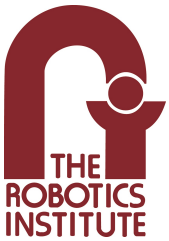
Traffic Control with Connected Vehicle Routes in SURTRAC

Allen Hawkes

May 7, 2016



Robotics Institute

Carnegie Mellon University

Pittsburgh, Pennsylvania 15213

Technical Report CMU-RI-TR-16-20

Thesis Committee

Prof. Stephen Smith

Prof. Aaron Steinfeld

Hsu-Chieh Hu

Contents

Abstract

Connected vehicles are entering the market, but will not form the vast majority of all vehicles for a few decades. Traffic control that relies on information from connected vehicles (CVs) is thus challenging because it must cope with the current low market penetration rate of CVs. I devise a method to extend Surtrac, an existing distributed traffic controller, to incorporate CV information in the form of routes. Vehicle routes are converted to arrival times for all waypoint intersections, and then incorporated into Surtrac's optimization. This method, when tested on a 36-intersection 1-way grid, improves vehicle delay by 25% for CVs at low penetration rates and by 37% when all vehicles are connected. The method also benefits non-CVs, even when nearly all other vehicles are connected. Results are also given for a 23-intersection area of Pittsburgh, where a comparison is made between this method, Surtrac, and existing well-tuned fixed timings. The extension of Surtrac to use CV routes improves traffic at all penetration rates, and provides motivation for early adoption of a route-sending system.

1. Introduction

1.1. Motivation

Traffic control is highly important simply due to the vast number of people it affects daily. While traffic controllers themselves are by no means a new technology, research in traffic control is paramount because of the potential time and resource savings it can bring about. Commuters in the largest 471 US urban areas averaged 42 hours of delay per person in 2014 alone [21], highlighting the widespread need for improvement in traffic control. Many relatively new technologies do exist, like mobile phones or Vehicle-to-Infrastructure communication, that have yet to be fully integrated with existing traffic networks. These technologies enable improved vehicle detection and prediction, motivating research that best leverages the new data to improve traffic control.

1.2. Previous Work

A basic challenge for traffic control systems is to manage both local (i.e. by-the-second) demand of vehicles and network-level flows of vehicles. Some methods focus on the higher-frequency information from local demands at each intersection

by rapidly updating vehicle flow models and making adjustments to traffic signal plans every second [26, 2]. Others focus on lower frequency flows for traffic control by either optimizing a network offline, like SYNCHRO [15], or planning online with a rolling optimization horizon, like ALLONE-S [23] and OPAC [7]. However, these network-minded methods often suffer from an inability to be solved in real-time for realistic planning horizons [22]. Similarly, reinforcement learning methods for traffic control [2, 25] are not able to react quickly enough to real-world conditions at the local level to be practical. These approaches that seek to coordinate a traffic network tend to be centralized in nature and thus struggle with scalability issues [8], leading to decentralized methods that seek to use information to extend an otherwise myopic view of a traffic network [16].

1.2.1. Surtrac

One such decentralized method is a schedule-driven traffic control approach [32], which has since been developed into the Scalable Urban Traffic Control (Surtrac) system, currently deployed at 50 intersections throughout the city of Pittsburgh [27]. Surtrac uses greedy dynamic programming to minimize the delay of all local vehicles at an intersection, and obtains an extended optimization horizon by passing messages between intersections for vehicle outflows. At its core, Surtrac is a locally optimal system that is able to maintain a level of coordination with its nearest neighbors, which allows for a slightly more global solution [27]. This intersection-to-intersection coordination of traffic control is crucial for preventing gridlock, especially for tightly coupled intersections that are very close and have high volumes of vehicles between them [3]. However, Surtrac's coordination is

designed to only propagate vehicles leaving one intersection for another, and thus is limited in its long-range view of the network [27]. Further, it relies on turning proportions that estimate the trajectories of approaching flows. This work seeks to extend the planning horizon of Surtrac through additional sensing that originates from the vehicles themselves. Current traffic control strategies rely mainly on infrastructure-based vehicle detection, namely inductive loops in the ground, cameras, or radar arrays. Actuated control typically lengthens or shortens phases depending on the presence of vehicles, and requires manual tuning to adjust for a model of network flows. Adaptive control systems, like Surtrac, automate this process by estimating a portion of the network state from vehicle detections and adjusting phases accordingly.

1.2.2. CV Traffic Control

As connected vehicles (CVs), autonomous vehicles, and even drivers using mobile navigation applications become more prevalent, additional data can be made available to the infrastructure, through Vehicle to Infrastructure (V2I) communication or otherwise. Some traffic control strategies have been designed around CV information to obtain higher accuracy in local vehicle detection, where CV speed and position are used to build a 20 second prediction horizon in a decentralized adaptive setting [24]. Other work involves CV position, heading, and speed within a microscopic simulation model to predict future demand at an intersection in a 15 second rolling horizon, but is limited in real-world application due to its computational complexity [9]. Additionally, the type of vehicle, such as an emergency response vehicle, can be used for prioritization within a traffic control scheme, al-

though this often degrades performance for vehicles not receiving special treatment [13].

In current connected vehicle research, the market penetration rate of connected vehicles is an overarching parameter that determines the feasibility of a method based on the percent of participants. Because a US federal mandate is in place requiring all newly designed vehicles to contain V2I capabilities, some work considers the possibilities for the end-state of 100% market penetration, and demonstrate significant reductions in vehicle delay [5, 17]. Some works take this to an extreme, eliminating the infrastructure-based control and instead using autonomous vehicle cooperation to share intersection space [28, 17, 4, 11]. This 100% state will not exist for about another 25 years [9], so there is much interest in developing conventional systems that improve traffic conditions throughout a wide range of market penetration. Recent methods [12, 18, 6] require a minimum 30-40% of all vehicles as CV in order to demonstrate a benefit over a well-tuned simpler actuated system, and perform best at 100% CV penetration, where delays have been shown to decrease by 16% for a particular volume distribution. Currently, 9% of all vehicles in the United States are have some connected vehicle capabilities, and not all are able to contribute to the congestion-reduction methods described here [1]. Thus, a large gap looms between the current concentration of CVs within the US today, and the minimum 30%-40% required for these methods.

1.3. Problem Statement

We seek to create a traffic control method that best utilizes connected vehicle information at all penetration rates to reduce traffic congestion. Specifically, we

intend to demonstrate that a vehicle's route and position, when communicated to a network, can be used to decrease that vehicle's own delay, as well as the delay of other vehicles that do not send routes. Vehicle routes, which are a readily available form of data due to widespread usage of navigation devices and applications, can be communicated in a distributed manner through a custom V2I message or via a cloud-based framework, for example. Working within Surtrac's schedule-based control framework, routes are converted to predicted arrival times at an intersection, and then incorporated into each intersection's dynamic programming optimization. Surtrac's locally optimal solution, which has been shown in real-world deployments to improve delays, stops, and travel times by 41%, 31%, 26% respectively [27] provides an effective starting point for traffic control in urban settings.

This work builds upon Surtrac's infrastructure-based sensing by utilizing route choice of any vehicle, whether connected, autonomous, or otherwise, as a practical predictor of traffic at an intersection. We first examine varying features and heuristics to build a reasonably accurate model of estimated arrival time. Using these predictions, we analyze the system's performance on a simple 36-intersection test network at penetration rates from 0-100%. The delay of CVs, non-CVs, and all vehicles together is studied for varying penetration rates and arrival time heuristics, along with the relationship between the different types of vehicles. We then apply the modified Surtrac method to a realistic 23-intersection simulation, where standard Surtrac has already been shown to significantly improve delays in comparison to the existing control.

This work makes the following contributions:

1. Develop a method to utilize vehicle routes in a schedule-based traffic control framework.

2. Derive simple heuristics for determining vehicle weight and arrival time within a CV route.

3. Demonstrate lower delay, stops, and travel times than Surtrac, by simulating a realistic traffic network.

2. Overview of Surtrac

To understand the details of this new method, a brief explanation of Surtrac is first provided, specifically with regards to its scheduling algorithm. Surtrac runs in distributed fashion and takes as its primary input a sequence of clusters. Clusters are a basic representation of a vehicle or group of vehicles, represented as $c = (t_{arr}, t_{arr,end}, |c|)$. t_{arr} is the estimated arrival time of a cluster to an intersection, and $t_{arr,end}$ is the end of arrival of the cluster. $|c|$ is the number of vehicles, or weight, of a cluster. Vehicles that are spaced apart within a threshold are combined to form a single larger cluster. Likewise, vehicles that are expected to join a queue are clustered together. A sequence of clusters in time on a road is denoted as a road flow, and all non-conflicting road flows that can pass through an intersection simultaneously are together called an inflow. The scheduler manages the space of the intersection as a resource that must be allocated separately in time to all competing inflows by minimizing the delay caused by the duration of a series of phases. An example of two competing inflows and a resulting schedule of phases for each inflow is shown in Fig. 2.1.

The schedule is defined as a sequence of clusters assigned to phases in which they will pass through the intersection, also called a control flow. In Fig. 2.1, the gap between the beginning of a green phase and a stopped cluster is the start-up

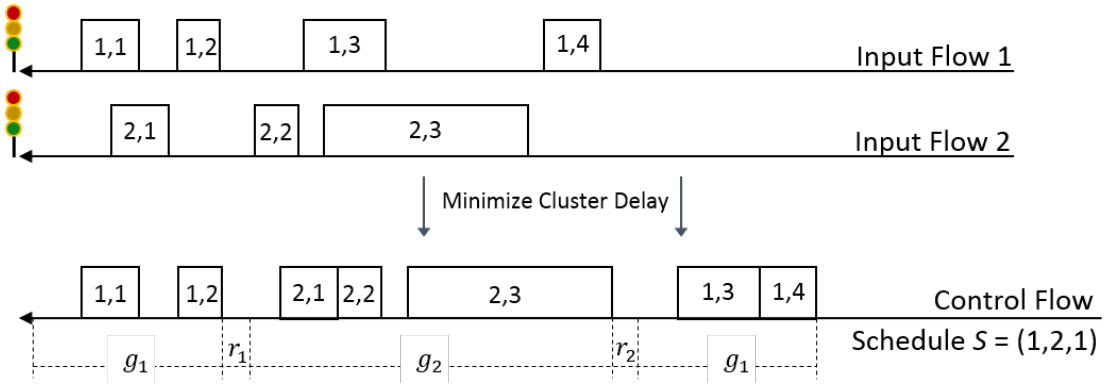


Figure 2.1.: Vehicles on roads approaching an intersection are represented as flows of clusters. Once scheduled, the clusters form a control flow, controlled by schedule S.

lost time (*slt*) [26]. Surtrac takes into account *slt*, minimum and maximum green times, red and amber clearances, and other constraints associated with standard traffic control to generate the control flow via a greedy dynamic programming algorithm [32].

Currently, clusters are generated from two sources: advance detection and outflow communication. Advance detection refers to vehicle counts that originate along the input roads around 100-300 meters upstream of an intersection, typically from inductive loops, cameras, or radar sensors. Outflow communication refers to the messages passed between nearest-neighbor Surtrac agents, which contain the clusters between neighbors from the upstream intersection's scheduled control flow. In practice, these two sources provide the same information, and at a horizon of no more than one intersection away from the intersection receiving information. While Surtrac builds turning proportions that allow for a probabilistic outflow to be communicated prior to vehicles making turning decisions, it was determined that a network of Surtrac controllers performs best when only real (detection-produced) outflows are communicated [27]. This lack of certainty in

turning proportions is the motivation for using known vehicle routes. While turning proportions may be accurate over the long term, they do not provide enough real-time information for Surtrac to schedule greedily and incorporate effectively.

3. Route Integration

3.1. Adding Clusters before Optimization

Since Surtrac operates on clusters, vehicle routes must also be converted to clusters and then incorporated into the existing representation of cluster arrivals. An intersection's input flows, as in Fig. 2.1, can be modified to account for future vehicles by inserting clusters into the existing time-ordered cluster series. Surtrac's optimization is then carried out after this cluster adding process. Most route information that Surtrac will receive as routes will be farther than one intersection away, as this is most useful, so these route-based clusters typically appear after the existing input flow clusters. In this case, the route-clusters are simply appended to the cluster flows. However, there may be cases in which route-based clusters precede the existing clusters, in which case they must be inserted, as shown in Fig. 3.1.

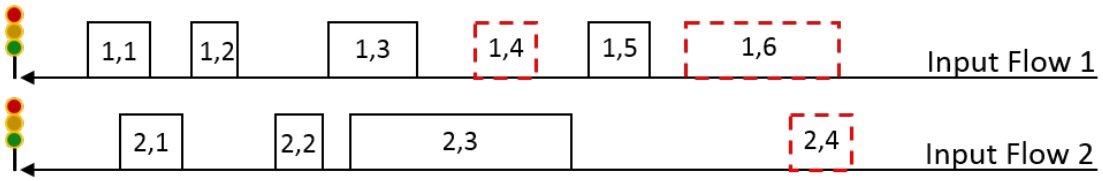


Figure 3.1.: Clusters (1, 6) and (2, 4) have been appended, and cluster (1, 4) has been inserted.

3.2. Converting Routes to Clusters

A cluster is appended or inserted according to its estimated arrival time (t_{arr}) at an intersection. Thus, the primary work in converting routes to clusters for Surtrac is to determine a t_{arr} that corresponds to a waypoint intersection through which a vehicle will pass. In this way, a route is defined as a set of clusters, each representing the same vehicle, but containing a unique t_{arr} for all Surtrac intersections on a vehicle's route. This process is outlined in Fig. 3.2, where a vehicle's route is converted to two unique clusters that have been inserted into the input flows of their respective intersections.

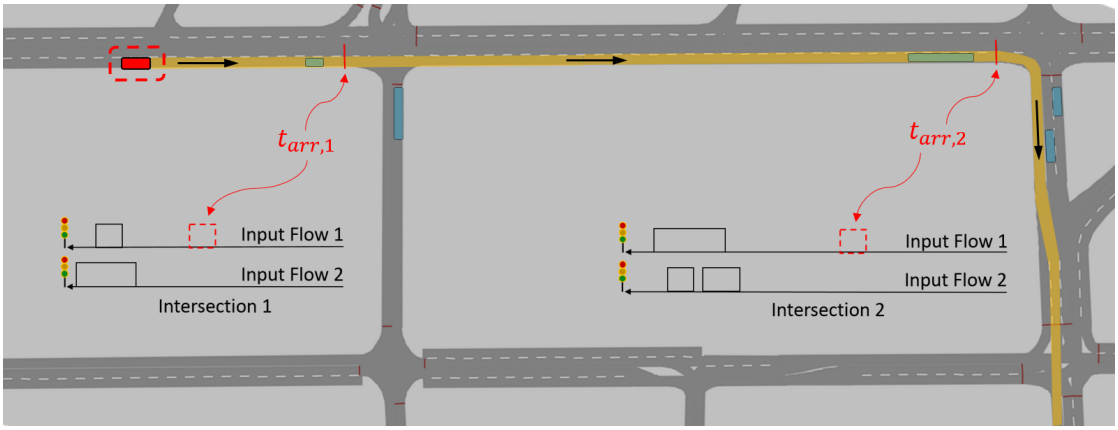


Figure 3.2.: Estimated arrival times at 2 intersections on a route lead to modified schedules. The red bars on the roads by each t_{arr} here indicate the stop-bar of an intersection. When a vehicle reaches a stop-bar, it will either pass through if the signal is green (and sometimes yellow), or come to a stop if it is red. Thus, t_{arr} is the point in time at which a signal controller commands the movement of a vehicle. The inset diagrams represent schedules of the two intersections. Each schedule contains clusters of competing flows, where each red dashed square indicates a cluster added from the highlighted vehicle's route. Intuitively, a cluster is predicted to arrive later for intersections that are farther along a route.

By contrast to conventional detection where a vehicle corresponds to a single piece of data, this method maintains multiple representations of the same vehicle

spread throughout a network. In this way, the route sending process can be thought of as a series of advance detectors, one for each intersection receiving CV clusters, that move continuously with the vehicle. The purpose these "detectors" serve is to continuously update the t_{arr} of a particular cluster at multiple intersections at once.

This does not conflict with Surtrac's passing of clusters between intersections, because only vehicles detected locally though conventional methods are shared between intersections. However, if a route is sent to the next intersection after a vehicle has passed over an advance detector, this will double-count a vehicle's cluster in the next intersection's schedule. Then, one cluster in the schedule will be from local detection, and the other from route-sending. It may actually be desirable to utilize double-counting of route-sending vehicles, because the information they send is more accurate than local detection in both real-time location and t_{arr} estimates. Sending routes to the closest intersection is evaluated in sec. 7.3.3.1. Surtrac has no memory or storage of past inserted clusters, and optimizes whatever cluster information is passed to it over a specific period of time. Thus, clusters calculated from vehicle routes are transmitted continually, or once per simulation second for our experiments as described in chapter 6.

Adding these vehicle clusters creates a more accurate representation of vehicular demand of intersection space. The existing representation within Surtrac is fairly accurate, but only contains information regarding vehicles in between neighboring Surtrac intersections. While conceptually, Surtrac is capable of passing flows through more than one intersection, the current implementation does not allow for this. Route-based information can increase the scheduling horizon by adding clusters which arrive further in the future than only 1 intersection away, which

allows for a more globally optimal traffic light schedule.

3.3. Communication of Clusters

The method by which routes are communicated throughout a network is treated as a centralized system that passes the appropriate messages to different Surtrac agents as necessary, as in chapter 6. This is done to maintain a focus on investigating the resulting behavior of incorporating routes to existing optimization.

It is also possible to apply a distributed message-passing scheme for propagating vehicle routes through a network in a variety of ways, and with little extra work. One scheme would be to have the closest intersection receive the entirety of a vehicle's route, including all the waypoint intersections, and then use a centralized server to distribute messages to all other intersections. Another method would be to use distributed message passing, without any centralized server, where messages are routed to their destination by way of other intersections. Many other general approaches exist, with determining factors such as number of CVs and number of intersections, so we leave the investigation of a distributed route-sending system for future work.

4. Arrival Time Heuristics

At the core of using routes for scheduling is estimating arrival times to intersections along a vehicle's route. Travel time estimation methods have been heavily researched, ranging from use of support vector regression on highway data [31] to real-time prediction with GPS-enabled devices [30, 14]. While it is useful to compare the performance of the arrival time estimation method derived here, the focus of this work is not to develop an advanced method for determining arrival (t_{arr}) or travel times. Rather, we seek to demonstrate a simple method for estimating t_{arr} that fits within and makes use of the Surtrac framework, and to show that the final heuristic is sufficient for allowing routes to be used by Surtrac. We leave fine-tuning and further development of highly accurate t_{arr} values to future work.

A vehicle's instantaneous location and the current network state are both primary informers of arrival time. For a single intersection, we define the expected arrival time of a cluster as in 4.1.

$$t_{arr,n} = t_{arr,n-1} + t_{lightDelay,n-1} + t_{freeFlow,n} + t_{queueDelay,n} \tag{4.1}$$

$t_{arr,n}$ is the sum of a vehicle's arrival and passage through the previous in-

tersection $(n-1)$, plus the time required to reach the stop-bar of the next intersection. Here, time to pass through an intersection from the stop-bar is defined as $t_{lightDelay,n-1}$, which is simply the delay incurred waiting at a stoplight. $t_{lightDelay,n-1}$ is a function of $t_{arr,n-1}$, meaning that a cluster's expected dwell time at an intersection can be found by accessing the schedule at $t_{arr,n-1}$. For a vehicle that has just entered a network, $t_{lightDelay,n-1} = 0$ because there is no previous stoplight. The time to reach the next stop-bar is the sum of edge travel at free-flow speed, $t_{freeFlow,n}$, and the delay incurred by vehicles already at the stop-bar, $t_{queueDelay,n}$. $t_{queueDelay,n}$ is a measure of start-up lost time (sec. 4.2).

Arrival time, as defined above, simplifies the schedule look-up process for $t_{lightDelay,n}$ and is the primary variable used in mapping routes to clusters. When the recursive component of the formulation, $t_{arr,n-1}$, is removed, we arrive at a definition for the through-intersection time.

$$t_{throughInt,n} = t_{freeflow,n} + t_{queueDelay,n} + t_{lightDelay,n} \tag{4.2}$$

4.2 defines the time to travel through an intersection and its upstream edge. A vehicle's total travel time through a signalized network is the sum of all individual through-intersection travel times, in 4.3.

$$t_{routeTotal} = \sum_{n-1}^{N_{int}} t_{throughInt,n} \tag{4.3}$$

Once t_{arr} is estimated, a value is assigned for the cluster end ($t_{arr,end}$) based on

the number of vehicles in the cluster times a constant. For simplicity, we assume the duration of a single-vehicle cluster to be 1 second. Each component of t_{arr} (and likewise $t_{throughInt}$) is derived below.

4.1. Edge Travel Times

The expected travel time on an edge, without delay, is simply the edge length divided by free-flow speed.

$$t_{freeFlow} = \frac{l_{edge}}{v_{freeFlow}} \tag{4.4}$$

This time is denoted as an edge's free-flow time, $t_{freeFlow}$. For each edge, or set of roads between intersections, an expected $t_{freeFlow}$ is maintained. To initialize this value, a set of travel times of vehicles on an edge is first collected. We use k-means clustering with k=2 to separate post-processed travel times into one of two classes: free-flow, and non-free-flow. Non-free-flow times account for vehicles that either experienced significant delays or stopped at the intersection. The free-flow cluster has a lower mean travel time, so we pick the smaller of the two clusters as our initialized expected travel time. While one could have simply selected an edge's speed limit as an initializing value, the actual edge travel time can deviate significantly from this value. Narrow lanes and a large pedestrian presence, for example, may cause a free-flow speed to be lower than would be expected from a speed limit.

The post-processed free-flow travel times are also updated in real-time through

velocity data provided by the route-sending CVs. Each vehicle communicates its instantaneous speed for the duration of driving on an edge, and its highest speed on the edge is saved as a single free-flow velocity data point. A rolling average updates an edge's free-flow time from the connected vehicle free-flow times, and uses a window of 10 – 100 data points, depending on the approximate market penetration of connected vehicles. This sliding window range is kept relatively low, in comparison to the large time frame over which traffic is analyzed, to provide an estimate that is closer to real-time conditions.

4.2. Queue Delay

Start-up lost time (slt) and saturation flow rate (sfr) are the primary factors in determining queue delay, which we define as the delay incurred by sequential acceleration of cars from a stopped state. Queue delay is approximated by applying the sfr to the cars preceding the cluster of interest within a queue, and adding slt, as in 4.5.

$$t_{queueDelay} = slt + \frac{N_{inQueue}}{sfr} \tag{4.5}$$

A typical value for both $\frac{1}{sfr}$ and slt is 2 seconds [32]. We approximate $N_{inQueue}$ with the number of vehicles that precede t_{arr} within the same phase. This value overestimates $N_{inQueue}$ because not all vehicles in front of a cluster are guaranteed to be queued, since some may in fact be free-flowing when they arrive at the stop-bar. To account for this, we simply use a lower $\frac{1}{sfr}$ of 1 second. The preceding

clusters are accessed via the scheduled control flow at the time $t_{arr,queue}$, an estimate of the time of arrival at the back of the queue, as in 4.6.

$$t_{arr,queue} = t_{arr,n-1} + t_{lightDelay,n-1} + t_{freeFlow,n} \tag{4.6}$$

Note that 4.6 is the same as the definition for $t_{arr,n}$, but without $t_{queueDelay,n}$. This approximation ignores the delay from deceleration to the rear of the queue, which would cause a predicted arrival to be earlier than the actual. However, it also ignores the shorter distance to the rear of the queue, for a later-than-actual arrival predition. We make the assumption that these two simplifications approximately cancel each other out.

4.3. Traffic Light Delay

While the delay caused by a traffic light can result in queuing and other delays, we define traffic light delay as the expected amount of time a vehicle will wait for green. Specifically, the expected wait time for a vehicle is the time between when a vehicle arrives at an intersection and when it is allowed to pass through. If the phase controlling vehicle movement is already green when the vehicle arrives, then $t_{lightDelay} = 0$. If the phase is not green, the Surtrac-scheduled flow can be accessed to determine the approximate next green time. By passing messages from each Surtrac agent to the centralized route-cluster manager, the scheduled control flows (Fig. 2.1) at each intersection are actively maintained in a lookup table. Note that while phase start and end times are not directly available from the scheduled

control flow, an approximation can be made by taking the first and last elements of a scheduled sequence of clusters.

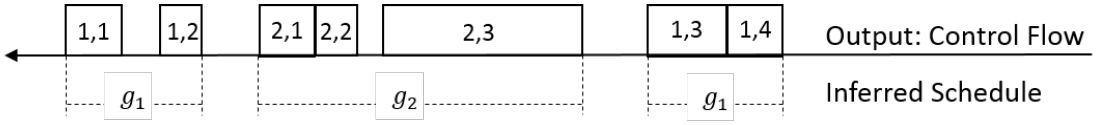


Figure 4.1.: Assigned schedule phases (below) as determined by Surtrac-output control flow.

The method above in Fig. 4.1, which uses raw output from Surtrac, lacks an understanding of phase behavior between clusters of different phases. This space is undefined and is provided to a Surtrac as a flexible space in which either the end of the previous phase can be extended or the start of the next can begin, depending on the demand of vehicles at decision time. In practice, however, Surtrac is designed to provide extensions to phases [27] in a way that causes it to use as much free space as possible for the active phase. Thus, the actual phase-switch time is approximately located at the first scheduled cluster of each next phase, as shown in Fig. 4.2.

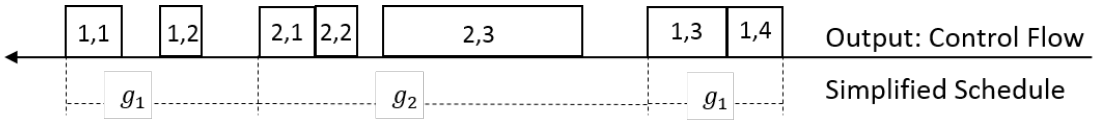


Figure 4.2.: Red and amber times are eliminated to simplify the schedule representation.

With a complete representation of the schedule, a cluster's expected waiting time at an intersection can be easily looked up through t_{arr}, detailed in Algorithm 4.1. This lookup process becomes more accurate once a vehicle sends its route to an intersection, since this places it in the schedule. Once a vehicle is in the scheduled flow, calculating the expected $t_{lightDelay}$ is simply a matter of understanding

where a cluster falls in a schedule, which is defined by t_{arr}. Future work will be done to uniquely identify and access clusters in a schedule, instead of relying on the schedule approximation described here. However, this approximation is still necessary for vehicles that are about to enter a network and would not yet appear in the schedule.

Algorithm 4.1 Estimation of $t_{lightDelay}$

1. Receive $t_{lightDelay,n-1}$, $t_{arr,n-1}$, $t_{freeFlow,n}$, $phase_{des}$
2. Initialize $t_{arr,n} = t_{arr,n-1}$
3. $t_{arr,queue} = t_{arr,n-1} + t_{lightDelay,n-1} + t_{freeFlow}$
4. $N_{inQueue} = countClustersInFront(t_{arr,queue})$
5. $t_{queueDelay} = slt + \frac{N_{inQueue}}{sfr}$
6. Send cluster message $c = (t_{arr}, t_{arr,end}, |c|)$ to Surtrac.
7. Receive scheduled clusters from Surtrac.
8. If $phase_{des} == schedule(t_{arr})$: $t_{lightDelay} = 0$.
9. Else: $t_{lightDelay} = find(schedule == phase_{des,next}) - t_{arr}$

In Algorithm 4.1, heuristics are calculated to find the arrival time, which is sent to Surtrac in the form of a cluster message. Once Surtrac has processed all clusters for a particular intersection, the algorithm finds the next place in the schedule where a cluster will be allowed to pass through the intersection for its desired phase. The delay caused by the traffic light is this next passage time, minus the vehicle's expected arrival time to the intersection.

5. Cluster Weighting

A cluster has not only a start and end time, but also a weight, used to define the approximate number of vehicles in a cluster. The weight of a cluster can also be thought of as a measure of certainty, that is, the closer a cluster's predicted arrival time matches actual arrival time, the closer the weight is to its true vehicle count. In the current Surtrac system, cluster weights are effectively the number of vehicles in the cluster, but with slight variation depending on the distance of the clusters from the intersection. Clusters passed between intersections have a low weight for long edges, where there is higher probability that part or all of a cluster may exit an edge mid-block, while shorter edges allow for higher weight clusters. Longer edges also cause lower cluster weights due to a higher uncertainty in arrival time than on short edges. This weighting provides a sense of fairness in the dynamic programming optimization, so that clusters more likely to pass through an intersection will have a larger effect on an intersection's schedule.

Route information enables a more intelligent weighting of CVs, instead of simply relying on edge length. Generally, the weights can be made high because of the added certainty that the vehicle will actually arrive at an intersection. There are cases in which a driver will deviate from a planned route and seek a new destination, but because Surtrac has no memory of vehicle clusters, these erroneous

clusters are not propagated through time. Additionally, connected vehicles allow for analysis of predicted and actual arrival times to an intersection, so that cluster weights can be scaled by arrival time uncertainty.

To combine these heuristics into a single weight for a cluster, we begin with a base weight of $w_b = 1$ for every route-sending CV. The first modifier of this base value is a weighting that accounts for higher uncertainty in arrival time for more distant intersections. Scaling values are chosen by running several simulations and averaging the t_{arr} errors for each number of intersections away from a cluster. A constant is applied to the inverse of errors, so that a low error results in higher weight, as in Tab. 5.1. Errors are acquired from simulations by grouping the arrival errors based on the distance from the intersection when the t_{arr} estimates were sent. The errors shown here are approximate, but represent the trend of errors observed in the 36-intersection network of sec. 7.1.1. Generally, error increases as waypoint intersection distance increases, with the exception of the closest intersection. Arrival prediction for very short distances, specifically one intersection away, can have high error due to complex inter-vehicle dynamics that do not average out in short distances [19].

Ints away	1	2	3	4	5	6
t_{arr}error	25%	20%	22%	25%	27%	30%
w_i	1	1.5	1.25	1	.75	.5

Table 5.1.: Weights and error percentages for number-of-intersections-away

The second weight modifier for CV clusters is based on approximate market penetration rate of CVs. As more vehicles communicate their routes, traffic light schedules become more accurate and contain a more complete picture of all vehicles. A simple scaling factor is applied so that connected vehicle weights increase

proportionally to market penetration rate 5.1.

$$w_p = w_{cur} w_{p,min} + \rho_{CV} \frac{w_{p,max} - w_{p,min}}{\rho_{CV,max} - \rho_{CV,min}} \tag{5.1}$$

Here, w_{cur} is the current weight of a cluster from any previos weighting steps. $w_{p,min}$ and $w_{p,max}$ are the weights that correspond to the minimum and maximum market penetrations ($\rho_{CV,min}, \rho_{CV,max}$) of CVs. We use $w_{p,min} = 1$ at the corresponding penetration rate $\rho_{CV,min} = 0$, and $w_{p,min} = 3$ for $\rho_{CV,min} = 1$. In total, a single cluster's weight is defined as the product of all three weights as in 5.2.

$$|c| = w_b * w_i * w_p \tag{5.2}$$

This total cluster weight, $|c|$, is used within the cluster format $c = (t_{arr}, t_{arr,end}, |c|)$.

6. Integration with Vissim

Routes are incorporated into the Surtrac scheduler via the existing Surtrac simulation framework, which relies on Vissim, a widely used microscopic multi-modal traffic simulator. The series of data structures and scripts that go between Vissim and Surtrac, known as Visco, serve as the starting point for adding route processing. In the existing framework, messages containing vehicle detections and signal states are sent to Surtrac, which in turn sends back messages commanding signal state changes (Fig. 6.1).

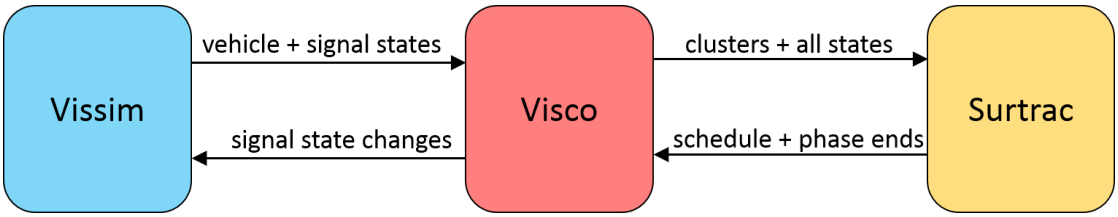


Figure 6.1.: Flow of data between Vissim and Surtrac, with newly added route/cluster sending and schedule receiving.

Route information is added by creating additional message types within Surtrac to receive cluster messages from and send schedule messages to the route cluster server. The route cluster server aggregates relevant connected vehicle information from Vissim, at every time tick, and schedule data from Surtrac, when available, to build the cluster messages according to the methods described in chapter 4 and

chapter 5. A vehicle's location and route are direct inputs to cluster arrival time and weight, while the vehicle speed is used to build a sense of edge freeflow travel time, as noted in sec. 4.1. Clusters on the same edge are aggregated and sorted by arrival time, and sent to an intersection as a single message at each time step. An intersection's Surtrac agent will thus receive a message for each edge that is an input to the intersection, and add the ordered clusters as in Fig. 3.1. These input edges are the edges for which the optimization algorithm picks an ordering of clusters to create a scheduled control flow. The scheduled control flow is sent back to the route cluster server to manage the schedule lookup process described in Algorithm 4.1. A summary of this whole process appears in Algorithm 6.1, where each step is preceded by the process that carries it out.

Algorithm 6.1 Data Flow between Sutrac and Vissim

1. Vissim: Run a simulation step.
2. Visco: Access data for stock Surtrac = all signal and detector states.
3. Visco: Access data for route-sending = all vehicle states.
4. Visco: Predict t_{arr} for n intersections for each CV, save as a cluster.
5. Visco: Send all signal, detector, cluster states to Surtrac.
6. Surtrac: Minimize delay of intersection over all local detections + CV clusters.
7. Surtrac: Send commands for phase transitions and schedules.
8. Visco: Update all schedules for t_{arr} predictions.
9. Visco: Change any Vissim signal states as commanded by Surtrac.

7. Results

7.1. Simulation Networks

7.1.1. 36-Intersection Test Grid

Algorithm 6.1 is first tested on a uniform 36-interestion 1-way grid, designed without turns so as to simplify the route choice and ensure a route is followed. The network is also designed in such a way as to ensure that Surtrac is constrained, though not unfairly so. Surtrac is bounded by a minimum green time for each phase, and this minimum is set to be longer than the average free-flow travel time of each edge. This simply enforces that the decision created by the schedule stays within its own local planning horizon of a upstream single intersection. The simplicity of this network (Fig. 7.1) lends itself to easier understanding of network-level trends that result from route-sending.

7.1.2. Baum and Centre 23-Intersection Network

A realistic 23-intersection network based on the Baum and Centre roads in Pittsburgh, Pennsylvania is also used to evaluate the effects of using route information. While the 6x6 network is designed to highlight improvements over Surtrac, this

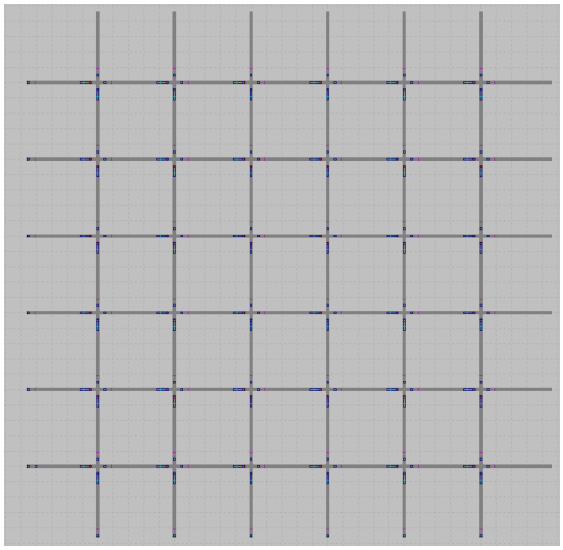


Figure 7.1.: 36-intersection grid with 1-way roads.

23-intersection network tests the algorithm in a more realistic setting. The 23-intersection network has already been analyzed to determine the improvement that Surtrac brings over existing optimized timings. This network will thus put into perspective the additional performance improvements on Surtrac by this algorithm, and demonstrate that using routes is applicable to more scenarios than those where Surtrac is intentionally constrained.

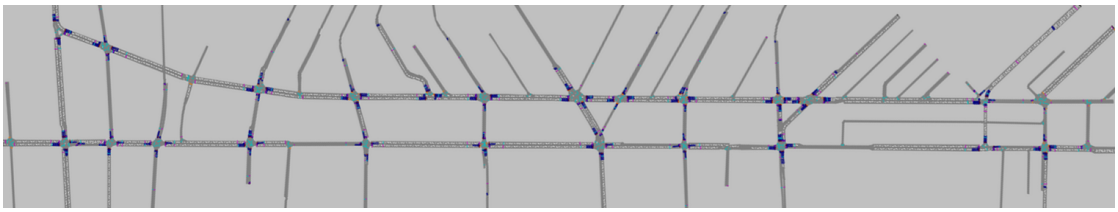


Figure 7.2.: 23-intersection network on Baum and Centre roads in Pittsburgh, Pennsylvania.

7.2. Control Variables

Market penetration rate of connected vehicles is a critical factor in determining performance of traffic control systems that rely on them. Therefore, the 36-intersection test network is analyzed across the full range of market penetration of connected vehicles, from 0% and in 10% increments through 100%, with extra points to test 1% and 99% CV rates. Each penetration rate point represents results averaged from a set of 5 randomly seeded simulations. Each seeded simulation runs for 1.5 hours (of simulation time) and uses a vehicle input volume profile specified in Tab. 7.1. All vehicle inputs in this network use the same volume profile.

Volume Profile	0-1800s	1800-3600s	3600-5400s
1	100	200	300
2	200	300	400

Table 7.1.: Volumes of vehicles present at all intersections in the 36-intersection network, at different simulation times.

Another important control variable is the number of intersections along a route (waypoints) with which each vehicle communicates. When vehicles are permitted to send their predicted arrival to a greater number of waypoints, the planning horizon of each Surtrac agent expands, simply because it utilizes information that is further into the future. On the contrary, limiting the allowable communication distance shortens the planning horizon. We choose to test 3 different intersection-horizons, namely {2, 4, 6} intersections away from a vehicles current location. For example, a two-intersection horizon indicates a vehicle will send its estimated arrival time and weight to the next 2 intersections along its path, as is shown being done in Fig. 3.2. Note that in Fig. 3.2, the vehicle does not send to the 3rd intersection. Finally, routes of connected vehicles are seeded for every [volume profile,

market penetration rate] pair for comparison between communication horizons.

7.3. Simulation Results

7.3.1. Volume Profile 1

Results for connected vehicles and non-connected vehicles are separated, and are shown along with aggregated results for all vehicles. Connected vehicles are also denoted as route-sending vehicles.. The aggregated performance results for all vehicles will obviously converge to the non-connected vehicles at 0% penetration rate, and to connected vehicle results at 100% penetration rate.

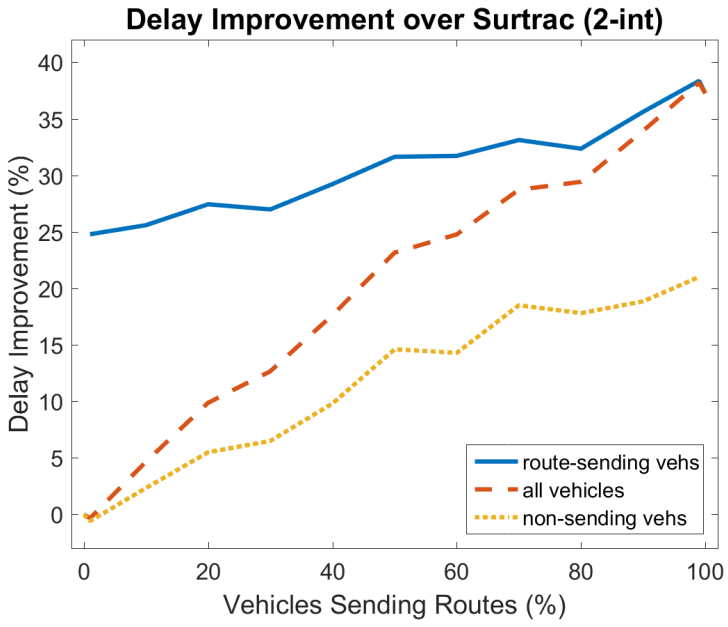


Figure 7.3.: The solid blue line is delay improvement by vehicles that send their routes to Surtrac. The dotted yellow line represents improved delay by non-sending vehicles. The red dashed line shows the overall network improvement.

Delay, the amount of time a vehicle loses while traveling at a speed lower than free-flow, is the primary performance factor considered here. The goal is to mini-

mize the delay throughout an entire traffic network. Improvement in delay refers to a percentage decrease in vehicle delay for two different scenarios. We present the results from the combination of all t_{arr} and weighting heuristics in Fig. 7.3, while each individual heuristic feature is more closely examined for performance in Appendix A.

Two prominent results are seen in Fig. 7.3 (and corresponding table of results Tab. 7.2). First, route-sending vehicles see a high benefit at extremely low penetration rates, gaining even a 25% reduction in delay with just 1% concentration of CVs. This suggests that even a single CV that sends routes would see a similarly high benefit. Note that for this lower volume test, a 1% CV concentration translates to only a couple of CVs in the network at a given time tick. Thus, the improvement at these low CV concentrations is due to the individual CV route-sending, rather than as a result cooperative behavior. This performance benefit for CVs increases as more vehicles send their routes, to a 37% delay improvement when 100% of vehicles are CVs.

Second, non-CVs are benefitted by the added information provided by route-sending CVs. Despite a slight -.6% performance decrease with 1% route-senders, non-CVs generally see a higher benefit as CV market penetration increases. When 99% of vehicles are sending routes, non-CVs still benefit despite being in the vast minority, undergoing 21% less delay than if no routes are sent.

Concentration of CVs	CVs	Non-CVs	Total (CVs + Non-CVs)
1%	24.8%	-.59%	-.35%
50%	31.6%	14.6%	23.2%
99%	38.4%	21.0%	38.2%

Table 7.2.: Percentage reduction of delay for CVs, Non-CVs, and all vehicles together, in comparison to stock Surtrac at critical CV compositions.

For all communication horizons, performance can also be analyzed in terms of number of stops that a vehicles makes, and its total travel time. The decrease in stops for connected vehicles at 1% market penetration is approximately 29.1%, and at 100% market penetration is approximately 54.3%, for the 2-intersection horizon. Tab. 7.3 summarizes these results, which follow a trajectory similar to the delay results in Fig. 7.3.

CVs Present	CVs		Non-CVs		Total	
	stops	travel time	stops	travel time	stops	travel time
1%	29.1%	13.2%	.43%	-.31%	.71%	-.17%
50%	47.8%	17.2%	32.7%	7.9%	40.3%	12.6%
99%	54.3%	20.8%	40.9%	11.6%	54.2%	20.7%

Table 7.3.: Percentage reduction in stops and travel times, in comparison to stock Surtrac at critical CV compositions.

7.3.2. Arrival Error Analysis

7.3.2.1. Error As CV Concentration Varies

Heuristic accuracy is analyzed through error in t_{arr}, as a function of both simulation time and market penetration rate. Fig. 7.4 shows that as more CVs enter the market, our route cluster server maintains a consistent t_{arr} predictive ability, which highlights its effectiveness at low penetration rates when schedules are an incomplete representation of all clusters.

7.3.2.2. Error As Vehicle Volumes Vary

Likewise, the error stays consistent throughout the duration of a simulation, even as vehicle volumes rise, with the exception of the 1-intersection-away prediction,

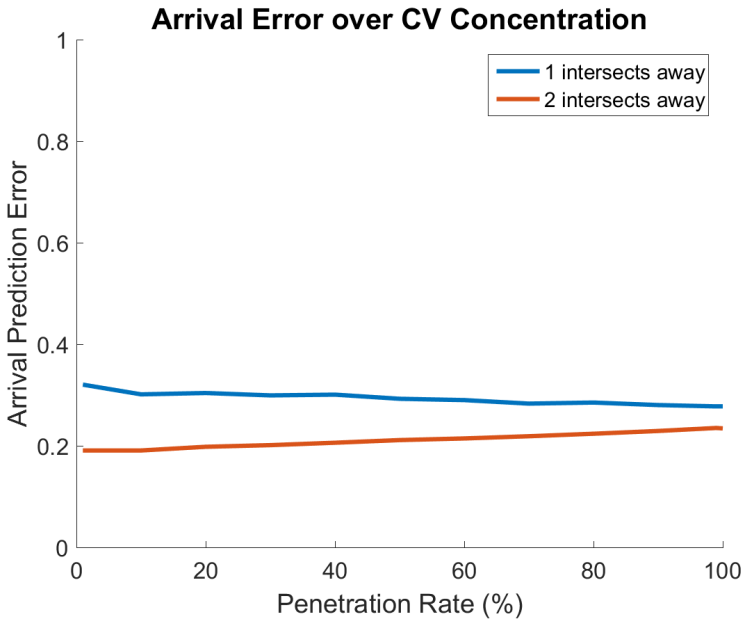


Figure 7.4.: Overall, error in t_{arr}stays approximately constant, as the error from different intersection distances converge.

discussed below in sec. 7.3.3.1. The predictions are robust to the queuing that takes place in this first volume profile, as seen in Fig. 7.5. As volumes increase at 3600 seconds, to the point where some queues are formed, there is a jump in prediction error, but it rounds off and begins to decrease. While future work can be done to improve the error behavior of the arrival prediction in the presence of queues, this method is sufficient in that the error does not grow out of control. Further, it should be noted that the goal is not to achieve an optimized t_{arr} estimate, but rather make use of a potentially poor estimate in a way that benefits a traffic control system.

7.3.2.3. Comparison of Error to Literature

As previously stated, the intent of our t_{arr} heuristic is to work in concert with $|c|$ to provide a better representation of vehicle arrival. However, it is still helpful

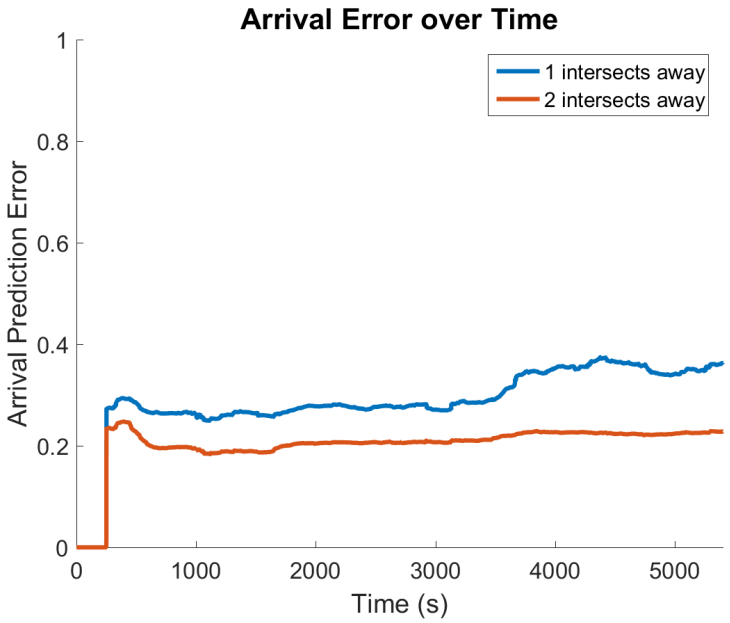


Figure 7.5.: Error in t_{arr}as volumes increase at 1800 seconds and 3600 seconds. At each time when volumes increase, the error is bumped up slightly and then levels out.

to anchor the error we observe, which falls between 20% - 30%, depending on the number of waypoint intersections receiving cluster messages. One recent work that uses mobile phone GPS data across a variety of techniques [31] reports a 10.4% error for the TRIP model, and 13.2% error for a linear regression method. These errors were accumulated over a minimum of 3 kilometers, however, while our prediction errors cover about 100 - 300 meters. As shown in Fig. 7.5, intersections that are further can be easier for predicting accurate arrival times, where local variations in traffic flows will be largely averaged out [19]. Likewise in [30], where a Support Vector Regression method is used, distances of 45 and 350 kilometers cause errors of 4.4% and 1.2%.

For methods predicting t_{arr} over shorter distances, errors are more comparable, although the primary concern is of buses or predicting public transit vehicles.

One neural network-based approach for modeling travel times between bus stops, which are typically only a few intersections away, observed an average prediction error of 18.3% [10]. Another work, which develops predicted bus arrival times to enable bus priority, creates a fused method that achieves 35% error at 200 meter distances and 5% error at 450 meter distances [29]. This more closely reflects our performance at 1-2 intersections, which is 100 - 300 meters in distance.

While our method may be simple, it benefits from use of primarily real-time traffic information, including the actual traffic light schedules. These real-time metrics lend themselves naturally to estimating short-term arrivals. We propose an integration with long-term historical data as a topic of future work.

7.3.3. Choosing Intersections for Route-Sending

7.3.3.1. Impact of Sending to Nearest Intersection

The large jump in 1-intersection-away error in Fig. 7.5 can be attributed to the use of double-counting CVs. Surtrac maintains local detections of vehicles between intersections. Thus, additional cluster information from routes will duplicate local detections if a vehicle transmits its route to the next intersection while between Surtrac intersections. Though this causes an overestimation of number of cars on a link, creating error in estimation of $t_{queueDelay}$, experimental evidence confirms that double counting is beneficial. When route-sending to the nearest intersection is eliminated, delay improvements rise similarly to the 50% CV mark, then decrease significantly (Fig. 7.6). This can be attributed to the greedy nature of the current Surtrac implementation, where clusters that are nearer in time to an intersection command the schedule determination. Thus, a high weight is achieved by double

counting these clusters, so that Surtrac is forced to build a schedule around clusters that are very near (and thus better satisfy the greedy assumption).

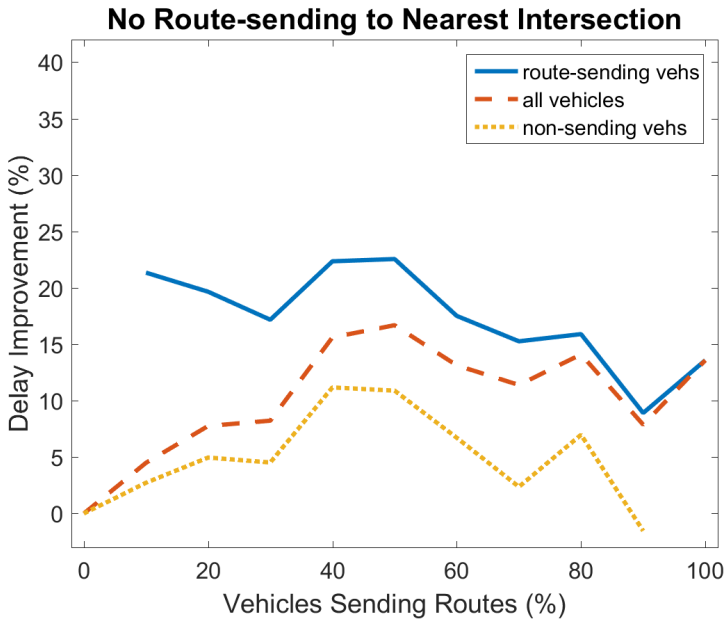


Figure 7.6.: Benefit of route sending decreases past 50% CV concentration when routes are not sent to the nearest intersection.

While sending routes to the nearest intersection may overestimate the number of clusters, it provides Surtrac with a better real-time estimate of when a CV arrives. Surtrac's local detection is simply time-shifted from its original upstream detection via the edge's speed limit. These local predictions may be accurate on average, but may vary wildly per vehicle. Because Surtrac uses a greedy approach to optimization, it relies more heavily on the accuracy of vehicles closer to an intersection, as it builds a schedule into the future. Thus, a higher t_{arr} accuracy in the short range will boost schedule accuracy at the beginning of the schedule, which allows for the greedy method to find a more globally optimal solution.

7.3.3.2. Sending Routes to More Intersections

Expanding the communication horizon out to 4 and 6 intersections, we see a similar increase in performance, but diminishing returns in terms of messages being sent. In Fig. 7.7 there is no additional improvement in delay compared to Fig. 7.3, and in fact the 100% CV state has a lower delay improvement of 34% compared to 37% before.

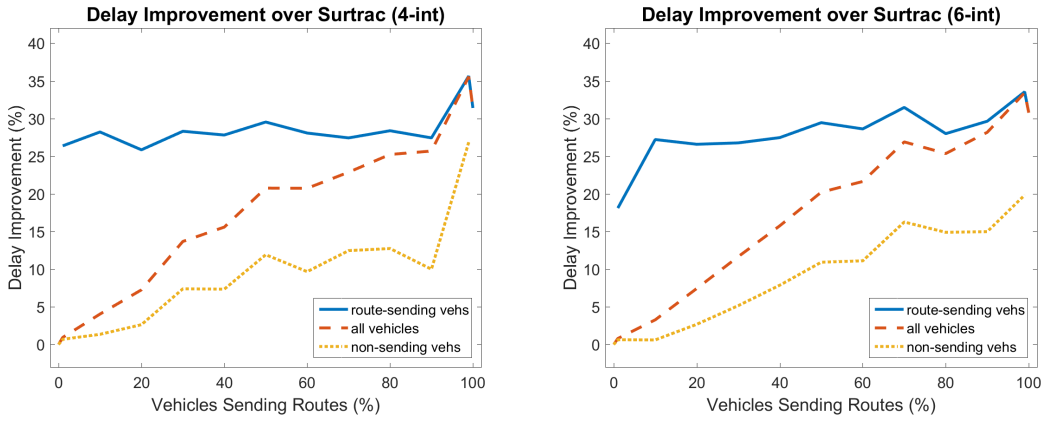


Figure 7.7.: There is no significant benefit for sending to 4 or 6 intersections, in this network, compared to 2 intersections.

As has been mentioned, Surtrac's greedy scheduling method naturally relies more on the clusters closest to an intersection, explaining the large benefit of sending schedules to the nearest two intersections. Sending to more intersections does have some benefit, but could be better used in a scheduler that considers a larger search space rather than greedily choosing local optimum [32]. Adding information that is further away, in this case, only adds to the error in the schedule without providing useful information that will be leveraged within the current phase.

7.3.4. Volume Profile 2

Within the 36-intersection grid, we increase the volumes of all vehicles inputs by 100 vehicles per hour for the entire simulation, and observe similar results. The same CV concentrations are used here, with the exception of the 1% and 99% points, and are averaged over 4 random seeds.

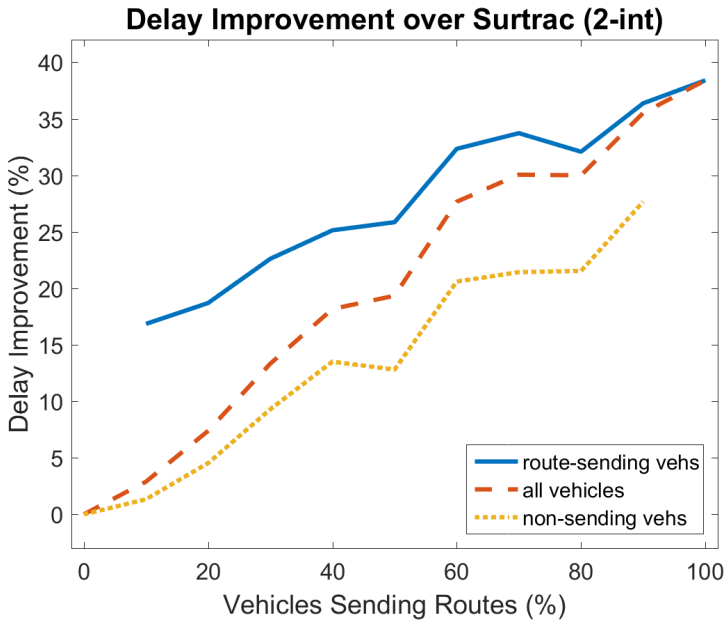


Figure 7.8.: When vehicle volumes increase, we observe a similar total improvement in delay. In comparison to the lower volume scenario, the rates of delay improvement are higher for both CVs and non-CVs alike.

The most noticeable difference is the smaller gap between delay improvements in CVs and non-CVs, i.e. between the solid blue and dotted yellow lines. One possible explanation for this is that the higher volume of vehicles, combined with the limited choice of routes, allows for non-CVs to be better represented in the schedule through the routes the CVs send. Essentially, because the vehicles can only travel in a straight path in the 36-intersection network, vehicles that do not send routes are closely following and reaping the benefit of other vehicles that are.

As CVs better represent the non-CVs, the trends of improvement are also similar, contributing to the approximately parallel trendlines seen between the two vehicle types. This behavior is a limitation of this test network, but it very effectively highlights the room for improvement in Surtrac through use of routes.

Another observation is that the delay improvement at low route-sending percentages is only 17% compared to the initial 25% from the first volume profile. This can be attributed to a larger error at low CV rates for the higher volume of vehicles, as shown in Fig. 7.9. This larger t_{arr} error starting at the 10% CV rate worsens the ability of the routes to contribute to schedule improvement.

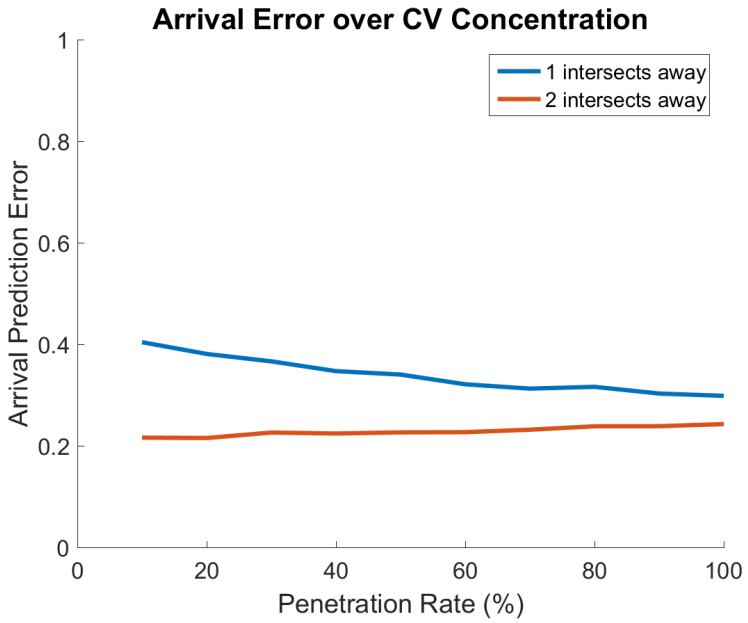


Figure 7.9.: Error for 1-intersection waypoint prediction is initially higher, at about 40%, than the lower volume scenario.

In Fig. 7.9, the initial value for t_{arr} error is much higher than with lower volumes in Fig. 7.4, but at 100% concentration of route-senders, converges to the same value. One simple explanation for the higher error at lower concentrations is that there are simply a higher number of vehicles that are unmodelled in the

schedule. Thus, when t_{arr} is estimated through use of the schedule-lookup process, the number of vehicles that are predicted to precede a cluster within a queue is vastly underestimated. While the sign of the error is not shown in the plot, the 1-intersection error is positive, which has been defined to mean the prediction was earlier than the actual arrival, which is plausible given the severe underestimation of queue size.

The trend appears to be that as the volumes in a network cause all edges to approach saturation (past capacity), the delay improvement curves for CVs and non-CVs will converge. This would make sense given this network architecture - as vehicles become more packed together, the CV route information becomes a nearly perfect predictor of the non-CV routes, in terms of arrival times. The whole network would still see a benefit, but the high initial benefit for CVs observed in Fig. 7.3 would not exist. This requires further experiments for testing, which we propose for future work.

7.3.5. Baum and Centre Results

A realistic volume profile is used for testing the Baum and Centre networks, built from turning proportions and volumes data provided by the City of Pittsburgh. Unlike the 36-intersection grid, each input is unique over the duration of the simulation. The simulation period covers the AM period, from 6:30am - 10:00am, and follows the ramp-up/down schedule in Tab. 7.4.

6:30 - 7:00	7:00-7:30	7:30-9:00	9:00-9:30	9:30-10:00
45%	67%	100%	67%	45%

Table 7.4.: Ramp up and down times and corresponding percentages of the maximum volumes at each input.

First, we demonstrate the utility of stock Surtrac, without route-sending, in comparison to the existing fixed timings, which have been optimized for the given vehicle volumes. The improvements in delay, stops, and travel time are shown in Tab. 7.5, averaged for all vehicles throughout the duration of the simulation, for a single random seed.

Method	Delay	Stops	Travel Time
Existing	241.2 s	15.3	421.9 s
Surtrac	135.7 s	7.2	312.5 s
Improvement	43.7%	52.9%	25.9%

Table 7.5.: Performance values for existing vs Surtrac-controlled timings. Improvement of Surtrac over existing is shown in the last row.

There is a massive advantage of Surtarc over existing in simulation, as delays are reduced by 44%. Buidling off of Surtrac's performance, we apply the same route-sending method as used in the 36-intersection test cases, across 4 different CV penetration rates, as in Tab. 7.6, where results are reported in terms of improvement over existing timings. Note that the 0% CV rate is the same as stock Surtrac with no route-sending. Results are from vehicles sending routes to 2 future waypoint intersections.

CV rate	Delay	Stops	Travel Time
0%	43.7%	52.9%	25.9%
10%	46.2%	58.8%	27.3%
25%	54.6%	68.6%	32.0%
50%	51.8%	66.7%	30.6%
100%	54.6%	68.6%	32.1%

Table 7.6.: Route-sending improvement over the existing timings control for Baum-Centre network.

The results in Tab. 7.6 provide an absolute comparison against a non-Surtrac

method on the same network, but comparison against stock Surtrac should also be noted, as in Tab. 7.7.

CV rate	Delay	Stops	Travel Time
10%	4.3%	12.5%	1.9%
25%	19.2%	33.3%	8.2%
50%	14.4%	29.2%	6.2%
100%	19.3%	33.3%	8.4%

Table 7.7.: Route-sending improvement over Surtrac for Baum-Centre.

Generally, results show a performance improvement for this realistic scenario, although there are diminishing benefits for higher concentrations of CVs. A likely explanation is the lower level of route control available in the Baum and Centre model than the 36-intersection grid.

7.3.5.1. Loss of Route Control

Though Vissim has fixed random seeds for our simulations, the vehicle routes are not constant and are not controllable. To determine routes of vehicles, we run a Surtrac simulation with no route-sending and record the paths of all vehicles for future route-sending simulations. Due to the way in which vehicles are dynamically assigned routes in this specific model, vehicles typically deviate from the final 1/3 of their pre-recorded path, on average. Once a vehicle deviates from its path, we no longer send cluster messages, and the vehicle effectively becomes a non-CV. Though unintended, this serves to model a scenario in which people have selected a route, but then deviate due to a change of plans or otherwise. Traffic control performance is still improved with this large population of deviating drivers, showing the system is somewhat robust to dropped routes.

While route deviation was not an issue in the 36-intersection grid, it has consequences on Surtrac performance here. Specifically, when a vehicle deviates, a cluster prediction in a Surtrac schedule becomes irrelevant, making that cluster detrimental to decision making prior to its removal from the system. When more vehicles are sending routes, more detrimental clusters are being added to Surtrac schedules, which causes a diminished benefit in delay reduction at higher CV penetration rates. In future work, we hope to remodel the Baum-Centre network or test on others that enumerate all routes such that vehicles are assigned routes according to the seed, eliminating this issue. There is some benefit to testing under heavily broken routes as done here, for modelling people who may be unlikely to follow a GPS path.

7.4. Discussion

7.4.1. 36-Intersection versus Baum-Centre

In comparison to Surtrac without route-sending, the 36-intersection network performs much better than Baum and Centre, reaching delay improvements of 38% compared to 19.3%. The higher performance on the 6x6 network is due largely to its even route distribution in comparison to the Baum-Centre network, in addition to the route control problem (sec. 7.3.5.1). Baum-Centre contains some routes that are very sparse, so if a CV makes a highly inaccurate prediction along one such route, there will be no cars at the intersection at the predicted time. Non-CVs at the intersection may then undergo a longer-than-expected wait time. The 6x6 network has evenly distributed routes, with all inputs using the same volume pro-

files. If a CV makes an inaccurate route prediction in this case, it is compensated by the other vehicles in different positions along the same route.

The uniform-CV distribution assumption may lead to suboptimal behaviors in some real-world settings. Specifically, a uniform spread of CVs ensures that the proportion of CVs on competing input flows approximates the proportion of all vehicles on those input flows. In reality, there may be an uneven distribution of CVs, such that there are significantly more on one side of an intersection – in this case non-CVs may not benefit as much from CV route-sending as they are overshadowed by cluster information from the opposing side.

7.4.2. CV-based Improvements in Literature

Traffic-related research on the network level can be difficult to relate to other works due to a variety of simulators, networks, and algorithms that rely on a variety of pieces. While we are able to compare to both existing timings and stock Surtrac performance on Baum and Centre, there is limited literature that attempts other approaches on this same set of intersections. Regardless, it is still useful to compare against performance benefits that other CV-based traffic conrol approaches produce.

One platoon-based approach designed for arterials that uses V2I communication [12] shows improvement for various vehicle volumes across compared to other methods. Vehicle delay is reduced by 16-29% compared to ACS Free, 1-10% over ACS Coord, and 17-20% over TSP Coord, depending on volumes. These improvements are at 100% concentration of CVs. For reference, coordinated approaches refer to those where neighboring intersection behavior is accounted for, while free

indicates an intersection runs without coordination.

Another CV-based approach shows improvements over a well-tuned fully actuated network of intersections, through use of only location and speed of vehicles. A 16% reduction in delay is achieved by minimizing queue lengths for the lower of the two vehicle volume scenarios tested [6].

These works do not directly ground our results, as they are tested on different networks and compared against different baseline methods. At 100% concentration of CVs, our results show a large improvement, as do the other methods listed here. At low CV concentrations under 30%, other methods are unable to benefit all connected vehicles , while our method reduces Surtrac's CV delay by 25-30%.

8. Conclusion

This method differentiates itself from other CV traffic control work in that it does not require a threshold market penetration of CVs to reap their benefit. Others have taken CV-only approaches and control traffic purely through their provided information, so a 30-40% concentration of these vehicles is needed to sufficiently capture traffic conditions [12, 18, 6]. This work, however, integrates both CVs and non-CVs within a unified traffic control framework, employing informative routes to lower vehicle delays across the full range of CV market penetration (Fig. 7.3).

A distinct advantage of this method is that it creates a high motivation for early adoption of route-sending or CV technology. In the 6x6 network, the first 1% of route-sending adopters experience, on average, 25% less delay when routes are sent. Thus if owners of CVs or autonomous cars are willing to provide routes, they are benefited greatly. By contrast, control systems with a CV threshold provide no benefit for the first 30-40% CVs, at which point CV-only traffic control becomes feasible.

We have shown that combining CV information with vehicle detections can boost Surtrac performance. By communicating routes as predicted arrival times, CVs not only gain a lower traffic delay, but also improve delay conditions for non-CVs.

9. Future Work

We have assumed that at least some routes are provided by vehicles, but this technique can also be extended to include vehicle route prediction. One such method employs a Maarkov Decision Process that relies on context such as snow, rush hours, or type of roads to predict routes from a trained model [33]. While the focus here has been on urban environments, other strategies that model lane change decisions on longer roads like highways could be used to predict where a car will travel base on lane choice [20]. These methods can be used in to compliment any actual route-sending that takes place, to build a real-time model of all route-choices in a network.

By combining detections of CVs and non-CVs alike, this work demonstrates how Surtrac can be extended to incorporate multiple modes of vehicle detection. This same framework can also be applied to incorporate routes from other types of agents, namely buses, emergency vehicles, cyclists, or pedestrians. Because a route can be converted into a cluster format usable by the Surtrac scheduler, any agent that is able to send its route can be included in the optimization of intersection space. Through manipulation of cluster weights, Surtrac's objective function can allow for prioritization of emergency vehicles or better handling of buses and bus stops. Pedestrians and cyclists can actually be included in the system in its current

state, requiring only a device that permits communication of routes.

As more modes of transportation are included within Surtrac, there is a greater need for accurate arrival estimates for each mode of travel. While some advanced estimation methods [30] exist that may be applied to this route-sending system, many arrival-estimating programs are already in use, specifically on mobile devices, that could be leveraged for a large population of drivers. Given the availability of routes today, there is great potential for integrating Surtrac with one of these systems to improve traffic in places where Surtrac is already deployed, or in future locations.

As demonstrated in this work, this integration benefits drivers greatly by reducing delay, but an even greater benefit could result if Surtrac is able to suggest a new route to a vehicle, through knowledge of real-time network data. Load-balancing could be used to increase the capacity of a network by suggesting routes that distribute flows more evenly, for example. Many more improvements can be made to Surtrac with the integration of route information, moving it toward a unified traffic system that optimizes globally across all modes of transportation.

Acknowledgments

I would like to thank my advisor, Dr. Stephen Smith, for his patience and guidance throughout my time at the Robotics Institute. I would also like to thank the other membors of my committee, Dr. Aaron Steinfeld and Hus-Chieh Hu, for their input toward and review of my work. Thank you to Dr. Gregory Barlow for teaching me the ways of Surtrac, and to Dr. Zachary Rubenstein for helping hone my research ideas. I thank all those who contributed toward my attendance of the Robotics Institute, those who worked alongside me in learning, and those who inspired me to finish in a timely manner.

A. Performance of t_{arr} and $|c|$ Features

All features listed in chapter 4 are intuitive and should lend themselves to a good approximation of t_{arr}. However, each should be examined separately for performance, to confirm that all features are contributing positively to the overall heuristic. Features were developed in sequential order, according to the order provided below. Thus, each performance will be in comparison to the previous feature added. This does not examine the full individual performance of each feature, but rather shows that each feature (or closely tied set of features) adds value. We likewise examine the benefits of each weighting mechanism for determining a cluster's $|c|$.

A.1. Traffic Light Delay and Edge Travel Time

The simplest method for approximating t_{arr} is to sum the expected node and edge costs for a graph representation of a traffic network. Node cost is the time spent waiting at an intersection, and edge cost is the time spent traveling a road that inputs into an intersection. This first test involves using fixed (i.e. non-

updating) travel times for each individual edge, and a simple schedule lookup process to determine approximate wait time for a cluster. Wait time is determined by difference between arrival and next available green phase for a cluster (sec. 4.3).

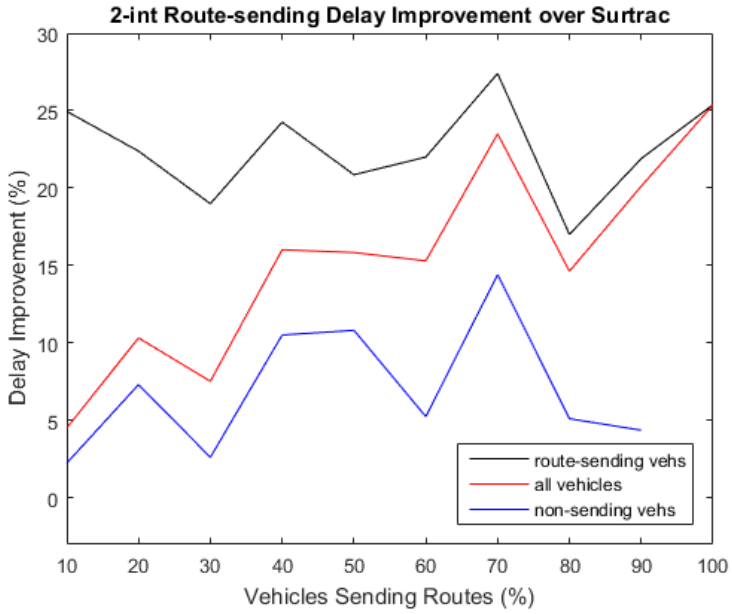


Figure A.1.: Large improvements over existing Surtrac-caused delays, for simple edge travel time and schedule look-up.

We run a single random seed over the market penetrations 10% to 100%, and observe the change in delay in Fig. A.1. Note that the variation in the curves is due largely to the randomization and noise from using a single random seed, for a relatively short simulation time of 5400 seconds. Clearly, these features provide a large benefit, for CVs which average about a 25% decrease in delay, and for non-CVs which average about a 7% delay decrease. We increase the number of intersections each vehicle is in communication with, to 6 intersections.

This larger communication horizon for CVs (Fig. A.2) results in similar benefits up to about the 70% mark and then decreases, especially for non-CVs. As a result of sending clusters to more intersections with all weights being equal, there is

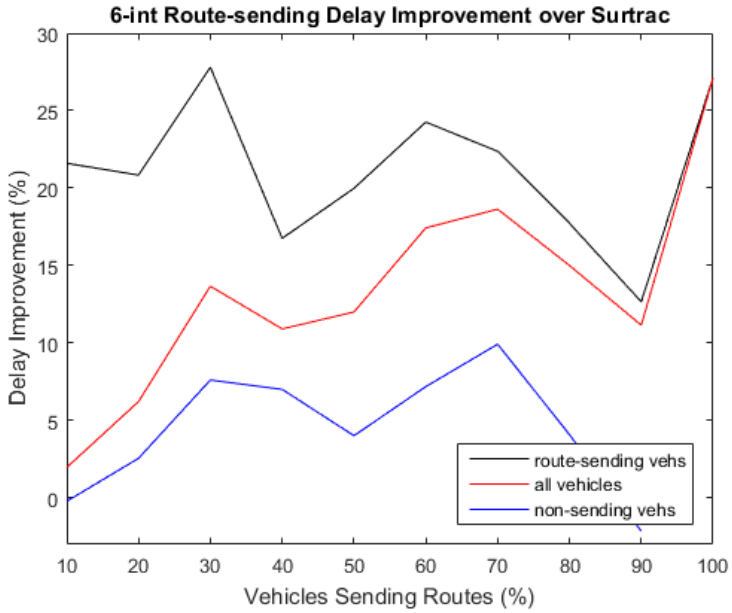


Figure A.2.: With a larger 6-intersection communication, non-CVs suffer at high penetration rates of CVs.

likely an observed cost for sending highly erroneous predictions very far away. We theorize that there is a significant part of arrival prediction missing that contributes to this decrease in performance.

A.2. Queue Delay

When traffic conditions are free-flowing and the network is sparse with only a few cars, using edge travel times and traffic light wait times are an appropriate approximation for travel time. However, as more vehicles enter the system, queues begin to form, and time is required to dissipate a queue before a vehicle reaches an intersection stop-bar. To analyze the addition of our $t_{queueDelay}$ estimation, we demonstrate that prediction error when volumes change is better managed than without queue estimation.

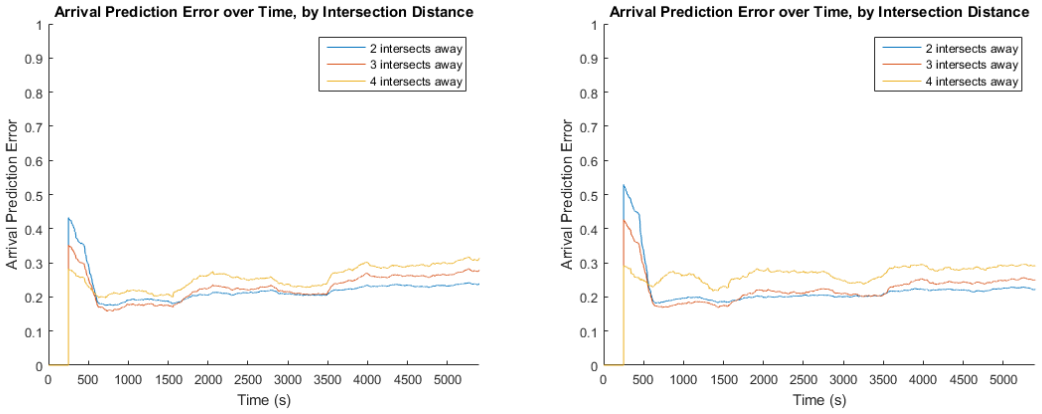


Figure A.3.: (Left) Error beginning at 3600 seconds begins to grow and continues through 5400 seconds. (Right) Error beginning at 3600 seconds increases, then levels off.

In Fig. A.3, t_{arr} error without queue prediction (left) for the 3 intersections receiving clusters begins at around 20%, and increases steadily to 25% over the course of the simulation. From 3600 seconds to 5400, error begins to grow quickly as more vehicles enter the network. However for the queue-estimation method (right), error starts at a higher value around 23%, and stays approximately constant throughout the simulation. Most importantly, as vehicle volumes increase from 200/hour to 300/hour, the error stabilizes after 500 seconds, while the non-queue estimation method has no such stabilization.

A.3. Variable Weights: Intersection Error

Compared to Fig. A.1, which has fixed weights for every vehicle, the results below deal with variably weighted clusters. First, a weight that corrects for error in intersections along a vehicle's route is examined. Here, intersection-distance is considered, in terms of number of intersections away a Surtrac agent is from a CV. Error for each distance of intersection is averaged over the course of several

simulations, to arrive at a weighting scheme that approximates that in Tab. 5.1. All weighting method results are shown for a single seed.

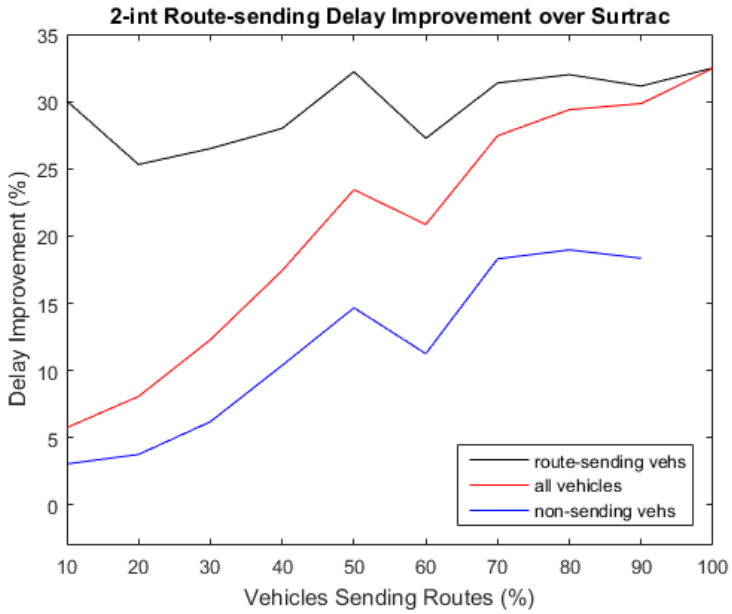


Figure A.4.: Performance in delay reduction is improved by varying the weights of clusters to match the error of estimation to intersections at different distances on a route.

A increase to 33% delay improvement is observed here, for all vehicles at 100% CV rate, compared to 25% delay improvement without variable weights, in Fig. A.1.

A.4. Variable Weights: Penetration Rate

Finally, an additional weight is added to account for the increased reliability that comes when more vehicles are in communication with Surtrac.

As the weight of CVs increases with the penetration rate of CVs, an increase from 33% in Fig. A.4 to 37% in Fig. A.5 is observed at the 100% CV rate.

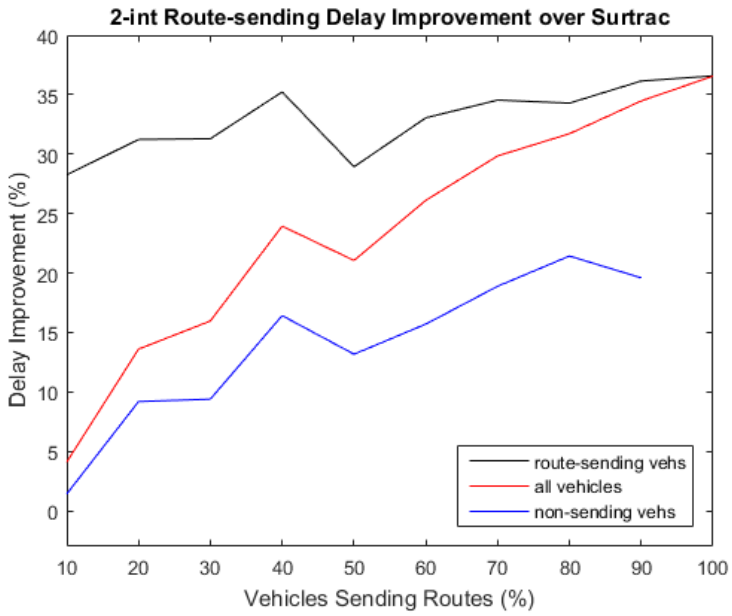


Figure A.5.: Delay is reduced further by adding a weight that changes according to the percentage of CVs in the network.

Bibliography

[1] "Connected car - united states market forecast." [Online]. Available: https://www.statista.com/outlook/320/109/connected-car/united-states#

[2] C. Cai, C. K. Wong, and B. G. Heydecker, "Adaptive traffic signal control using approximate dynamic programming," *Transportation Research Part C: Emerging Technologies*, vol. 17, no. 5, pp. 456–474, 2009.

[3] R. Cervero, "Unlocking suburban gridlock," *Journal of the American Planning Association J. of the Am. Planning Association RJPA*, vol. 52, no. 4, pp. 389–406, 1986.

[4] K. Dresner and P. Stone, "A multiagent approach to autonomous intersection management," *J. Artif. Int. Res.*, vol. 31, no. 1, pp. 591–656, Mar. 2008. [Online]. Available: http://dl.acm.org/citation.cfm?id=1622655.1622672

[5] D. Fajardo, T.-C. Au, S. Waller, P. Stone, and D. Yang, "Automated intersection control," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 2259, pp. 223–232, 2011.

[6] Y. Feng, K. L. Head, S. Khoshmagham, and M. Zamanipour, "A real-time adaptive signal control in a connected vehicle environment,"

Transportation Research Part C: Emerging Technologies, vol. 55, pp. 460 – 473, 2015, engineering and Applied Sciences Optimization (OPT-i) - Professor Matthew G. Karlaftis Memorial Issue. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0968090X15000091

[7] N. Gartner, F. Pooran, and C. Andrews, "Optimized policies for adaptive control strategy in real-time traffic adaptive control systems: Implementation and field testing," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 1811, pp. 148–156, 2002.

[8] D. Gettman, S. Shelby, L. Head, D. Bullock, and N. Soyke, "Data-driven algorithms for real-time adaptive tuning of offsets in coordinated traffic signal systems," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 2035, pp. 1–9, 2007.

[9] N. Goodall, B. Smith, and B. Park, "Traffic signal control with connected vehicles," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 2381, pp. 65–72, 2013.

[10] Z. Gurmu and W. Fan, "Artificial neural network travel time prediction model for buses using only gps data," *Journal of Public Transportation JPT*, vol. 17, no. 2, pp. 45–65, 2014.

[11] M. Hausknecht, T.-C. Au, and P. Stone, "Autonomous intersection management: Multi-intersection optimization," *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2011.

[12] Q. He, K. L. Head, and J. Ding, "Pamscod: Platoon-based arterial multi-

modal signal control with online data," *Transportation Research Part C: Emerging Technologies*, vol. 20, no. 1, pp. 164–184, 2012.

[13] ——, "Multi-modal traffic signal control with priority, signal actuation and coordination," *Transportation Research Part C: Emerging Technologies*, vol. 46, pp. 65–82, 2014.

[14] T. Hunter, P. Abbeel, R. Herring, and A. Bayen, "Path and travel time inference from gps probe vehicle data."

[15] D. Husch and J. Albeck, *Synchro Studio 7: Synchro plus SimTraffic and 3D viewer*. Trafficware, 2006.

[16] A. Jonsson and M. Rovatsos, "Scaling up multiagent planning: A best-response approach," 2011. [Online]. Available: http://aaai.org/ocs/index.php/ICAPS/ICAPS11/paper/view/2696

[17] J. Lee and B. Park, "Development and evaluation of a cooperative vehicle intersection control algorithm under the connected vehicles environment," *IEEE Trans. Intell. Transport. Syst. IEEE Transactions on Intelligent Transportation Systems*, vol. 13, no. 1, pp. 81–90, 2012.

[18] J. Lee, B. B. Park, and I. Yun, "Cumulative travel-time responsive real-time intersection control algorithm in the connected vehicle environment," *Journal of Transportation Engineering J. Transp. Eng.*, vol. 139, no. 10, pp. 1020–1029, 2013.

[19] M. J. Lighthill and G. B. Whitham, "On kinematic waves. ii. a theory of traffic flow on long crowded roads," *Proceedings of the*

Royal Society of London A: Mathematical, Physical and Engineering Sciences, vol. 229, no. 1178, pp. 317–345, 1955. [Online]. Available: http://rspa.royalsocietypublishing.org/content/229/1178/317

[20] Y. Luo, D. Turgut, and L. Boloni, "Modeling the strategic behavior of drivers for multi-lane highway driving," *Journal of Intelligent Transportation Systems*, vol. 19, no. 1, pp. 45–62, 2015. [Online]. Available: http://dx.doi.org/10.1080/15472450.2014.889964

[21] B. of Transportation Statistics, "Annual person-hours of highway traffic delay per auto commuter." [Online]. Available: http://www.rita.dot.gov/bts/sites/rita.dot.gov.bts/files/publications/national_transportation_statistics/html/table_01_69.html

[22] M. Papageorgiou, C. Kiakaki, V. Dinopoulou, A. Kotsialos, and Y. Wang, "Review of road traffic control strategies," *Proceedings of the IEEE Proc. IEEE*, vol. 91, no. 12, pp. 2043–2067, 2003.

[23] I. Porche and S. Lafortune, "Adaptive look-ahead optimization of traffic signals," *ITS Journal - Intelligent Transportation Systems Journal*, vol. 4, no. 3-4, pp. 209–254, 1999.

[24] C. Priemer and B. Friedrich, "A decentralized adaptive traffic signal control using v2i communication data," *2009 12th International IEEE Conference on Intelligent Transportation Systems*, 2009.

[25] S. Richter, D. Aberdeen, and J. Yu, "Natural actor-critic for road traffic optimisation," in *NIPS*, 2006.

[26] A. Sharma, D. Bullock, and J. Bonneson, "Input-output and hybrid techniques for real-time prediction of delay and maximum queue length at signalized intersections," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 2035, pp. 69–80, 2007.

[27] S. F. Smith, G. J. Barlow, X.-F. Xie, and Z. B. Rubinstein, "Smart urban signal networks: Initial application of the surtrac adaptive traffic signal control system." in *ICAPS*, D. Borrajo, S. Kambhampati, A. Oddi, and S. Fratini, Eds. AAAI, 2013.

[28] R. Tachet, P. Santi, S. Sobolevsky, L. I. Reyes-Castro, E. Frazzoli, D. Helbing, and C. Ratti, "Revisiting street intersections using slot-based systems," *PLOS ONE PLoS ONE*, vol. 11, no. 3, 2016.

[29] C.-W. Tan, S. Park, H. Liu, Q. Xu, and P. Lau, "Prediction of transit vehicle arrival time for signal priority control: Algorithm and performance," *IEEE Trans. Intell. Transport. Syst. IEEE Transactions on Intelligent Transportation Systems*, vol. 9, no. 4, pp. 688–696, 2008.

[30] Y. Wang, Y. Zheng, and Y. Xue, "Travel time estimation of a path using sparse trajectories," in *KDD 2014*. ACM, August 2014. [Online]. Available: http://research.microsoft.com/apps/pubs/default.aspx?id=217493

[31] C.-H. Wu, J.-M. Ho, and D. T. Lee, "Travel-time prediction with support vector regression," *IEEE Transactions on Intelligent Transportation Systems*, vol. 5, no. 4, pp. 276–281, Dec 2004.

[32] X.-F. Xie, S. F. Smith, L. Lu, and G. J. Barlow, "Schedule-driven intersection

control," *Transportation Research Part C: Emerging Technologies*, vol. 24, pp. 168–189, 2012.

[33] B. D. Ziebart, A. L. Maas, A. K. Dey, and J. A. Bagnell, "Navigate like a cabbie: Probabilistic reasoning from observed context-aware behavior," in *Proceedings of the 10th International Conference on Ubiquitous Computing*, ser. UbiComp '08. New York, NY, USA: ACM, 2008, pp. 322–331. [Online]. Available: http://doi.acm.org/10.1145/1409635.1409678