needs to be switched on or off, and perform the appropriate start-up or shut-down sequence
- Determine if a new cycle length needs to be calculated
- Determine if queue adjustment is necessary
- Determine if the signal needs to be switched from stopping amber to red

- Determine if the signal needs to be switched from red to starting amber (in the UK, starting amber is a short phase before the signal turns green)
- Determine if the required number of vehicles were released, and switch the signal from green to stopping amber
- Determine if the maximum green time has been exceeded, and switch the signal to stopping amber
- During the startup sequence, determine if the initial green time has been exceeded, and switch the signal to stopping amber

The other significant functions in this module are the functions that calculate the smoothed flow, speed, and occupancy required by this algorithm. An example of an input file is shown in Appendix A.

3.3.2 ALINEA/Q

The ALINEA/Q algorithm uses video detectors to measure the number of vehicles in a queue. This new module contains a function to count the number of vehicles in a ramp queue, as well as to determine the number of vehicles entering a ramp, both of which are required inputs for this algorithm. An example of an input file for this algorithm is shown in Appendix B.

3.3.3 Linked Algorithm

Because of its superior ability to perform mathematical computations involving matrices and vectors, the code for this algorithm was written in MATLAB. The MATLAB engine

(Mathworks, 2003) was linked to MITSIMLab in order to allow MATLAB and MITSIMLab to communicate with each other. This procedure is described in Appendix D.

The MATLAB program requires as input a vector of occupancy measurements from each sensor. It returns as output, a vector of metering rates, which are converted into cycle lengths. This module works by calculating the occupancy measurements, and sending them to MATLAB, which it receives as a vector. MITSIMLab then calls the MATLAB program, which calculates the desired metering rates, and converts it into a cycle length. MITSIMLab then receives the cycle lengths from MATLAB, and then it proceeds just like any other ramp metering algorithm in MITSIMLab. The set point occupancies, the control gain matrix, and the function to convert the flow rates to cycle lengths are all entered in the MATLAB code, and can be very easily changed simply by modifying the MATLAB script file. An example of a MITSIMLab input file for the linked algorithm is shown in Appendix C. Figure 3.4, below, shows how MITSIMLab and MATLAB communicate with each other:

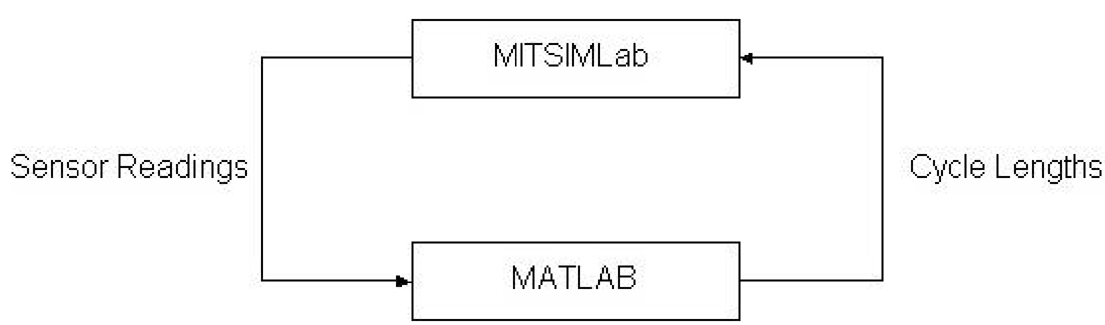


Figure 3.4: Communication between MITSIMLab and MATLAB

3.4 Calibration and Validation of MITSIMLab

In order for simulation results to be useful, it is necessary to calibrate and validate the model, comparing it to field data. This is necessary because drivers in different areas behave differently, and similar roads may have very different operating characteristics. The calibration process has two main steps:

- Origin-Destination Flow Estimation
- Parameter Calibration

3.4.1 Origin-Destination Flow Estimation

MITSIM requires as input a table of origin-destination (OD) flows. Because it is extremely difficult and costly to accurately measure OD flows in the field, it is often necessary to estimate them by using point flows. Traffic data from the loop detectors was obtained from the UK Highway agency. The first five weeks of data, in the spring of 2001, the ramp meters were switched off, making this convenient data to use for calibration purposes. To use as field data input, the data was aggregated into 15-minute intervals, during the peak traffic period, 6 AM – 9 AM, Monday through Friday. In order to insure that the model represents typical conditions, any data which was affected by an incident, according to the incident log, was excluded, times when a sensor was malfunctioning, according to the sensor log, were excluded, and UK bank holidays were excluded.

In order to estimate the OD flows, a procedure involving the least-squares optimization method was used (Cascetta et al, 1993). This method was implemented in MATLAB.

This program works by iteratively calling MITSIMLab, and then reading the MITSIMLab output files after each iteration. The program requires as input a file containing the field count data for each day, a variance-covariance matrix, and a seed OD matrix. The seed OD matrix was calculated using the average counts, but not taking into account the travel time between sensors (which the OD estimation program does). MITSIMLab returns traffic counts at each sensor, as well as returning an assignment matrix. The assignment matrix, contains the proportion of vehicles from each OD pair for each time interval that crosses a sensor during a given interval. The assignment matrix is weighted by multiplying by a weighting matrix, **W**. Each day of measured flows and the seed OD flows are also weighted by multiplying by the variance-covariance matrix, and placed in a vector **Y**. The simulated traffic counts are placed in a vector **X**. The OD flows is then calculated by minimizing the error, using the least squares method:

$$\mathbf{OD} = \arg\min\,[(\mathbf{AX}-\mathbf{Y})^T\,\mathbf{W}(\mathbf{AX}-\mathbf{Y}) + (\mathbf{X}-\mathbf{X}_{\mathbf{seed}})^T\,\mathbf{U}(\mathbf{X}-\mathbf{X}_{\mathbf{seed}})] \qquad (3.1)$$

This process is then repeated for a set number of iterations, until a final OD matrix is created. For more information, see (Darda, 2002).

3.4.2 Calibration of Parameters

In addition to estimating the OD flows, it is also necessary to calibrate parameters that affect traffic flow. This is needed because drivers on different roads in different areas will drive differently. An iterative, probabilistic algorithm known as the Box (1956) method was used for the calibration.

A sensitivity analysis was performed, showing that there are three parameters that have the most effect on traffic flow. Two of these parameters, the sensitivity parameters for acceleration and deceleration, are from the car-following model (Ahmed, 1999). These sensitivity parameters capture network conditions, weather conditions, road geometry, and other information not captured by the other explanatory variables. The third parameter that was calibrated was a scaling factor to adjust a driver's desired speed, based on the speed limit. Because the M27 network has only one path between any two points, there was no need to calibrate the route choice parameters.

The Box method was implemented in MATLAB. A matrix containing traffic speed at each sensor for the days (similar to the counts matrix), as well as a variance-covariance matrix was needed for input. The Box algorithm works as follows:

1. MITSIM is run K times, to determine a complex of K points. The user defines the parameters used in the first run, and a random number generator determines the parameters used in the next K-1 runs. For each run, the objective function is calculated and saved. The objective function is defined as:

$$F(k) = \sum_{i} \sum_{d} (s_h(i,d) * v(i,d) - s_s(i) * v(i,d))^2 \qquad (3.2)$$

where:

$s_h(i,d)$ = historical speed for sensor-time combination i, on day d

$v(i,d)$ = variance for sensor-time combination i, on day d

$s_s(i)$ = simulated speed for sensor-time combination I

2. If an explicit constraint is violated, the point is moved a small distance ä inside the limit. If an implicit constraint is violated, the point is moved one half of the distance to the centroid of the other points.
3. The data point with the highest objective value is removed from the complex. That point is replaced by a point with parameters á times as far from the centroid of the remaining points as the distance of the rejected point, on the line joining the rejected point and the centroid. A value of á = 1.3 was used, as recommended by Box. MITSIM is then run again to obtain the objective function for the new data point.
4. If the same point repeats as the point with the highest objective function on consecutive trials, it is moved one half of the distance to the centroid of the remaining points.
5. If the new point violates the constraints, it is adjusted, as in step 2.
6. The algorithm converges when the objective function is within â units for ã consecutive iterations. Also, the algorithm is stopped when a predefined number of iterations are completed. In either case, the parameter combination with the lowest objective function value becomes the selected parameters.

For more information on this algorithm, see (Darda, 2002).

3.4.3 Calibration Results

Figures 3.5 through 3.11 plot the simulated counts versus the actual counts for each sensor location.

Figure 3.5: Sensor 9252B - Start of study area

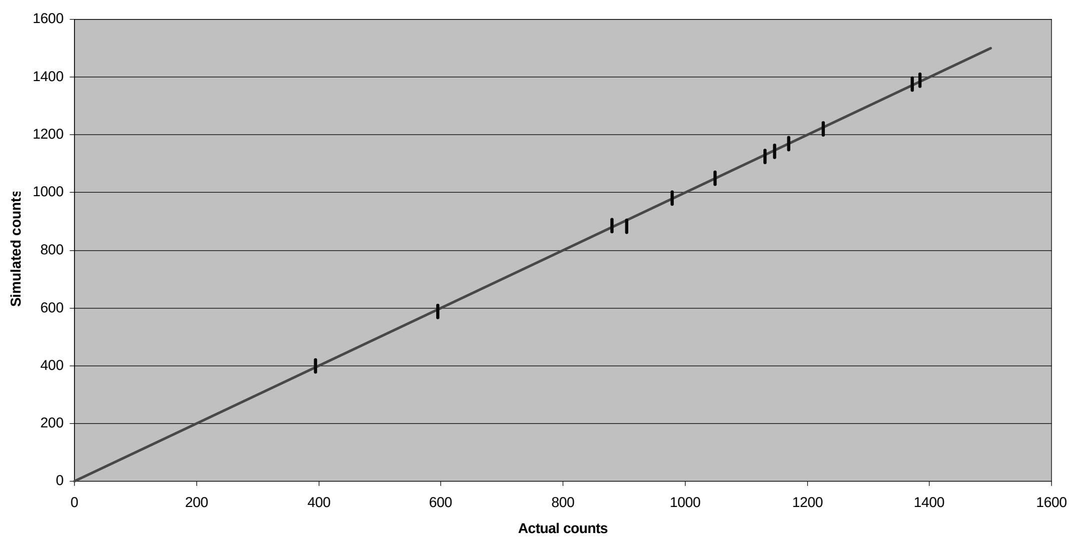


Figure 3.6: Sensor 9376A2 - Junction 10 On-ramp

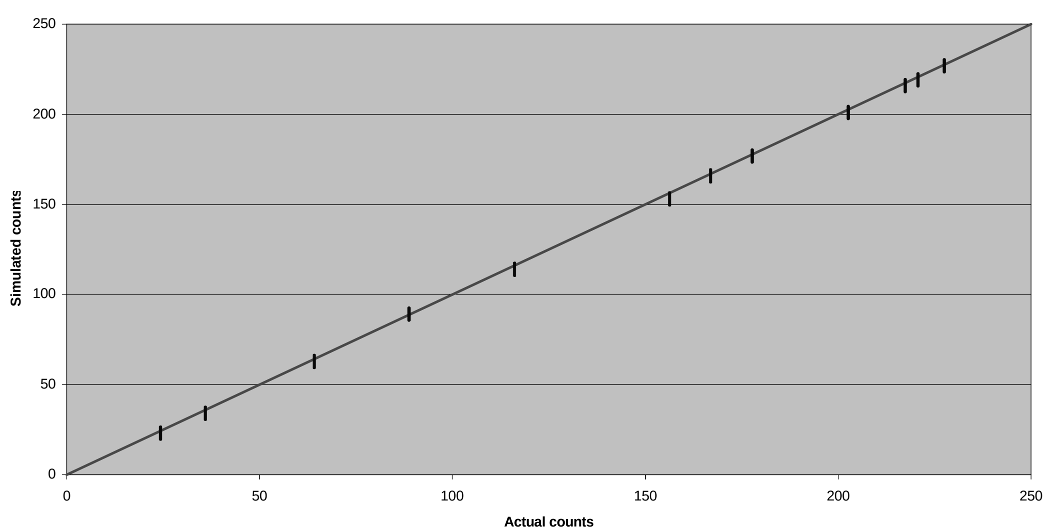


Figure 3.7: Sensor 9376A3 - West of Junction 10

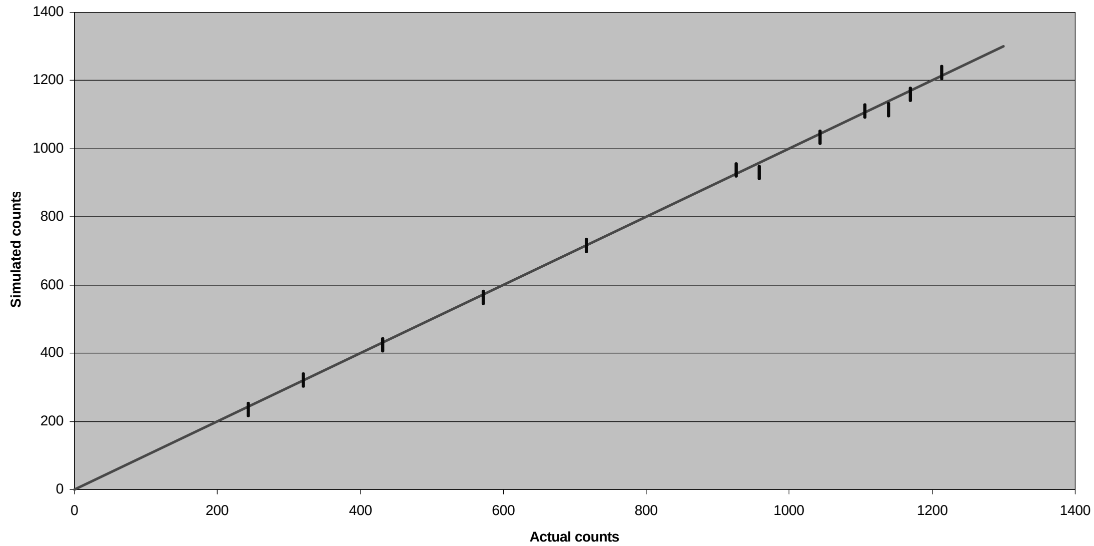


Figure 3.8: Sensor 9385A - East of Junction 10

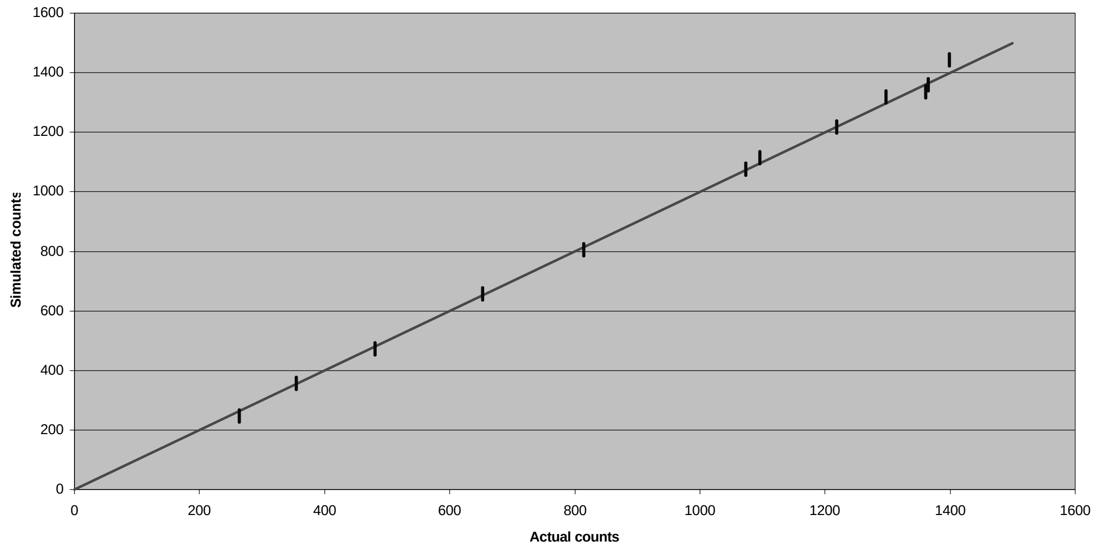


Figure 3.9: Sensors 9394A1 / 2 - Between Junctions 10 and 11

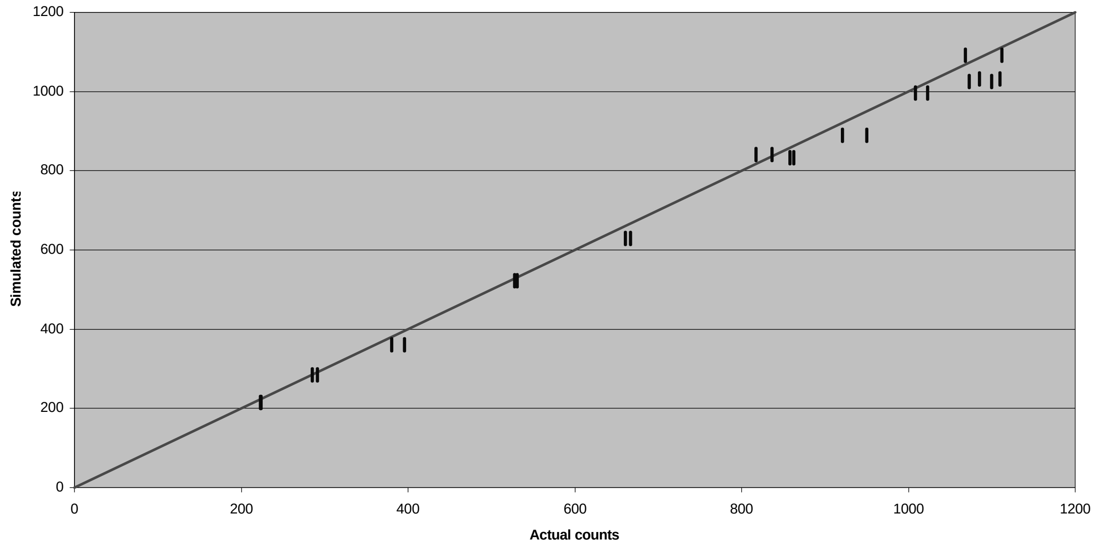


Figure 3.10: Sensor 9396A2 - Junction 11 Onramp

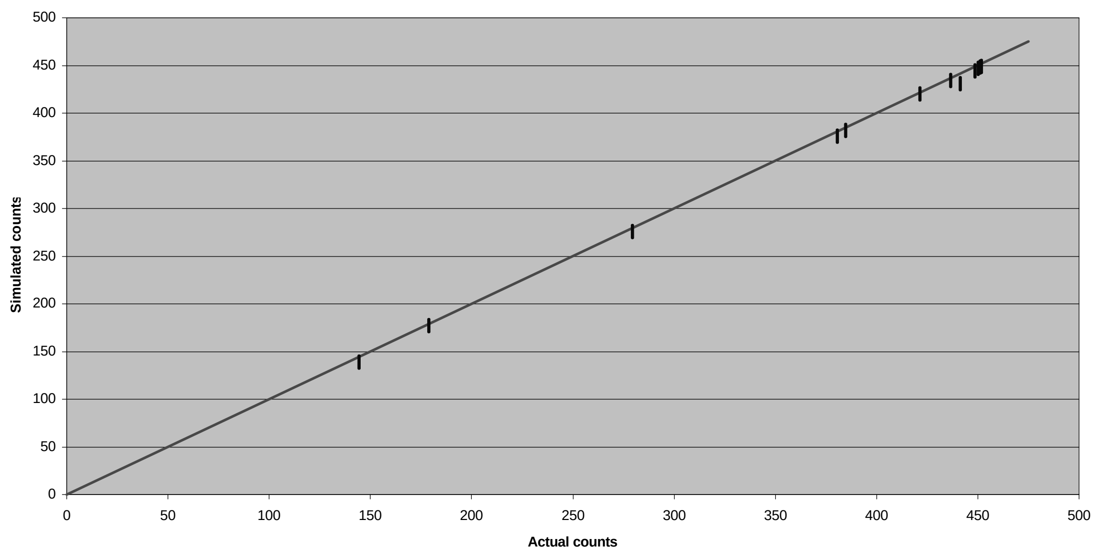
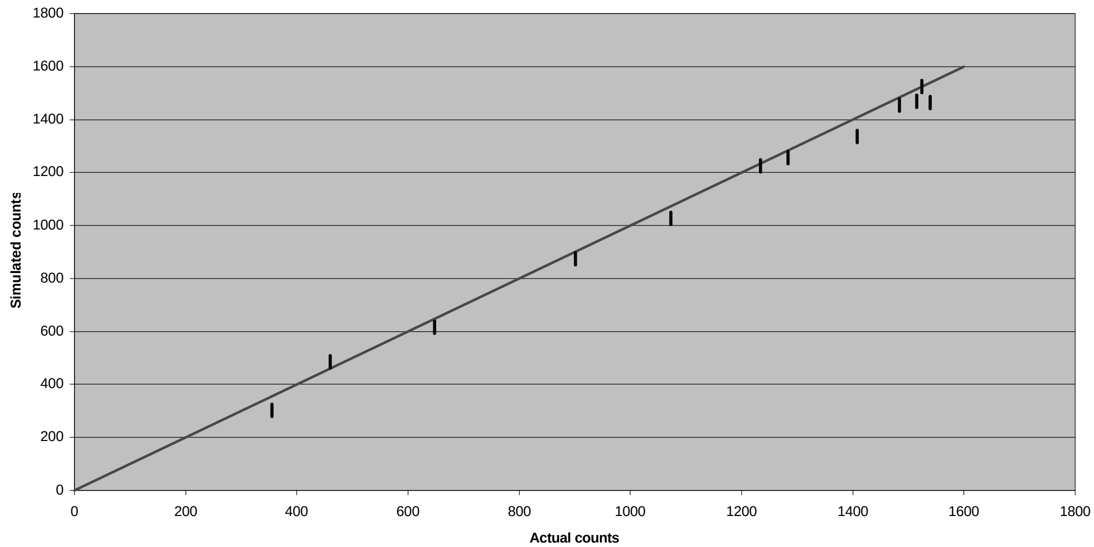


Figure 3.11: Sensor 9413A - East of Junction 11



Figures 3.12 through 3.15 plot the simulated versus actual speeds for each sensor location.

Figure 3.12: Sensor 9376A3 - West of Junction 10

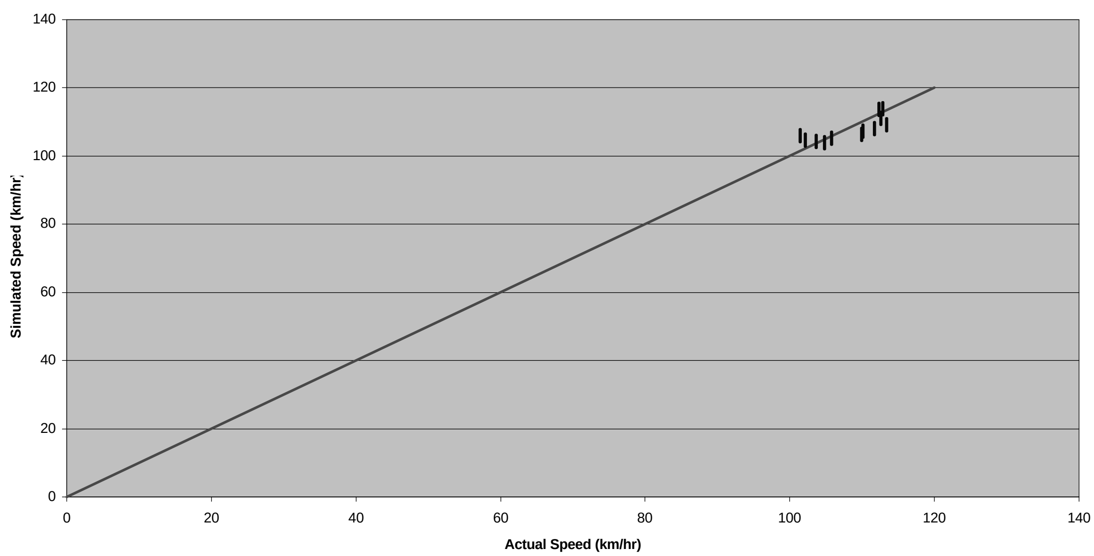


Figure 3.13: Sensor 9385A - East of Junction 10

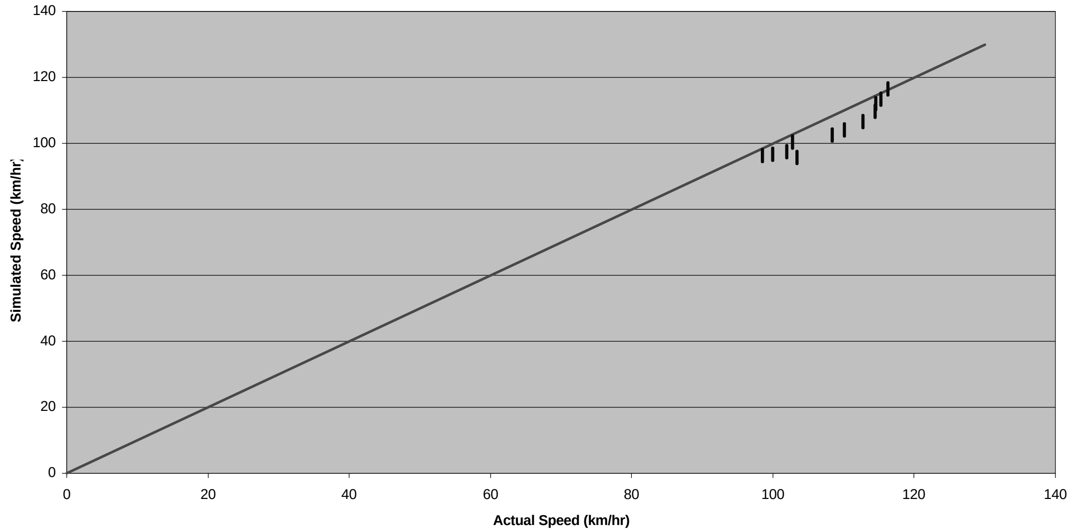


Figure 3.14: Sensors 9394A1 / 2 - Between Junctions 10 and 11

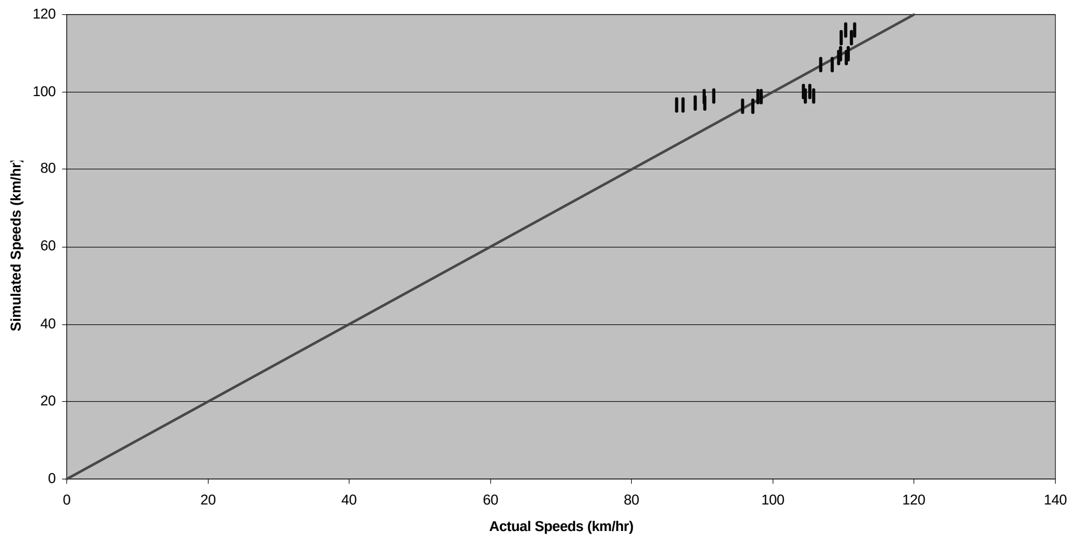
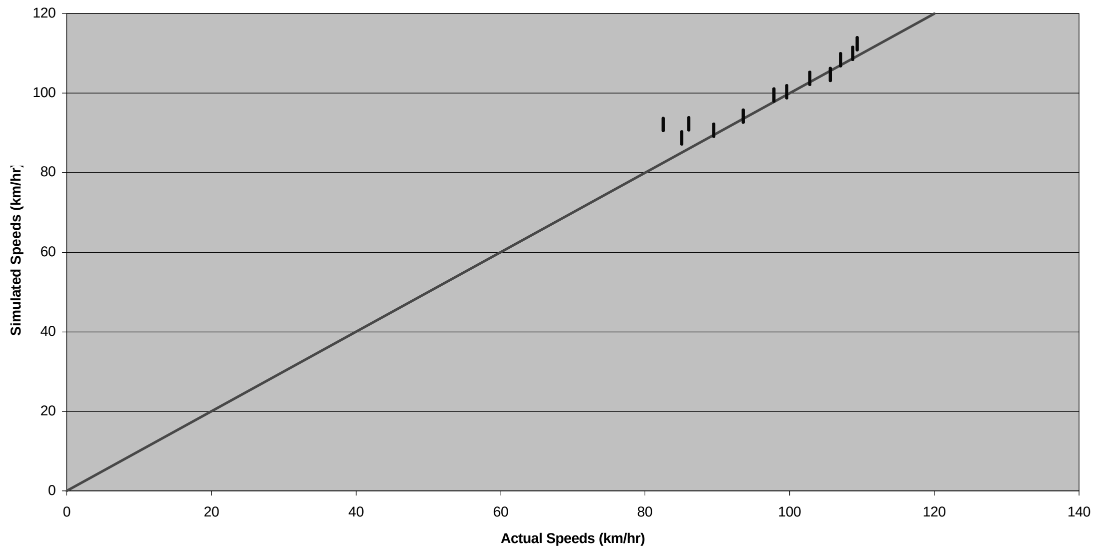


Figure 3.15: Sensor 9413A - East of Junction 11



3.4.4 Validation

The final step is validation. Validation is a process where different scenarios are simulated, to determine that the calibrated model accurately reflects reality, under different conditions than what was used in the calibration. There were two criteria checked:

- Comparison of the simulated traffic counts with the actual traffic counts
- Comparison of the simulated traffic speeds with the actual traffic speeds

Figures 3.16 through 3.22 show the simulated counts versus the actual counts:

Figure 3.16: Sensor 9252B - Start of study area

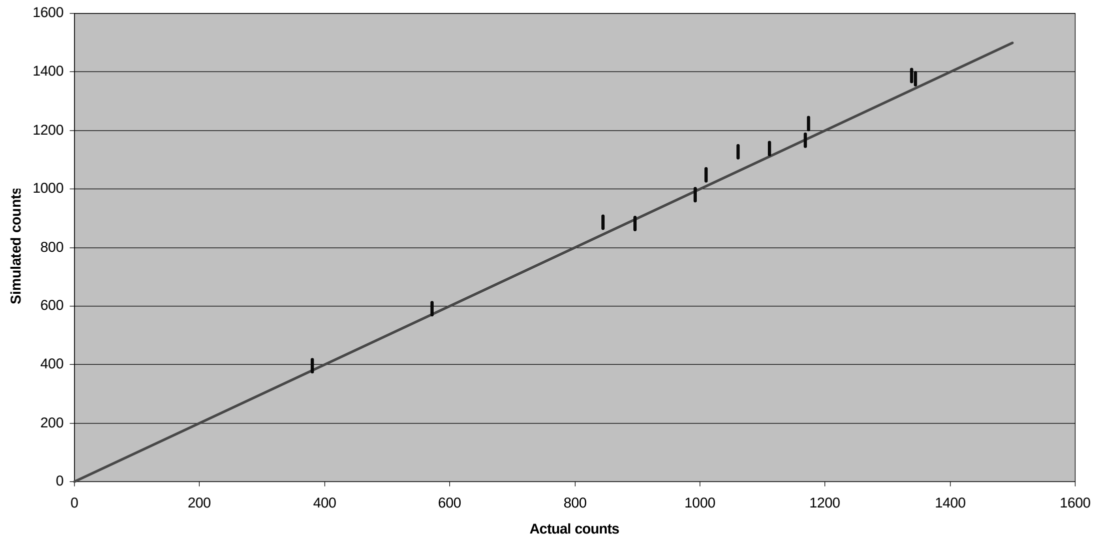


Figure 3.17: Sensor 9376A2 - Junction 10 On-ramp

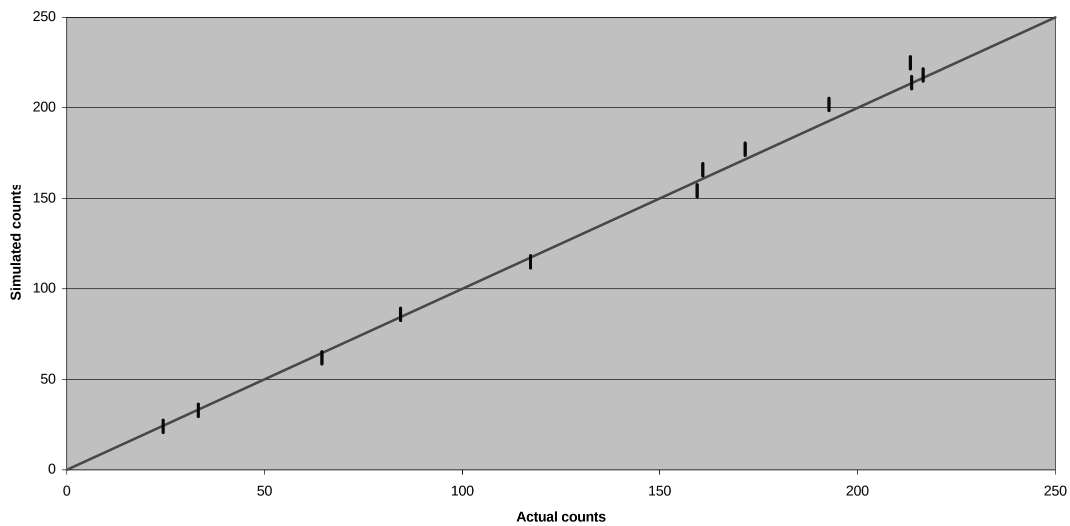


Figure 3.18: Sensor 9376A3 - West of Junction 10

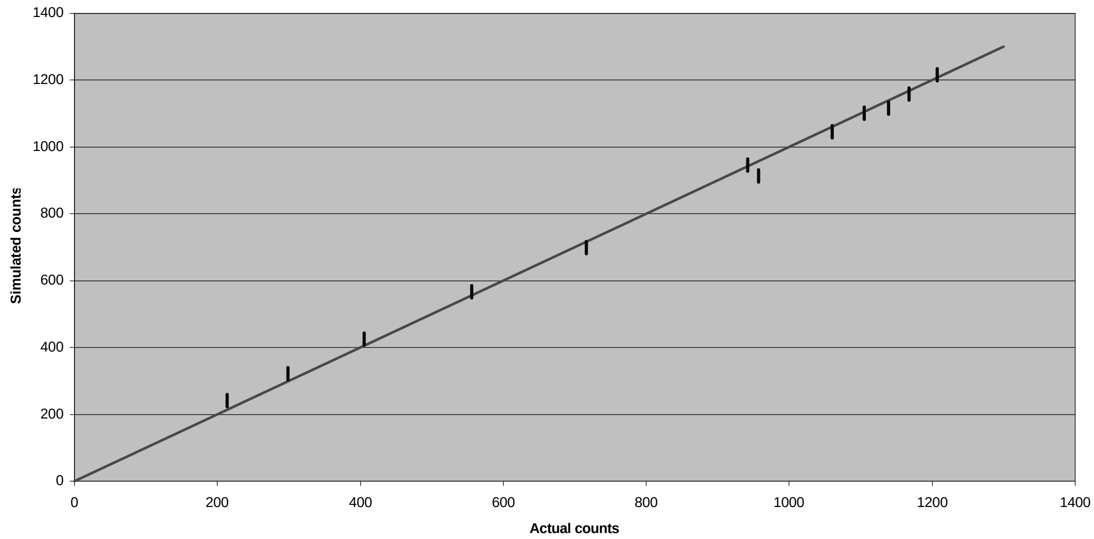


Figure 3.19: Sensor 9385A - East of Junction 10

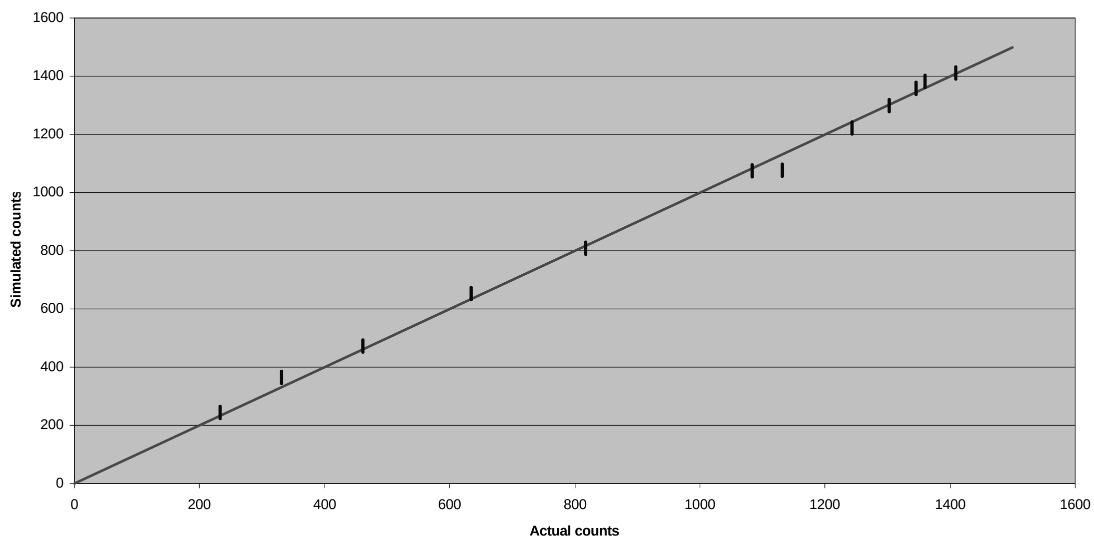


Figure 3.20: Sensors 9394A1 / 2 - Between Junctions 10 and 11

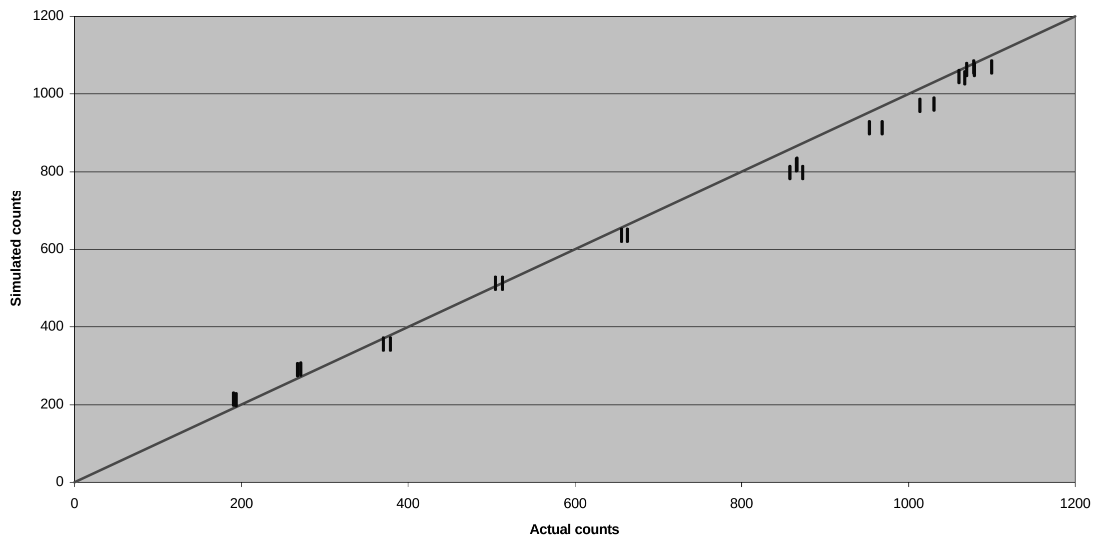


Figure 3.21: Sensor 9396A2 - Junction 11 Onramp

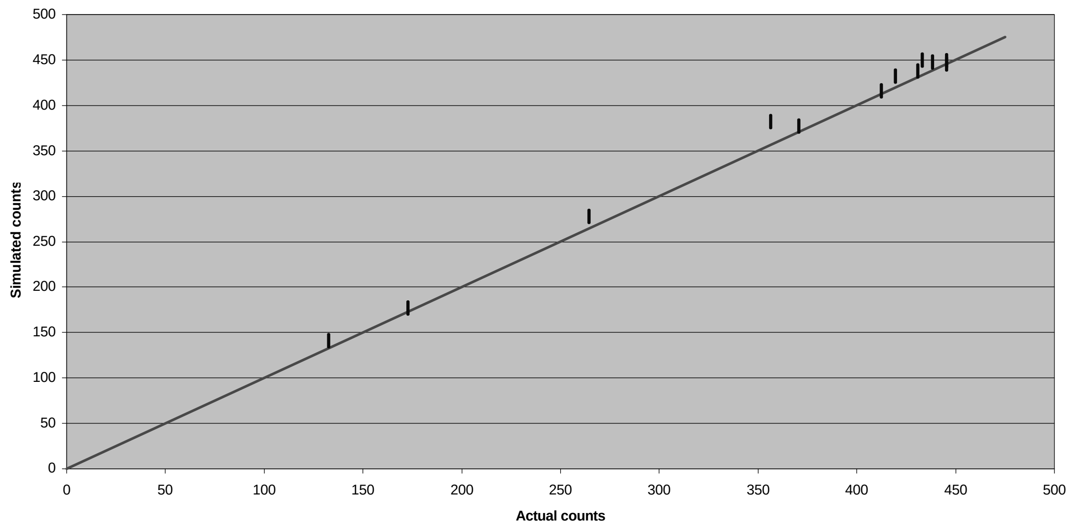
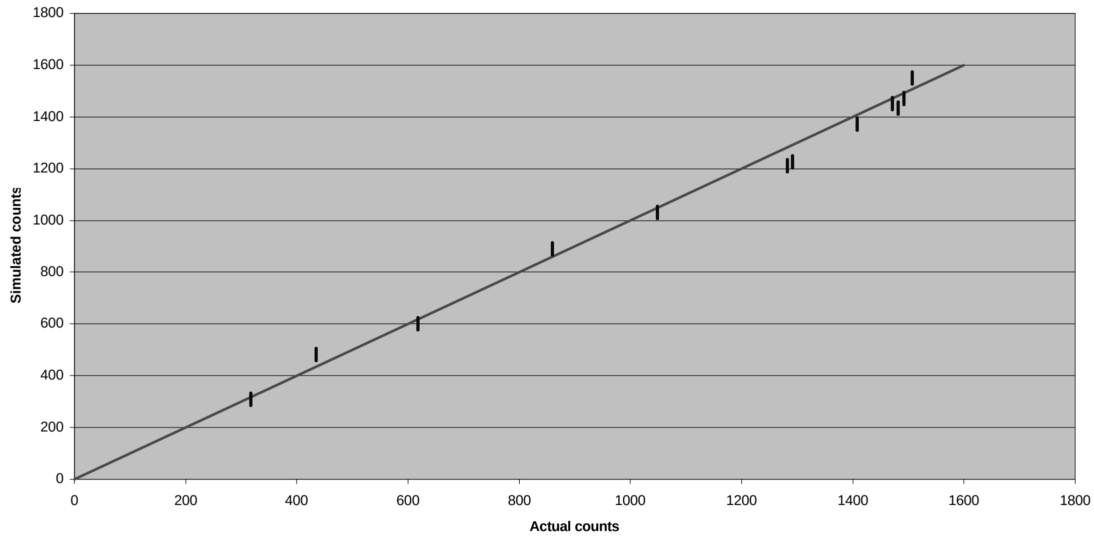


Figure 3.22: Sensor 9413A - East of Junction 11



Figures 3.23 through 3.26 plot the simulated speeds against the actual speeds:

Figure 3.23: Sensor 9376A3 - West of Junction 10

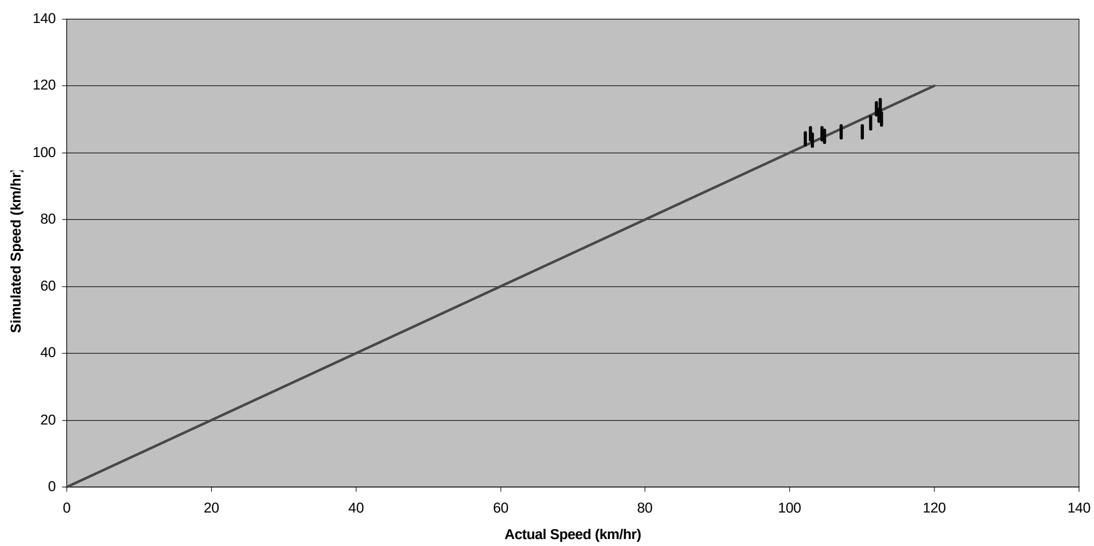


Figure 3.24: Sensor 9385A - East of Junction 10

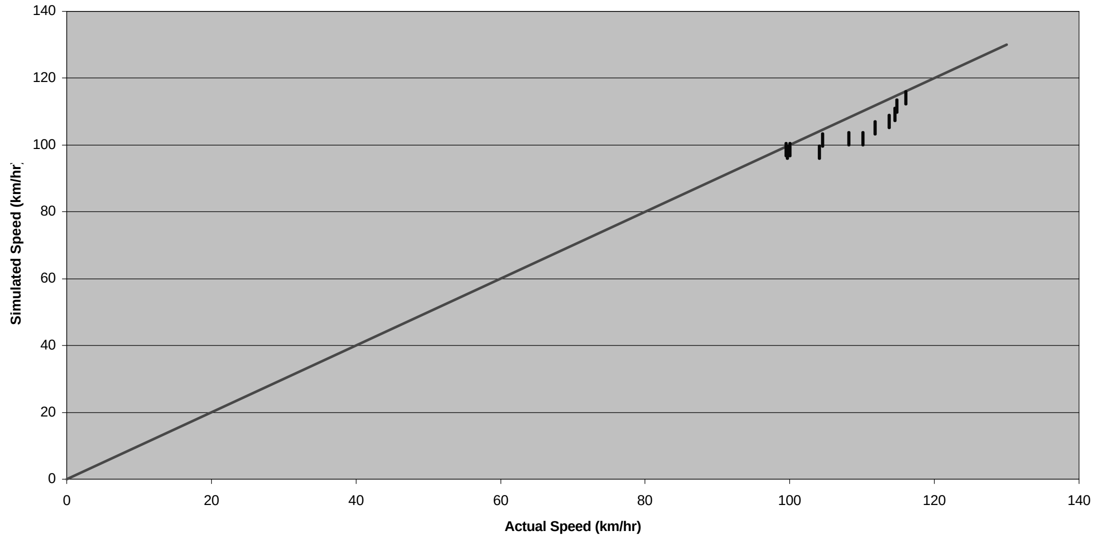


Figure 3.25: Sensors 9394A1 / 2 - Between Junctions 10 and 11

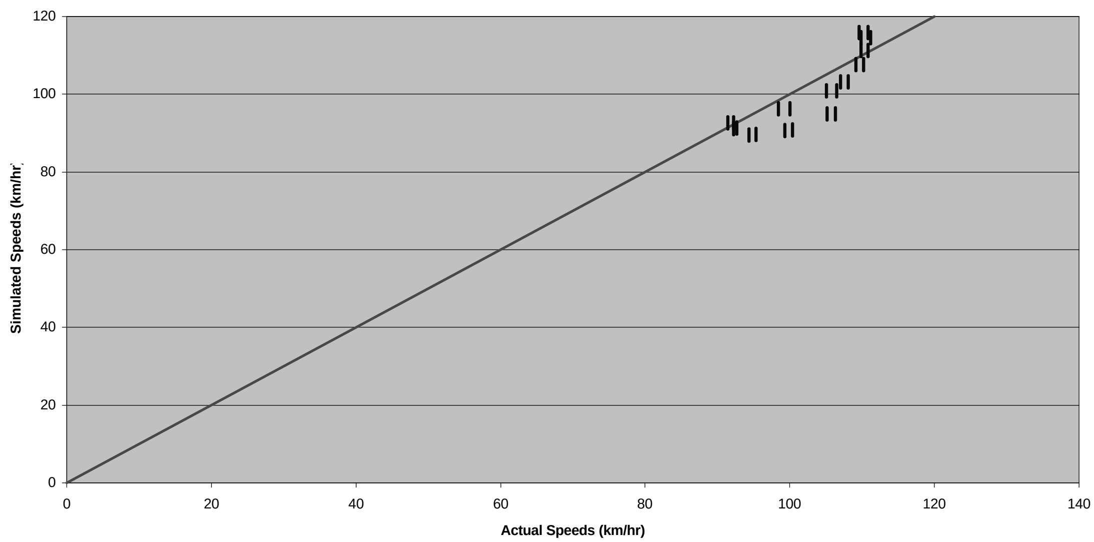
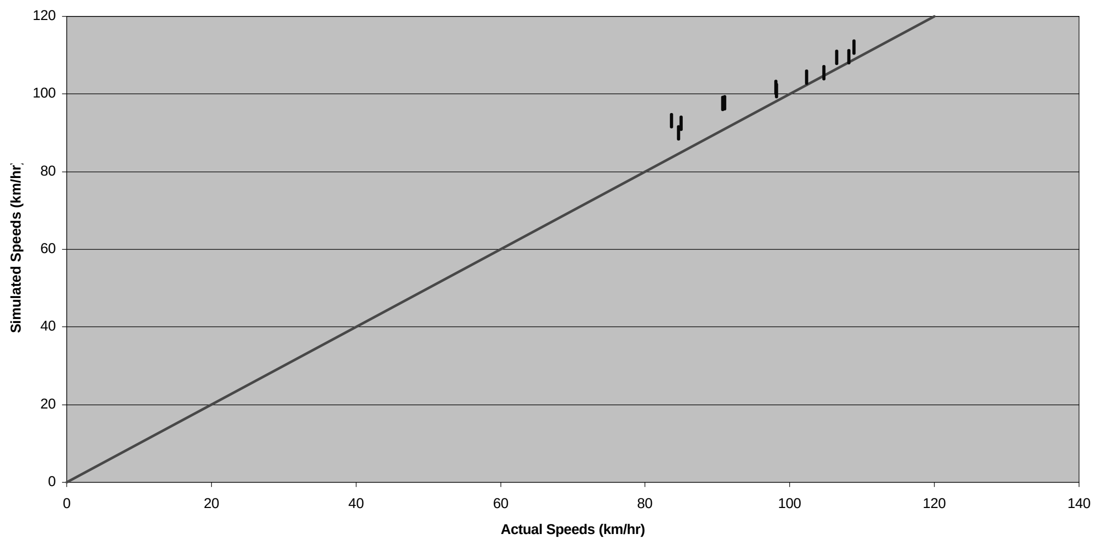


Figure 3.26: Sensor 9413A - East of Junction 11



Chapter 4

Results From M27 Network

The local ALINEA-based RMPS strategy, as well as the other four ramp metering strategies being studied were tested on the M27 network. Because none of the strategies showed any great improvements, various scenarios were tested to determine if and when ramp metering was ever effective for this network.

4.1 Results Using Average Demand Level

Data on travel time was collected in MITSIMLab in order to evaluate the effectiveness on ramp metering. In order to ensure statistical significance, the simulation was run 10 times, and the results were averaged, and the average travel time for vehicles traveling the mainline, vehicles entering from an on-ramp, as well as the total average travel time for all vehicles were calculated.

Table 4.1, below, shows the percentage of time savings for both the RMPS algorithm, as well as each of the four algorithms being studied:

	Mainline	Ramps	Total
RMPS Algorithm	-0.4%	-3.8%	-1.0%
ALINEA	-0.3%	-0.1%	-0.2%
ALINEA / Q	-0.1%	0.1%	0.0%
FLOW	-0.6%	-8.2%	-2.1%
Linked Algorithm	-0.2%	-2.9%	-0.7%

Note that most of these values are negative, meaning that ramp metering actually increased travel time. This is due to the fact that at the average demand level, traffic is not too congested, so ramp metering cannot do much to improve traffic flow. Also, due the heavy on-ramp volumes (particularly at Junction 11), the queue adjustment algorithm was activated most of the time, and this reduced the efficiency of the ramp metering algorithm.

4.2 Testing Other Scenarios

Because ramp metering was shown to hurt traffic when average demand existed, other scenarios were tested to determine if and when ramp metering is effective for this network. Because the average traffic volume was too light to cause much congestion, scenarios were tried with 10% and 20% increase in traffic demand. Also, because the average scenario had very high ramp volumes, scenarios were tested with medium and low percentages of traffic on the ramps.

4.2.1 Experimental Design

Since there are 3 demand levels and 3 ramp to mainline traffic ratios, there are a total of 3x3, or 9 combinations. Because this is a relatively small number of combinations, a full factorial design, testing all 9 combinations, can be run. In order to ensure statistical significance, each scenario was run 10 times.

Table 4.2, below, shows the parameters used for each scenario:

Scenario	Demand Level	Ramp Volume
1	100%	100%
2	110%	100%
3	120%	100%
4	100%	80%
5	110%	80%
6	120%	80%
7	100%	60%
8	110%	60%
9	120%	60%

4.2.2 Results

Table 4.3, below, shows the percent time savings for mainline traffic for each algorithm:

			Corridor Travel Time Savings (%)				
Scenario	Demand Level	Ramp Volume	RMPS	ALINEA	ALINEA / Q	Flow	Linked
1	100%	100%	-0.4%	-0.3%	-0.1%	-0.6%	-0.2%
2	110%	100%	-0.5%	-0.4%	-0.2%	-0.8%	-0.9%
3	120%	100%	1.1%	-2.1%	-1.5%	-2.9%	-3.8%
4	100%	80%	-0.2%	0.0%	-0.5%	-0.4%	-0.2%
5	110%	80%	0.2%	-1.0%	-0.2%	-1.1%	-0.7%
6	120%	80%	0.2%	1.7%	1.9%	1.5%	1.9%
7	100%	60%	-0.4%	0.0%	-0.2%	-0.7%	-0.5%
8	110%	60%	-1.4%	-1.6%	-1.9%	-1.2%	-0.6%
9	120%	60%	1.4%	1.4%	1.7%	1.5%	1.8%

Table 4.4, below, shows the percent time savings for ramp traffic for each algorithm:

			Ramp Travel Time Savings (%)				
Scenario	Demand Level	Ramp Volume	RMPS	ALINEA	ALINEA / Q	Flow	Linked
1	100%	100%	-3.8%	-0.1%	0.1%	-8.2%	-2.9%
2	110%	100%	-8.1%	-1.2%	-0.6%	-11.2%	-3.8%
3	120%	100%	-23.4%	-3.2%	-4.6%	-15.1%	-6.4%
4	100%	80%	-4.6%	0.0%	-0.9%	-5.9%	-2.7%

5	110%	80%	-8.5%	-2.8%	-1.2%	-10.3%	-3.7%
6	120%	80%	-9.1%	-4.0%	-4.2%	-14.3%	-1.2%
7	100%	60%	-5.4%	0.1%	-0.2%	-5.4%	-2.5%
8	110%	60%	-13.4%	-2.5%	-3.8%	-11.6%	-2.6%
9	120%	60%	-13.1%	-2.1%	-1.8%	-12.5%	-1.1%

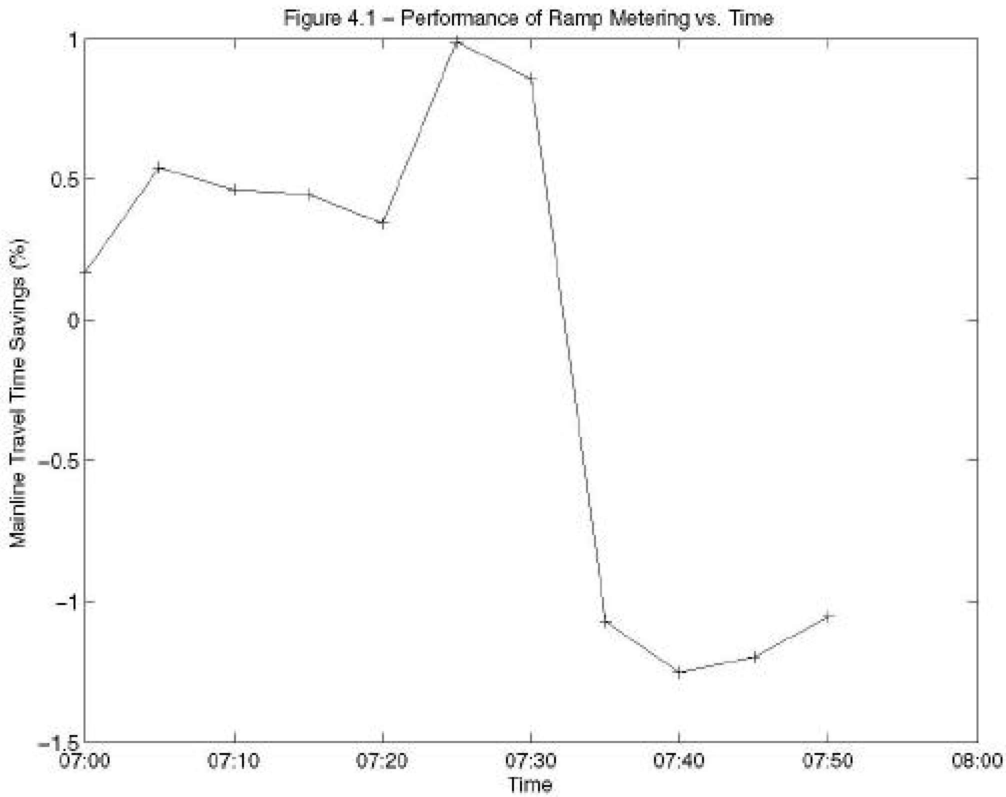
Table 4.5, below, shows the percent time savings for total traffic for each algorithm

			Total Travel Time Savings (%)				
Scenario	Demand Level	Ramp Volume	RMPS	ALINEA	ALINEA / Q	Flow	Linked
1	100%	100%	-1.0%	-0.2%	0.0%	-2.1%	-0.7%
2	110%	100%	-2.0%	-0.6%	-3.0%	-2.9%	-1.5%
3	120%	100%	-3.9%	-2.4%	-2.2%	-5.5%	-4.4%
4	100%	80%	-0.9%	0.0%	-0.5%	-1.2%	-0.6%
5	110%	80%	-1.2%	-1.3%	-0.4%	-2.6%	-1.2%
6	120%	80%	-1.2%	0.9%	1.0%	-0.8%	1.4%
7	100%	60%	-1.0%	0.0%	-0.3%	-1.2%	-0.7%
8	110%	60%	-2.8%	-1.7%	-2.1%	-2.5%	-0.8%
9	120%	60%	-0.2%	1.0%	1.3%	-0.1%	1.5%

4.3 Analysis of Results

These results show that for the M27 network, ramp metering does not significantly improve traffic under any situation, and significantly hurts it under most. The best results were when the ramp volumes were moderate or low, and the total volume was high. The main reason is because the network has only two ramps and they are so far apart, that ramp traffic is not a major cause of delays. When the volume was at 120% and the network was congested, ramp metering showed some benefits, as long as the ramp volume was low enough to not constantly invoke the queue control algorithm.

One particularly surprising result was that in some cases, most notably at the highest total volume (120%) and the highest ramp volume (100%), ramp metering actually increased the mainline travel time. There are two possible reasons for this. The first reason, specific to M27, is that since this freeway is near a port, it has very high truck traffic. Because trucks often accelerate slowly, especially on an upgrade, they have a very difficult time reaching freeway speed after being stopped at a ramp meter. The heavy truck traffic also causes instability in the measured occupancies, leading to an unstable metering rate. If one cycle has a very restrictive rate, it could cause a queue to build up on the ramp. In the next cycle, if the metering rate is much less restrictive, it can spill the queue onto the freeway, causing more vehicles to enter in a shorter period of time than would have without the metering. The second reason is that at very high mainline volumes and very high ramp volumes, the ramp meters spend much of their time in queue override mode. Figure 4.1, below, shows a plot of the ramp metering performance over time. Notice how it shows some slight benefits early in the simulation, but once the queue override mode is activated, the freeway is flooded with vehicles, traffic seriously degrades, and is unable to recover for a while:



Because ramp metering did not show much benefit on this network, a generic network was created in order to test more variables. In particular, this network will test the effects of geometry, which cannot be tested on the M27 network, and is likely to be a major factor in the performance of ramp metering. This case study is discussed in Chapter 5.

Chapter 5

Results from Generic Network

The generic network was created in order to allow greater flexibility to test various scenarios to determine if and when ramp metering is effective. Four variables were tested: demand level, spacing between ramps, distribution of traffic among ramps, and percentage of traffic using the ramps. Four levels of demand were tested, four ramp spacings were tested, three distribution patterns were tested, and three ramp traffic percentages were tested. Four ramp metering strategies were tested: ALINEA, ALINEA / Q, Flow, and Linked. All four algorithms were described in great detail in Chapter 2.

5.1 Experimental Design

This experiment has 4 OD levels, 4 ramp spacings, 3 distribution patterns, and 3 ramp traffic percentages. That is a total of 4x4x3x3, or 144 combinations. Due to time constraints and constraints of computing power, it was not possible to test all 144 combinations. Therefore, a fractional factorial design was needed. In order to insure that there was no interaction between variables, an orthogonal matrix, with 16 scenarios was used (Addelman, 1962). In order to ensure statistical significance, each scenario was run 10 times, and the results were averaged.

Table 5.1, below, shows the 16 combinations that were used:

Scenario	Demand Level	Ramp Spacing	Ramp Distribution	Ramp Percentage
1	6600 vph	2000 ft	1	25%
2	6600 vph	4000 ft	2	35%
3	6600 vph	8000 ft	3	30%
4	6600 vph	16,000 ft	2	30%
5	6900 vph	2000 ft	2	30%
6	6900 vph	4000 ft	1	30%
7	6900 vph	8000 ft	2	35%
8	6900 vph	16,000 ft	3	25%
9	7200 vph	2000 ft	3	35%
10	7200 vph	4000 ft	2	25%
11	7200 vph	8000 ft	1	30%
12	7200 vph	16,000 ft	2	30%
13	7500 vph	2000 ft	2	30%
14	7500 vph	4000 ft	3	30%
15	7500 vph	8000 ft	2	25%
16	7500 vph	16,000 ft	1	35%

Note that for the ramp distribution, #1 means that the upstream ramps have the heaviest volume, #2 means that all ramps have equal volume, and #3 means that the downstream ramps have the heaviest volume.

5.2 Results

Table 5.2, below, shows the average mainline travel time savings for each scenario:

					Corridor Travel Time Savings (%)			
Scenario	Demand Level	Ramp Spacing	Ramp Distribution	Ramp Percentage	ALINEA	ALINEA / Q	Flow	Linked
1	6600 vph	2000 ft	1	25%	-0.6%	-0.3%	0.0%	-0.4%
2	6600 vph	4000 ft	2	35%	0.8%	0.2%	0.5%	0.3%
3	6600 vph	8000 ft	3	30%	-0.7%	-1.2%	-1.1%	-1.0%
4	6600 vph	16,000 ft	2	30%	0.1%	0.0%	-0.4%	0.2%
5	6900 vph	2000 ft	2	30%	2.3%	2.5%	2.4%	0.9%
6	6900 vph	4000 ft	1	30%	-0.5%	1.1%	0.8%	-0.1%
7	6900 vph	8000 ft	2	35%	-0.8%	0%	-0.3%	-0.7%
8	6900 vph	16,000 ft	3	25%	0.5%	0.4%	0.5%	0.7%
9	7200 vph	2000 ft	3	35%	7.6%	8.6%	8.1%	5.2%
10	7200 vph	4000 ft	2	25%	3.0%	2.0%	2.7%	1.1%

11	7200 vph	8000 ft	1	30%	-0.4%	0.2%	0.0%	-1.0%
12	7200 vph	16,000 ft	2	30%	-0.2%	-0.1%	-0.2%	-0.3%
13	7500 vph	2000 ft	2	30%	3.4%	5.6%	8.0%	2.9%
14	7500 vph	4000 ft	3	30%	1.9%	3.2%	3.3%	2.5%
15	7500 vph	8000 ft	2	25%	1.1%	0.9%	0.5%	0.2%
16	7500 vph	16,000 ft	1	35%	-0.3%	-0.2%	0.0%	-0.1%

Table 5.3, below, shows the average ramp travel time savings for each scenario:

					Ramp Travel Time Savings (%)			
Scenario	Demand Level	Ramp Spacing	Ramp Distribution	Ramp Percentage	ALINEA	ALINEA / Q	Flow	Linked
1	6600 vph	2000 ft	1	25%	-16.5%	-17.6%	-10.9%	-14.0%
2	6600 vph	4000 ft	2	35%	-6.2%	-9.1%	-14.0%	-6.0%
3	6600 vph	8000 ft	3	30%	-3.2%	-4.2%	-4.1%	-3.2%
4	6600 vph	16,000 ft	2	30%	-2.1%	-1.9%	-2.1%	-1.4%
5	6900 vph	2000 ft	2	30%	-92.9%	-108.0%	-55.2%	-75.9%
6	6900 vph	4000 ft	1	30%	-15.5%	-17.0%	-26.3%	-10.4%
7	6900 vph	8000 ft	2	35%	-5.6%	-8.1%	-5.1%	-2.9%
8	6900 vph	16,000 ft	3	25%	-1.6%	-1.8%	-1.9%	-0.9%
9	7200 vph	2000 ft	3	35%	-106.3%	-114.4%	-108.6%	-101.7%
10	7200 vph	4000 ft	2	25%	-28.7%	-34.1%	-40.3%	-24.2%
11	7200 vph	8000 ft	1	30%	-8.5%	-13.7%	-12.0%	-4.2%
12	7200 vph	16,000 ft	2	30%	-3.3%	-3.9%	-4.8%	-2.0%
13	7500 vph	2000 ft	2	30%	-113.3%	-121.3%	-115.9%	-107.2%
14	7500 vph	4000 ft	3	30%	-51.5%	-59.4%	-68.5%	-35.8%
15	7500 vph	8000 ft	2	25%	-9.0%	-13.7%	-12.1%	-3.3%
16	7500 vph	16,000 ft	1	35%	-8.6%	-10.5%	-8.1%	-3.7%

Table 5.4, below, shows the average total travel time savings for each scenario:

					Total Travel Time Savings (%)			
Scenario	Demand Level	Ramp Spacing	Ramp Distribution	Ramp Percentage	ALINEA	ALINEA / Q	Flow	Linked
1	6600 vph	2000 ft	1	25%	-2.6%	-2.4%	-1.4%	-2.0%
2	6600 vph	4000 ft	2	35%	-0.4%	-1.3%	-1.9%	-7.2%
3	6600 vph	8000 ft	3	30%	-1.1%	-1.6%	-1.5%	-1.3%
4	6600 vph	16,000 ft	2	30%	-0.5%	-0.5%	-0.8%	-0.3%
5	6900 vph	2000 ft	2	30%	-11.2%	-13.2%	-5.8%	-10.0%
6	6900 vph	4000 ft	1	30%	-2.6%	-1.4%	-3.0%	-1.5%

7	6900 vph	8000 ft	2	35%	-1.7%	-1.4%	-1.1%	-1.1%
8	6900 vph	16,000 ft	3	25%	0.0%	-0.1%	0.1%	0.4%
9	7200 vph	2000 ft	3	35%	-7.9%	-8.0%	-7.7%	-9.4%
10	7200 vph	4000 ft	2	25%	-0.5%	-2.0%	-2.1%	-1.7%
11	7200 vph	8000 ft	1	30%	-1.6%	-1.9%	-1.8%	-1.4%
12	7200 vph	16,000 ft	2	30%	-0.8%	-1.0%	-0.8%	-0.6%
13	7500 vph	2000 ft	2	30%	-8.8%	-7.5%	-4.6%	-8.4%
14	7500 vph	4000 ft	3	30%	-4.7%	-4.5%	-5.5%	-2.3%
15	7500 vph	8000 ft	2	25%	-0.1%	-0.8%	-1.0%	-0.3%
16	7500 vph	16,000 ft	1	35%	-2.5%	-3.4%	-2.1%	-1.2%

5.3 Analysis of Results

These results show that ramp metering can potentially cause great improvements to the mainline travel time, although the conditions under which it is beneficial are relatively narrow. The most important factors affecting the usefulness of ramp metering are the ramp spacing and the demand level. Ramp metering is only useful when the ramps are spaced relatively close together. This is because when the ramps are close together, the bottleneck caused by the traffic merging has a significant impact on the mainline traffic. However, when the ramps are spaced further apart, the ramp traffic has less of an effect on the mainline, and thus ramp metering has less potential to be effective. This was shown in both the generic network, and in the M27 network (where the two on-ramps are spaced very far apart). Also, ramp metering was shown to be useful only at the higher OD levels, where congestion occurred. At the lower OD levels, ramp metering causes vehicles to unnecessarily stop before entering the freeway, and this reduces the efficiency of the traffic flow.

In all cases, ramp metering caused an increase in the travel time for ramp vehicles, as well as the total travel time. However, the numbers are misleading for several reasons.

First of all, particularly in the networks with the short ramp spacing, vehicles spend a disproportionate amount of time on the ramps, and not much time on the network. Once they finally enter the network, their travel time would be faster than it was with no ramp metering. In real life, vehicles entering a freeway will remain on the freeway for a while, rather than just disappearing when they reach an artificial end of network.

Also, in the real world, ramp metering has side effects that are beneficial to traffic flow that are not captured in the simulation. In real life, decreasing the travel time for mainline vehicles while increasing travel time for ramp vehicles encourages drivers taking short trips to find an alternate route. This encourages use of the freeway for distance travel, where it is intended to be, while encouraging local traffic to use local roads. This leads to greater efficiency of the total network. Also, in real life, when one ramp has a very long queue, drivers are encouraged to instead use another nearby ramp, if one is available, in order to decrease traffic on a ramp that is over capacity, while making use of other ramps that are below capacity. Another side effect is that ramp meters decrease accidents on the freeway, which further improves travel time, but is not captured in the simulation. One more solution that could be used to improve ramp traffic would be to add an HOV priority lane, in order to allow buses and carpools to bypass the queue and enter the freeway with little delay. HOV priority lanes would encourage drivers to switch to modes with greater occupancy, and would reduce the number of cars on the freeway, and decrease congestion and decrease travel time.

The results showed that there was little difference between the algorithms, and that coordination generally did not significantly improve travel time. This is consistent with other studies. Generally, ALINEA/Q improved mainline traffic, but hurt ramp traffic, relative to the traditional ALINEA algorithm. This is because the traditional ALINEA algorithm uses a binary queue control algorithm, where once the queue rises above a certain threshold, ramp metering is completely suspended, causing vehicles to flood the freeway. However, the ALINEA/Q algorithm only increases the metering rate enough to maintain the maximum allowable queue length, in order to improve the mainline traffic flow as much as possible, without completely backing up the ramp traffic.

The results were consistent with Hasan (1999) showing that FLOW was most useful at very high traffic volumes. The FLOW algorithm has two major flaws. The first flaw is that the local part of the algorithm is based on percent occupancy, which, according to Smaradgis and Papageorgiou (2003), is the least effective local algorithm. This algorithm measures only upstream occupancy, and is an open loop algorithm, not taking into effect the actual performance of the traffic downstream of the ramp. The second flaw is that it requires the operator to predetermine where a bottleneck can occur, and requires the ramp weighs to be determined offline, using historic data, with no theoretical formula to determine the weights.

The Linked algorithm is theoretically the most promising algorithm, although the results showed that it was not very effective. Since the Linked algorithm has not yet ever been used in the field, all that was developed was a theoretical algorithm, with no field implementation. In particular, no queue control was developed for the theoretical algorithm. In this implementation, the same queue control algorithm used in the RMPS algorithm was used, which is not based on any theoretical framework. The linked algorithm may performed better if it was refined to field use, and if a more robust queue control algorithm was used.

It is also important to note that most simulation studies of ramp metering were performed on macroscopic simulators, while MITSIMLab is a microscopic simulator. The macroscopic simulators are not able to model the effects of individual vehicles. Smaradgis and Papageorgious (2003) used a macroscopic simulator, and shows the ALINEA algorithm to be very effective, since it kept the downstream occupancy at the set point, with very little deviation. However, in MITSIMLab, as well as in real life, congested traffic flow is unstable, and individual drivers behave differently, and thus it is not possible for the downstream occupancy to always be at the optimal level. Also, many studies involving ALINEA study ALINEA with no queue control at all, whereas the traditional ALINEA algorithm studied here used a binary queue control algorithm.

5.4 Regression Analysis

For each of the four algorithms, a regression analysis was performed on the corridor travel time in order to determine the sensitivity of each parameter. Each parameter is converted to dummy variables, and the following equation is used:

$$Y = a_0 + b_1X_1 + b_2X_2 + b_3X_3 + b_4X_4 + b_5X_5 + b_6X_6 + b_7X_7 + b_8X_8 + b_9X_9 + b_{10}X_{10} + e \quad (5.1)$$

Where:

- Y = Corridor Travel Time Savings (%)
- X_1 = 1 if OD Demand at 6900 vph, 0 otherwise
- X_2 = 1 if OD Demand at 7200 vph, 0 otherwise
- X_3 = 1 if OD Demand at 7500 vph, 0 otherwise
- X_4 = 1 if Ramp Spacing at 4000 ft, 0 otherwise
- X_5 = 1 if Ramp Spacing at 8000 ft, 0 otherwise
- X_6 = 1 if Ramp Spacing at 16000 ft, 0 otherwise
- X_7 = 1 if Upstream Ramps have most traffic, 0 otherwise
- X_8 = 1 if Downstream Ramps have most traffic, 0 otherwise
- X_9 = 1 if 30% of total traffic enters from on-ramps
- X_{10} = 1 if 35% of total traffic enters from on-ramps

5.4.1 ALINEA

Table 5.5, below, shows the results of the regression analysis for ALINEA

	Coefficients	Standard Error	t-statistic
á	1.945	1.203	1.616
â 1	0.475	0.949	0.501
â 2	2.600	0.949	2.740
â 3	1.861	0.993	1.873
â 4	-1.875	0.949	-1.976
â 5	-3.375	0.949	-3.556
â 6	-2.914	0.993	-2.934
â 7	-1.427	0.873	-1.635
â 8	1.113	0.822	1.354
â 9	-0.321	0.802	-0.401
â 10	0.943	1.172	0.805

Adjusted R^2 = 0.629

The intercept coefficient, á, shows that in the case of a total demand of 6600 vph, ramp spacing of 2000 ft, even distribution among on-ramps, and 25% ramp traffic, the corridor travel time improves by nearly 2% when ALINEA is used. The t-statistics show that performance substantially increases when the total demand reaches 7200 vph, by 2.6%, and less so when the demand reaaches 7500 vph, by only 1.9%. This makes sense, because at a demand level of 7200 vph, ramp metering has a chance to improve travel time, while at the greater congestion of 7500 vph, ramp metering is less effective, since traffic will always be congested. The t-statistics also show that larger ramp spacings significantly reduce the effectiveness of ramp metering, while ramp traffic distribution and percentage of ramp traffic do not have any significant effect on performance. When the ramp spacing reaches 8000 ft, the ramp metering performance decreases by 3.4%, making ramp spacing the most sensitive parameter.

5.4.2 ALINEA / Q

Table 5.6, below, shows the regression analysis for the ALINEA / Q algorithm:

	Coefficients	Standard Error	t-statistic
á	1.138	1.279	0.889
â 1	1.325	1.009	1.313
â 2	3.000	1.009	2.973
â 3	3.225	1.056	3.054
â 4	-2.475	1.009	-2.453
â 5	-4.125	1.009	-4.088
â 6	-3.550	1.056	-3.362
â 7	-0.663	0.928	-0.714
â 8	1.363	0.874	1.559
â 9	0.750	0.853	0.879
â 10	2.100	1.246	1.686

Adjusted R^2 = 0.685

This analysis, as expected, shows that ALINEA / Q behaves similarly to ALINEA. With a total demand of 6600 vph, ramp spacing of 2000ft, 25% ramp traffic, and even distribution among ramps, ramp metering improves corridor travel time by 1.1%. As with ALINEA, the ramp spacing and OD level have the strongest impact on performance. The major difference between ALINEA and ALINEA / Q is that ALINEA / Q performs best at, the highest OD level, 7500 vph, where it improves travel time by an additional 3.2%. This is because at the highest OD level, the queue control algorithm is invoked more frequently, and thus ALINEA / Q's more efficient queue control algorithm has more of an effect on performance.

5.4.3 FLOW

Table 5.7, below, shows the regression analysis for the FLOW algorithm:

	Coefficients	Standard Error	t-statistic
á	1.900	1.451	1.309
â 1	1.100	1.145	0.961
â 2	2.900	1.145	2.533
â 3	3.633	1.198	3.033
â 4	-2.800	1.145	-2.446
â 5	-4.850	1.145	-4.236
â 6	-4.217	1.198	-3.519
â 7	-1.017	1.053	-0.966
â 8	1.050	0.991	1.059
â 9	0.750	0.968	0.775
â 10	1.733	1.413	1.227

Adjusted R^2 = 0.667

This analysis shows that FLOW performs similarly to the other algorithms. With a total demand level of 6600 vph, ramp spacing of 2000 ft, 25% ramp traffic, and even distribution among ramps, FLOW improves mainline travel time by 1.9%. These results are consistent with Hasan (1999) showing that FLOW is most effective at very high OD levels, with a total demand of 7500 vph improving the performance of ramp metering by an additional 3.6%. As with the other algorithms, of all the parameters, ramp spacing had the most effect on the travel time savings. When the ramp spacing reaches 8000 ft, the effectiveness of the ramp metering is reduced by 4.9%, which is a greater decrease than in the local algorithms. This is because when the ramps are spread far apart, the traffic at one ramp has little impact on the traffic at another ramp, which makes coordination less useful.

5.4.4 Linked Algorithm

Table 5.8, below, shows the regression analysis for the Linked Algorithm:

	Coefficients	Standard Error	t-statistic
á	0.657	0.757	0.868
â 1	0.425	0.597	0.712
â 2	1.475	0.597	2.471
â 3	1.902	0.625	3.046
â 4	-1.200	0.597	-2.011
â 5	-2.775	0.597	-4.649
â 6	-1.723	0.625	-2.758
â 7	-0.673	0.549	-1.226
â 8	1.275	0.517	2.467
â 9	0.179	0.504	0.354
â 10	1.210	0.737	1.642

Adjusted R^2 = 0.735

This regression shows that the Linked algorithm performs fairly similarly to the other algorithms. A total demand of 6600 vph, a ramp spacing of 2000 ft, 25% ramp traffic, and even ramp traffic distribution shows that the linked algorithm improves mainline travel time by 0.7%. Like FLOW, the other coordinated algorithm, this algorithm is shown to perform best under highest demand levels, where at a demand level of 7500 vph, the linked algorithm improves travel time by an additional 1.9%. One result unique to this algorithm is that it performs significantly better when the downstream on-ramps have the most volume, improving performance by an additional 1.3%. This can be explained that due to the fact that this is a preventative (rather than reactive) algorithm, it can use the upstream conditions to predict congestion at the downstream end of the network, while the other algorithms do not have this ability.

Chapter 6

Conclusions

This project showed that under the right conditions, with closely spaced ramps and heavy traffic, ramp metering can be very beneficial to the mainline traffic. However, the conditions under which ramp metering is beneficial are fairly narrow. Although ramp metering may increase delays to the ramp traffic, this can actually promote more efficient use of the network.

6.1 Summary of Findings

- Ramp metering has the most significant impact on mainline traffic flow when the ramps are spaced closely together. Ramp metering can significantly improve traffic when the ramps are spaced at 2000 ft. However, once the ramp spacing reaches around 8000 ft, ramp metering ceases to have any significant benefits. This is because closely spaced ramps can significantly impact the flow of traffic on the mainline, whereas if the ramps are spread farther apart, they have less of an impact
- Ramp metering is only useful at high volumes, where flow breaks down. In order for ramp metering to be effective, the traffic volumes upstream of the ramp must be below capacity, while the traffic volumes downstream of the ramp must be above capacity

- The traditional ALINEA algorithm performs best when the total demand is slightly above capacity, 7200 vph in this case. When the volume reaches 7500 vph, the controller spends much of its time in queue override, which causes ramp metering to shut off, and defeats any benefits.
- ALINEA / Q performs significantly better than ALINEA at very high traffic volumes, 7500 vph. This is because rather than using a binary on / off queue algorithm, this algorithm takes into account both mainline traffic conditions as well as queue length in calculating the metering rate. The queue override algorithm raises the metering rate just enough to maintain the queue at its maximum allowable length, making maximum use of the ramp queue storage space.
- The coordinated algorithms: FLOW and the Linked Algorithm, also perform better at very high traffic volumes, 7500 vph. This is because the coordinated nature of these algorithms make them better able to handle highly congested traffic at locations away from the ramp, and allows ramps upstream of a bottleneck to be metered more restrictively, rather than placing all the burden on a single ramp.
- For coordinated algorithms, the performance degrades at higher ramp spacings, starting around 8000 ft, even more than the local algorithms. This is because when the ramps are spaced further apart, traffic at one ramp has very little impact on the traffic at another ramp, thus defeating the purpose of coordination.
- When the downstream ramps have the most traffic, the Linked algorithm performs significantly better, while the other algorithms are not sensitive to traffic

distribution. This is due to the predictive nature of the linked algorithm, which allows it to use upstream measurements to predict the downstream conditions, and meter each ramp accordingly

- Ramp metering, by itself, significantly increases delay to ramp traffic and to the total traffic. However, this encourages more efficient use of the network. It encourages long distance traffic to use the freeway, while local traffic would be encouraged to use local streets.

6.2 Future Work

Because ramp metering by itself was shown to increase total travel time, due to increased ramp travel time, further studies can be performed to study the side effects of ramp metering, and how they can improve overall traffic:

- The effect of diversion to local streets could be studied. Currently, many drivers enter a freeway, and exit only a few exits later. This frequent entering and exiting causes turbulence in the traffic stream. A long ramp queue may encourage some drivers who would only be entering the freeway for a short distance to divert to a local street, thus leaving more space on the freeway for long distance traffic, which it is intended to serve.
- The effect of traffic diverting to less congested ramps could be studied. Ramp metering can often cause a long queue on a single, heavily congested on-ramp. Ramp metering may encourage traffic using this ramp to instead enter via a different ramp, with a shorter queue and excess capacity, making the most out of existing capacity.

- The effect of having an HOV priority lane could be studied. An HOV priority lane would allow buses and carpools to jump the queue, avoiding most of the delay caused by ramp meters. This may encourage drivers who previously drove a single occupancy vehicle to start carpooling or riding a bus, in order to reduce the number of cars on the road.
- Enhancements can be made to the field implementation of the Linked algorithm. In particular, a more efficient queue control algorithm could be developed. Since this algorithm has not yet been used in the field, only the theoretical algorithm has been developed, which has not yet been refined for use in the field. In particular, the algorithm, as designed, has no queue control built into it. For this particular implementation, the queue control algorithm used in the RMPS on M27 was used. A more efficient algorithm that takes into account both the mainline volumes and the queue lengths, such as the RMPS queue control algorithm, could be used here. Also, the queue control algorithm proposed by Gordon (1996) could be used.

Appendix A

Sample Input File for RMPS Algorithm

```
07:00:00        # Becomes in effect at 7:00AM
{
  2       #ControllerID - Junction 10
  9       #ControllerType (9 = RMPS)
  60    #updateStepSize_ , step size for calculating parameters
  12      #SignalType (ramp meter)
  1       # NumEgresses (number of outgoing links)
  10  # Ramp Number
   {
    2  # nApproaches (number of signals)
     {
          0 1       # SignalIDs
     }
    { 0.250 #Flow smoothing coeficient
      0.250 #Speed smoothing coeficient
      1.000 } #Ocupancy smoothing coeficient


#Switch on and off parameters
 60 # On / off review period (sec)
300 # Minimum flow threshold (vph)
3 # Number of lanes in mainline
1 # number of mainline sensors per lane
# Sensors    Rising Flow   Rising speed  Falling Flow   Falling speed
{
  27 28 29       4800         53           3900          47
}
#Note: speed is given in mph, even though the UK data is in km/h

# Cycle Length thresholds
#  Rising Occupancy   Falling Occupancy   Cycle length
{
   0               0                 12
   16.20             14.60             15
   18.20             16.40             20
   20.20             18.20             30
   21.70             19.10             40
   22.20             20.00             60
```

```
}
# Number of Mainline lanes
3

# Number of Mainline Detectors per lane
1

#Detector numbers
{
   27 28 29
}


{
  9 # release loop number
  2 # number of vehicles permitted to cross line per cycle
  0 # lane offset (sec)
  2 # amber time with no queue (sec)
  1 # amber time with queue (sec)
 30 # maximum idle time (sec)
  4 # green time if no release loop is present (sec – never used)
  3 # minimum red time (sec)
 30 # start sequence green time (sec)
 15 # start sequence red time (sec)
 60 # maximum starting green time (sec)
  1 # starting amber time (sec)
 20 # green time when queue is present (sec)
 10 # red time when queue is present (sec)
  3 # amber time when queue is present (sec)
  2 # extended starting amber time (sec)
}

# Queue control parameters
# number of queue loop detectors per lane
5
# loop_number weight
{
  10 9
  12 11
  14 15
  16 30
  18 50
  11 9
  13 11
  15 15
  17 30
```

```
  19 50
}

# metering rate adjustment thresholds for queue control
{
   100 200 300 9998 9999
}

# occupancy level thresholds for queue control
{
   20 40 60
}
} # end of controller

} # end of all controllers
```

Appendix B

Sample Input File for ALINEA / Q Algorithm

```
7:00:00          # Becomes in effect at 7:00AM
{
  2      #ControllerID
  15     #ControllerType (15 = ALINEA / Q)
  30   #updateStepSize_ , step size for calculating parameters (sec)
  12     #SignalType (ramp meter)
  1      # NumEgresses (number of outgoing links)
   {
    2  # nApproaches (number of signals)
     {
       0 1       # SignalIDs
     }
     2
     # 1 = Single Metering, 2 = Platoon Metering
      {
       240 #minMeteringRate_      queueing concern
       2640 #maxMeteringRate_      capacity
       30 #cycle_
      2 #yellow
      2 #lost time
      0 #minRed_
      26 #max_red
     }
    {
       70.0 #regulator   a constant parameter
       30.0 #desiredMeasurement_  target occupancy (%)
      3  # num of mainline occupancy detectors
      { 27 28 29 } #mainline occupancy detector IDs
    }
   // Queue Control Parameters
    {
      40     # Maximum Allowable queue length (veh)
      2      # Number of ramp lanes
```

```
      28  #num of queue detectors for queue override per lane
         {
       47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66
       67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86
       87 88 89 90 91 92 93 94 95 96 97 98 99 100 101 102
         } # List of all queue detectors – Must be set to 1 sec sampling rate
     { 74 102 } # Demand Detectors
    }
  }              # End of a controller
}        # End of All controllers
```

Appendix C

Sample Input File for Linked Algorithm

```
07:00:00        # Becomes in effect at 7:00AM
{
  2      #ControllerID
  14     #ControllerType (14 = Linked)
  60   #updateStepSize_ , step size for calculating parameters (sec)
  12     #SignalType (12 = ramp meter)
  1      # NumEgresses (number of outgoing links)
   {
    2  # number of ramps in region
    2  # nApproaches (number of signals)
      {
       #signalID RampID
        0           0
        1           0
        2           1
        3           1
      }

3 #number of mainline lanes
4  #number of detector stations
{
  33 34 35 #DetectorIDs, detector station 1
  27 28 29 #DetectorIDs, detector station 2
  36 37 38 #DetectorIDs, detector station 3
  30 31 32 #DetectorIDs, detector station 4
}

{
  10 #Minimum Cycle (sec)
  4 # green (sec)
  1 # start amber (sec)
  2 # stop amber (sec)
  30 #startup green (sec)
  2 #startup stop amber (sec)
  15 #startup red (sec)
}
```

```
5  # number of queue detectors per lane on first ramp
3  # number of queue detectors per lane on second ramp
{ #detectorID   Weight
  10 9
  12 11
  14 15
  16 30
  18 50
  11 9
  13 11
  15 15
  17 30
  19 50

  21 10
  23 30
  25 50
  22 10
  24 30
  26 50
}

{
100 200 300 9998 9999 # metering rate adjustment thresholds for queue control Ramp 1
60 120 180 9998 9999 # metering rate adjustment thresholds for queue control Ramp 2
}

{
20 40 60  # occupancy level thresholds for queue control Ramp 1
20 40 60 # occupancy level thresholds for queue control Ramp 2
}
} #End of a control region
} #End of all controllers
```

Appendix D

Connecting MATLAB and MITSIMLab

In order to implement the linked ramp metering algorithm, code written in MATLAB was used. Use of the MATLAB Engine is necessary to link MITSIMLab and MATLAB so that MITSIMLab can call MATLAB, and to pass data between the two programs.

The MATLAB Engine should be installed when MATLAB is installed. In order to use the functions in the MATLAB Engine, its header file must be included, by adding the following line to your source code:

```
#include <engine.h>
```

Also, in your class definition, you must define a pointer to a MATLAB Engine object:

```
Engine* pMatlab_;
```

Obviously, you can substitute any object name you wish for pMatlab_. The underscore (_) character is a general convention used for class members.

In order to run MATLAB code, the following steps must be followed:

1. Open the MATLAB Engine
2. Convert any data you want to send into MATLAB into a MATLAB array
3. Send the data to MATLAB
4. Execute the MATLAB code

5. Send the MATLAB output to MITSIMLab
6. Convert the output to a C / C++ array (and if you wish, to a vector, or any other data structure)
7. Close the MATLAB Engine

Opening the MATLAB Engine

The first step is to open the MATLAB engine. When you initialize the class (in its constructor, or elsewhere), the engine object pointer should be set to a null pointer:

```
pMatlab_ = NULL;
```

The first time the engine is needed, a test should be run to see if the engine points to null. If it does, the engOpen function should be used to open the engine:

```
if (pMatlab_ == NULL) {
    if (!(pMatlab_ = engOpen(NULL))) {
            cerr << "Error:: Can not open MATLAB";
            theException->exit(1);
    }
    cout << "Launch Matlab Engine." << endl;
}
```

Since pMatlab_ was previously set to null, it will remain null until the engine is open. The engOpen function opens the engine. If its argument is null, it will just open the engine, otherwise you can pass it a string, and MATLAB will execute the string as a command. If for some reason the engine fails to load, engOpen will return false; the above test will display the error message and throw an exception. This is important, since the rest of the program will depend on the MATLAB engine. If this code is part of a loop, if the engine is already open, pMatlab will point to something other than null, and the program will continue without needing to reopen the engine. When you load the engine, there will be a slight delay, and the MATLAB welcome screen will briefly pop up. However, once the engine is loaded, MATLAB will run reasonably quickly.

Converting input into a MATLAB array

The MATLAB engine defines a class mxArray, which contains data in a form understandable to MATLAB, in order to pass data to and from MATLAB. A pointer to an mxArray object must be defined:

```
mxArray occArray;
```

Next, the array must be initialized:

```
occArray=mxCreateDoubleMatrix(1,nDetectors_,mxREAL);
```

The mxCreateDoubleMatrix creates a MATLAB array of type double (double precision floating point). The first two parameters specify the size of the matrix. Since we are sending a row vector, the array has 1 row and nDetectors_ columns. The parameter mxREAL tells the engine that this array contains the real (as opposed to imaginary) component. Since most MITSIMLab work involves only real numbers, mxREAL is almost always used.

In order to convert the data into a MATLAB array, the data must first be stored as a standard C / C++ array:

```
double tempOcc[nDetectors_];
```

If the input data is in another data structure (such as a vector) it must first be converted into a standard array.

Finally, use the memcpy function to copy the actual data:

```
memcpy(mxGetPr(occArray),tempOcc,sizeof(tempOcc));
```

The memcpy function works by copying the data pointed to by tempOcc to the memory pointed to by mxGetPr(occArray), copying sizeof(tempOcc) bytes. tempOcc obviously points to the first element of the tempOcc array. The mxGetPr function returns a pointer to the first element of occArray.

Sending the data to MATLAB

Once the data is converted into a MATLAB array, it is easy to use the engPutVariable function to send the data to MATLAB:

```
engPutVariable(pMatlab_,"occInput",occArray);
```

This sends the data in occArray (created in the previous step) and creates a MATLAB matrix called occInput. The matrix name that MATLAB uses must be sent as a string. The MATLAB code that is run in the next step has access to occInput.

Running the MATLAB code

The function engEvalString is used to run MATLAB code:

```
if (engEvalString(pMatlab_,"linked")) {
        cerr << "Error:: Calling Matlab." << endl;
        engClose(pMatlab_);
        theException->exit (1);
}
```

MATLAB will run either an internal command called linked (if it existed), or run a file called linked.m. The engEvalString function allows you to run anything that can be run from the standard MATLAB command line; in this case it is used to execute an m-file. The only difference is that when run through the engine, you cannot see any MATLAB screen display, since it is run in the background. engEvalSring returns 0 if the code runs without error. The above test is used to find whether or not it causes an error, anf if It does, it will display an error message, close the engine, and throw an exception.

Sending MATLAB output to MITSIMLab

The MATLAB output must be received as an mxArray:

```
MxArray* matlabCycle_;
```

The engGetVariable function is used to receive the data:

```
matlabCycle_=engGetVariable(pMatlab_,"cycle");
```

This function gets a MATLAB matrix called cycle, and sends it to MITSIMLab, in a MATLAB array called matlabCycle_.

Converting output to a C / C++ array

The MATLAB data must now be converted into a format understandable by your program.

```
double* cycles;
cycles=mxGetPr(matlabCycle_);
```

The mxGetPr function takes the data pointed to by matlabCycle_, and copies it to the standard array pointed to by cycles. cycles is a standard C / C++ array, it can now be converted into a vector or any other data structure, or used directly as an array.

Closing the MATLAB Engine

To close the MATLAB engine, simply use the engClose function, either in the destructor, or elsewhere:

```
engClose(pMatlab_);
```

Compiling and linking the code

In order to compile and link the code, there are several libraries that must be attached. All of these should be installed with MATLAB. In the makefile, the –L switch is used to specify a search path for libraries:

-L/opt/matlabR13/extern/lib/glnx86

glnx86 should be changed to whatever architecture is being used.

To link the libraries, the –l switch is used in the makefile:

-lm -leng –lmx

These switches link the libraries libm.so, libeng.so, and libmx.so.

Finally, before you run make, and whenever you execute MITSIMLab, the LD_LIBRARY_PATH environment variable must be set:

setenv LD_LIBRARY_PATH /opt/matlabR13/extern/lib/glnx86:$LD_LIBRARY_PATH

Again, glnx86 should be substituted with whatever architecture is being used.

It is important to set this environment variable before you run make, and also before MITSIMLab is run. Also, it is important to set the variable before pvm is started. In the case that the program will not run no matter what you do, then halt pvm, set the environment variable, and then start pvm again, and it should work.

Bibliography

Addleman, S. (1962) Orthogonal Main-Effect Plans for Asymmetrical Factorial Experiments. *Technometrics*. Vol. 4, No. 4, Rochester, NY, pp. 21-54.

Ahmed, K. I. (1999) *Modeling Drivers' Acceleration and Lane Changing Behaviors*. PhD Thesis, Department of Civil and Environmental Engineering, Massachusetts Institute of Technology, Cambridge, MA.

Ben-Akiva, M. E., Koutsopoulos, H. N., Mishalani, R. G., and Yang, Q. (1997) Simulation Laboratory for Evaluating Dynamic Traffic Management Systems. *ASCE Journal of Transportation Engineering*, Vol. 123, No. 4, pp. 283-289.

Blosseville, J. M. (1985) *IRT Repor*t, INRETS, Arcueil, France.

Box, M., (1965) A New Method of Constrained Optimization and a Comparison With Other Methods, *Computer Journal*, Vol. 8, pp.42--52

Cambridge Systematics, Inc., SRF Consulting Group, Inc., and N.K. Fredrichs Consulting, Inc. (2000). *Twin Cities Ramp Meter Evaluation*. Oakland, CA.

Cascetta, E., Inaudi, D., and Marquis, G. (1993) Dynamic Estimators of Origin-Destination Matrices Using Traffic Counts. *Transportation Science* Vol. 27, pp. 363-373.

Darda, D. (2002) *Joint Calibration of a Microscopic Traffic Simulator and Estimation of Origin-Destination Flows.* MST Thesis, Department of Civil and Environmental Engineering, Massachusetts Institute of Technology, Cambridge, MA.

Gordon, R. L. (1996) Algorithm for Conrolling Spillback from Ramp Meters. *Transportation Research Record* 1554, pp. 162-171.

Gould, C., Munro, P., Hardman, E. (2002) M3 / M27 Ramp Metering Pilot Scheme (RMPS) – Implementation And Assessment. *Road Transport Information and Control* Conference Publication 486, London, UK.

Hadj Salem, H., Blosseville, J. M., Papageorgiou, M. (1988) ALINEA. *INRETS Report* No. 80, INRETS, Aruceil, France.

Hasan, M. (1999) *Evaluation of Ramp Control Algorithms Using a Microscopic Traffic Simulation Laboratory, MITSIM*. MST Thesis, Department of Civil and Environmental Engineering, Massachusetts Institute of Technology, Cambridge, MA.

Hourdakis, J., and Michalopoulos, P. G. (2002) Evaluation of Ramp Control Effectiveness in Two Twin Cities Freeways. *Transportation Research Board 81st Annual Meeting*, Washington, DC.

Hunter, C. D. (2000). *Intelligent Transportation Systems*. Course Syllabus. University of Rhode Island, Kingston RI.

Jacobson, L., Henry, K., and Mahyar, O. (1989) Real-time Metering Algorithm for Centralized Control. *Transportation Research Record* 1232, Transportation Research Board, National Research Council, Washington, DC, pp. 17-26.

Koble, H. M., Adams, T. A., Samant, V. S. (1980) Control Strategies in Response to Freeway Incidents. Report No. FHWA/RD-80/005, Federal Highway Administration, Washington, DC.

Kwon, E., Nanduri, S., Lau, R., and Aswegan, J. (2001) Comparative Analysis of Operational Algorithms for Coordinated Ramp Metering. *Transportation Research Record* 1748, pp. 144-152.

Levinson, D., Zhang, L., Das, S., and Sheikh, A. (2002) Ramp Meters on Trial: Evidence from the Twin Cities Ramp Meters Shut-off. *Transportation Research Board 81st Annual Meeting*, Washington, DC.

Lipp, L. E., Corcoran, L. J., Hickman, G. A. (1991) Benefits of Central Computer Control for Denver Ramp-Metering System. *Transportation Research Record* 1320, pp. 3-6.

Masher, D. P., Ross, D. W., Wong, P. J., Tuan, P. L., Zeidler, Peracek, S. (1975) Guidelines for Design and Operating of Ramp Control Systems. *Stanford Research Institute Report* NCHRP 3-22, SRI Project 3340, SRI, Menid Park, CA.

Mathworks, Calling MATLAB from C and Fortran Programs (External Interfaces/API). *MATLAB*. Online Documentation. http://www.mathworks.com/access/helpdesk/help/ techdoc/matlab_external/ch06eng3.shtml.

McKenna, P. (2003) *Design of the Linked Controller*. Personal e-mail.

Oh, H., and Sisiopiku, V. P. (2001) A Modified ALINEA Ramp Metering Model. *Transportation Research Board 80th Annual Meeting* 01-3096, National Research Council, Washington, DC.

Paesani, G., Kerr, J., Pervich, P., Khosravi, F. (1997) System Wide Adaptive Ramp Metering in Southern California. *ITS America Annual Meeting*, Washington, DC.

Papageorgiou, M., Blosseville, J. M., and Hadj Salem, H. (1990) Modeling and Real-time Control of Traffic Flow on the Southern Part of Boulevard Peripherique in Paris: Part II: Coordinated On-ramp Metering. *Transportation Researc*h. Vol. 24A, No. 5, pp. 361-370.

Papageorgiou, M., Hadj Salem, H., and Blosseville, J. M. (1991) ALINEA: A Local Feedback Control Law for On-Ramp Metering. *Transportation Research Record* 1320, pp. 58-64.

Papageorgiou, M., Hadj-Salem, H., and Midddelham, F. (1997) ALINEA Local Ramp Metering: Summary of Field Results. *Transportation Research Boar*d, *76 th Annual Meeting*, Washington, DC.

Roess, R. P., McShane, W. R., Prassas, E. S. (1998) *Traffic Engineering: Second Edition*, Prentice Hall, Upper Saddle River, NJ.

Taylor, C., Meldrum, D., and Jacobson, L. (1998) Fuzzy Ramp Metering Design Overview and Simulation Results. *Transportation Research Record* 1634, pp. 10-18.

Taylor, C. J., McKenna, P. G., Young, P. C., Chotai, A., Ben-Akiva, M., Toledo, T., and Scariza, J. R. (2002) *M3 / M27 Ramp Metering Pilot Scheme: Co-ordination of Ramp Metering Sites*. UK Highway Agency Contract No. 3/350. Deliverable 2, Lancaster, UK.

Taylor, C. J., Young, P. C., Chotai, A., Whittaker, J. (1998) Nonminimal State Space Approach to Multivariable Ramp Metering Control of Motorway Bottlenecks. *IEEE Proceedings: Control Theory and Applications*. Vol. 145, No. 5, pp. 568-574.

WSDOT's New Ramp Metering Algorithms Are Significantly Better Than Before (2000). *The Urban Transportation Monitor*, Vol. 14, No. 23, p. 6.

Smaragdis, E., and Papageorgiou, M. (2003) A Series of New Local Ramp Metering Strategies. *Transportation Research Board, 82nd Annual Meeting*, Washington, DC.

Yang, Q. (1997) *A Simulation Laboratory for Evaluation of Dynamic Traffic Management Systems*. PhD Thesis, Department of Civil and Environmental Engineering, Massachusetts Institute of Technology, Cambridge, MA.

Yang, Q. and Koutsopoulos, H. N. (1996) A Microscopicic Traffic Simulator for Evaluation of Dynamic Traffic Management Systems. *Transportation Research* 4c, pp. 113-129.

Zhang, L., and Levinson, D. (2003) Balancing Efficiencey and Equity of Ramp Meters. *Transportation Research Board 82nd Annual Meeting*, Washington, DC.

Zhang, M., Kim, T., Nie, X., Jin, W., Chu, L., Recker, W. (2001) *Evaluation of On-ramp Control Algorithms*. California PATH Research Report UCB-ITS-PRR-2001-36, Berkeley, CA.