Planning for Improving Throughput in Autonomous Intersection Management

Tsz-Chiu Au, Michael Quinlan, Nicu Stiurca, Jesse Zhu, Peter Stone
Department of Computer Science
The University of Texas at Austin
1 University Station C0500
Austin, Texas 78712-1188
{chiu,mquinlan,jzhu,nstiurca,pstone}@cs.utexas.edu

Abstract

The impressive results of the 2007 DARPA Urban Challenge showed that fully autonomous vehicles are technologically feasible with current intelligent vehicle hardware. It is natural to ask how current transportation infrastructure can be improved when most vehicles are driven autonomously in the future. Dresner and Stone proposed a new intersection control mechanism called *Autonomous Intersection Management* (AIM) and showed in simulation that intersection control can be made more efficient than the traditional control mechanisms such as traffic signals and stop signs. In this paper, we extend the study to the real world by examining the relationship between the precision of cars' motion controllers and the efficiency of the intersection controller. First, we propose a planning-based motion controller that can reduce the chance that autonomous vehicles stop before intersections. Second, we present a mixed reality simulation environment that allowed an autonomous vehicle in the real world to interact with many virtual vehicles in the AIM simulator. Finally, we experimentally determine a feasible set of parameters for the motion controllers in simulation so as to give a more accurate account of the behavior of autonomous vehicles.

Introduction

Recent advances in intelligent vehicle technology suggest that autonomous vehicles will become a reality in the near future (Squatriglia 2010). However, today's transportation infrastructure does not utilize the full capacity of autonomous driving systems. Dresner and Stone proposed a multiagent systems approach to intersection management, and in particular describe a *First Come, First Served* (FCFS) policy for directing vehicles through an intersection (Dresner and Stone 2008). This approach has been shown, in simulation, to yield significant improvements in intersection performance over conventional intersection control mechanisms such as traffic signals and stop signs. Despite, or perhaps because of the impressive performance, questions are often raised regarding its applicability to real autonomous vehicles. In this paper we take the first step to show that a real autonomous vehicle can adhere to the FCFS protocol and efficiently traverse an intersection.

In achieving this first step, we have obtained more accurate details regarding real-world parameters for future use in simulation. In particular we have confirmed that the noise and imprecision of a real vehicle is greater than that anticipated in the simulation. However, the FCFS algorithm was designed with parameters (in the form of buffers) that can be enlarged to account for these deficiencies. Even after the enlargement of these buffers we can still demonstrate that FCFS provides a substantial performance improvement over the four-way stop sign currently located at our test intersection.

In addition, we believe that it is possible to make this intersection control mechanism more efficient by considering how best autonomous vehicles can utilize the FCFS protocol. Armed with an increase in real-world knowledge, we leverage Little's law in queueing theory to understand how the performance of an autonomous vehicle relates to the overall intersection throughput. Then we identify approaches to improve the motion controllers of autonomous vehicles such that they can plan ahead of time when they make reservations in the FCFS systems and traverse the intersection at a higher speed. We predict that the use of these planning techniques can improve the throughput of intersections and reduce the traversal time of vehicles, thus providing motivation for autonomous vehicles to adopt these planning-based controllers.

Autonomous Intersection Management

Traffic signals and stop signs are very inefficient—not only do vehicles traversing intersections experience large delays, but the intersections themselves can only manage a limited traffic capacity—much less than that of the roads that feed into them. Dresner and Stone have introduced a novel approach to efficient intersection management that is a radical departure from existing traffic signal optimization schemes (Dresner and Stone 2008). The solution is based on a *reservation* paradigm, in which vehicles "call ahead" to reserve space-time in the intersection. In the approach, they assume that computer programs called *driver agents* control the vehicles, while an arbiter agent called an *intersection manager* is placed at each intersection. The driver agents attempt to reserve a block of space-time in the intersection.

The intersection manager decides whether to grant or reject requested reservations according to an *intersection control policy*. In brief, the paradigm proceeds as follows.

- An approaching vehicle announces its impending arrival to the intersection manager. The vehicle indicates its size, predicted arrival time, velocity, acceleration, and arrival and departure lanes.
- The intersection manager simulates the vehicle's path through the intersection, checking for conflicts with the paths of any previously processed vehicles.
- If there are no conflicts, the intersection manager issues a reservation. It becomes the vehicle's responsibility to arrive at, and travel through, the intersection as specified (within a range of error tolerance).
- The car may only enter the intersection once it has successfully obtained a reservation.

Figure 1 diagrams the interaction between driver agents and an intersection manager. A key feature of this paradigm is that it relies only on vehicle-to-infrastructure (V2I) communication. In particular, the vehicles need not know anything about each other beyond what is needed for local autonomous control (e.g., to avoid running into the car in front). The paradigm is also completely robust to communication disruptions: if a message is dropped, either by the intersection manager or by the vehicle, delays may increase, but safety is not compromised. Safety can also be guaranteed in mixed mode scenarios when both autonomous and manual vehicles operate at intersections. The intersection efficiency will increase with the ratio of autonomous vehicles to manual vehicles in such scenarios.

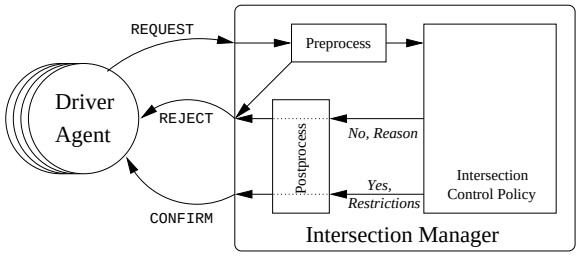
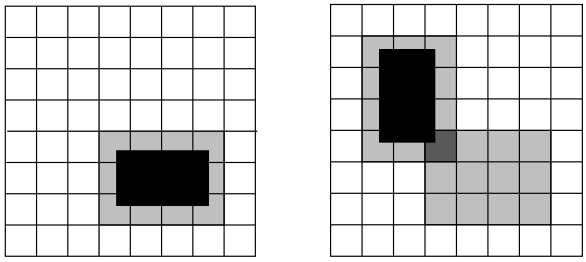


Figure 1: Diagram of the intersection system.

The prototype intersection control policy divides the intersection into a grid of *reservation tiles*, as shown in Figure 2. When a vehicle approaches the intersection, the intersection manager uses the data in the reservation request regarding the time and velocity of arrival, vehicle size, etc. to simulate the intended journey across the intersection. At each simulated time step, the policy determines which reservation tiles will be occupied by the vehicle.

If at any time during the trajectory simulation the requesting vehicle occupies a reservation tile that is already reserved by another vehicle, the policy rejects the driver's reservation request, and the intersection manager communicates this to the driver agent. Otherwise, the policy accepts the reservation and reserves the appropriate tiles. The intersection manager then sends a confirmation to the driver. If the reservation is denied, it is the vehicle's responsibility to maintain a speed such that it can stop before the intersection. Meanwhile, it can request a different reservation.



(a) Successful (b) Rejected

Figure 2: (a) The vehicle's space-time request has no conflicts at time t. (b) The black vehicle's request is rejected because at time t of its simulated trajectory, the vehicle requires a tile already reserved by another vehicle. The shaded area represents the *static buffer* of the vehicle.

Empirical results in simulation demonstrate that the proposed reservation system can dramatically improve the intersection efficiency when compared to traditional intersection control mechanisms. To quantify efficiency, Dresner and Stone introduce *delay*, defined as the amount of travel time incurred by the vehicle as the result of passing through the intersection. According to their experiments, the reservation system performs very well, nearly matching the performance of the optimal policy which represents a lower bound on delay should there be no other cars on the road (Figure 14 in (Dresner and Stone 2008)). Overall, by allowing for much finer-grained coordination, the simulation-based reservation system can dramatically reduce per-car delay by two orders of magnitude in comparison to traffic signals and stop signs.

Improving Throughput of Intersections via Planning Techniques

FCFS has been shown to be far more efficient than traffic lights and stop signs. But it is possible to make FCFS even more efficient by considering how autonomous vehicles can better utilize the FCFS protocol. In this section, we propose some planning techniques to improve the motion controller of autonomous vehicles such that the traversal time of the autonomous vehicles can be shortened and the overall throughput of the intersection can be increased.

Little's Law

First of all, let us consider factors that affect the maximum throughput characteristics of the FCFS protocol. An important result in queueing theory is Littles law (Little 1961), which states that in a queueing system the average arrival rate of customers λ is equal to the average number of customers T in the system divided by the average time W a customer spends in the system. In the context of intersection management, Little's law can be written as $L = \lambda W$, where

- L is the average number of vehicles in the intersection;
- λ is the average arrival rate of the vehicles at the intersection; and
- W is the average time a vehicle spends in the intersection.

Note that the arrival rate is equal to the throughput of the system since no vehicle stalls inside an intersection.

Little's law shows that the maximum throughput (i.e., the upper bound of λ) an intersection can sustain is equal to the upper bound of L divided by the lower bound of W, where the upper bound of L is the maximum number of vehicles that can coexist in an intersection, and the lower bound of W is the minimum time a vehicle spends in the intersection. Thus, Little's law shows that there are two ways to increase the maximum throughput: 1) increase the average number of vehicles in an intersection at any moment of time, and 2) decrease the average time a vehicle spends in an intersection.

A trivial upper bound on L is the area of the intersection divided by the average static buffer size of the vehicles. But this bound is rather loose and in practice unachievable. Nonetheless, it provides us some hints that the maximum throughput depends on the average static buffer size of the vehicles. But unfortunately the size of an intersection is a hard limit and the static buffer sizes cannot be too small—that is, there is little an intersection manager can do to squeeze more vehicles into the intersection. Therefore, we have to consider the second way to increase the maximum throughput.

Little's law shows that by reducing the average time a vehicle takes to traverse an intersection, the maximum throughput of the intersection can increase. In other words, a vehicle should maintain a high speed during the traversal of the intersection to shorten its traversal time. Vehicle's velocity in the intersection depends on two factors: 1) the initial velocity when the vehicle enters the intersection, and 2) the acceleration during the traversal. In the following sections, we will present two techniques that allow vehicles to maintain a high speed during the traversal.

Planning for Making Reservations to Avoid Stopping Before Intersections

One of the keys to entering an intersection at a high speed is to prevent vehicles from stopping before entering the intersection. FCFS, by itself, reduces the number of vehicles that stop at an intersection, and therefore it allows vehicles to enter an intersection at a high speed most of the time. In fact, it is one of the main reasons why FCFS is more efficient than traffic lights and stop signs (Dresner and Stone 2008). While FCFS has done a good job in this regard, there is still room for improvement on the autonomous vehicles' side such that driver agents can help by preventing themselves from stopping before an intersection.

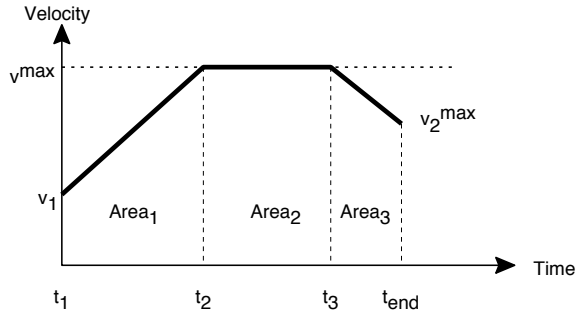
There are two scenarios in which an autonomous vehicle has to stop before an intersection in FCFS. First, the vehicle cannot obtain a reservation from the intersection manager and is forced to stop before an intersection. This happens when the traffic level is heavy and most of the future reservation tiles have been reserved by other vehicles in the system. Second, the vehicle successfully obtains a reservation but later determines that it will not arrive at the intersection at the time and/or velocity specified in the reservation. In this scenario the vehicle has to cancel the reservations and those reservation tiles may have been wasted. The effect of a reservation cancellation is not only that the vehicle in question has to stop, but also that temporarily holding reservation tiles may have prevented another vehicle from making reservation. Both of these effects lead to a reduction in the maximum throughput of the intersection.

A poor estimation of arrival times and arrival velocities can lead to the cancellation of reservations. In previous work, the estimation of arrival times and arrival velocities is based on a heuristic we called the *optimistic/pessimistic* heuristic, that derives the arrival time and arrival velocity based on a prediction about whether the vehicle can arrive at the intersection without the intervention of other vehicles (Dresner 2009). However, this heuristic presents no guarantee that the vehicle can arrive at the intersection at the estimated arrival time or the estimated arrival velocity; in fact, experiments have shown that vehicles are often unable to reach the intersection at the correct time and cancel their reservations. To avoid this problem we propose a new way to estimate the arrival time and arrival velocity. In our solution when a driver agent estimates its arrival time and arrival velocity, it also generates a sequence of control signals. These control signals, if followed correctly, ensure that the vehicle will arrive at the estimated arrival time and at the estimated arrival velocity. We can formulate this estimation problem as the following multiobjective optimization problem: among all possible sequences of control signals that control the vehicle to enter an intersection, find one such that the arrival time is the smallest and the arrival velocity is the highest.

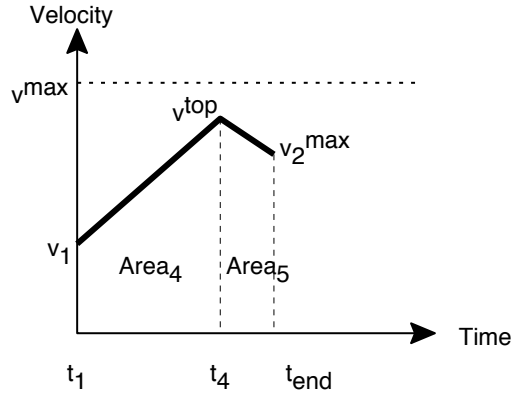
For an acceleration-based controller, the sequence of control signals is a time sequence of accelerations stating the acceleration the vehicle should take at every time step. We call a time sequence of accelerations an *acceleration schedule*. Like many multiobjective optimization problems, there is no single solution that dominates all other solutions in terms of both arrival time and arrival velocity. Here we choose arrival velocity as the primary objective, since a higher arrival velocity can allow the vehicle to enter the intersection at a higher speed. Our optimization procedure involves two steps: first, determine the highest possible arrival velocity the vehicle can achieve, and second, among all the acceleration schedules that yield the highest possible arrival velocity, find the one whose arrival time is the soonest.

We illustrate how the estimation procedure works using a time-velocity diagram as shown in Figure 3. In this figure, v_1 is the current velocity of the vehicle, t_1 is the current time, D is the distance from the intersection, a_{max} is the maximum acceleration, a_{min} is the minimum deceleration, v^{max} is the speed limit of the road, and v_2^{max} is the speed limit at the intersection. We can see that any function $v(\cdot)$ in the time-velocity diagram that satisfies the following constraints is a feasible velocity schedule for reaching the intersection, and the derivative of $v(\cdot)$ is an acceleration schedule for acceleration-based controller. (1) $v(t_1) = v_1$; (2) $v(t) \leq v^{max}$ for $0 \leq t \leq t_{end}$, where t_{end} is the arrival time; (3) $v(t_{end}) \leq v_2^{max}$; (4) $a_{min} \leq \frac{d}{dt}v(t) \leq a_{max}$ for

for $0 \leq t \leq t_{end}$ (i.e., the acceleration at any point in time must be within the limitations); and (5) $\int_{t_1}^{t_{end}} v(t)\,dt = D$.



(a) Case 1: $\text{Area}_1 + \text{Area}_3 \leq D$



(b) Case 2: $\text{Area}_1 + \text{Area}_3 > D$

Figure 3: The time-velocity diagrams for the estimation of arrival time and arrival velocity

Our objective is to find a function $v(\cdot)$ such that $v(t_{end})$ is as high as possible while t_{end} is as small as possible. We should only consider piecewise linear functions such that slopes of the line segments can be either a_{max}, a_{min}, or 0, because for any non-piecewise linear function that satisfies the constraints, we can always find a piecewise linear function with a smaller t_{end} and/or a larger $v(t_{end})$. First of all, find a point (t_2, v^{max}) in the velocity-time diagram such that (t_2, v^{max}) is an interception of the line extending from v_1 with slope a_{max} and the horizontal line $v = v^{max}$. Let Area_1 be the area of the trapezoid under the line segment from (t_1, v_1) to (t_2, v^{max}). Similarly, find an intercepting point (t_3, v^{max}) by extending v_2^{max} with slope a_{min} and let Area_3 be the area of the trapezoid under the line segment from (t_{end}, v_2^{max}) to (t_3, v^{max}). Note that Area_3 does not depend on the value of t_{end}. If $\text{Area}_1 + \text{Area}_3 \leq D$, the vehicle can accelerate to v^{max}, maintain the speed, and then decelerate to v_2^{max} and reach the intersection (Case 1). Then find d such that $d \times v^{max} = D - \text{Area}_1 + \text{Area}_3 = \text{Area}_2$. From this we can compute the acceleration schedule $\langle (t_1, a_{max}), (t_2, 0), (t_3, a_{min}) \rangle$ (Figure 3(a)). If $\text{Area}_1 + \text{Area}_3 > D$, the vehicle cannot accelerate to v^{max} because its velocity will exceed v_2^{max} when it enters the intersection (Case 2). Thus, we need to find a point (t_4, v^{top}) such that $\text{Area}_4 + \text{Area}_5 = D$ and the acceleration schedule is $\langle (t_1, a_{max}), (t_4, a_{min}) \rangle$ (Figure 3(b)). However, there are cases in which either $\text{Area}_4 > D$ or $\text{Area}_5 > D$ and we cannot find (t_4, v^{top}). In these cases, there is no feasible acceleration schedule such that the vehicle can arrive at the intersection while satisfying all the constraints.

We call the driver agent using this optimization procedure a *planning-based* driver agent, since it plans ahead of time the acceleration schedule before making reservations. To evaluate the planning-based driver agent we implemented it in the AIM simulator and conducted an experiment to compare it with the driving agent based on the optimistic/pessimistic heuristic implemented in (Dresner 2009). In this experiment, the intersection has four incoming lanes and four outgoing lanes in each of the four canonical directions. The speed limits of the lanes are set to be $25\ m/s$. The static buffer size of the vehicles are set to be $0.25m$, which is sufficient for the simulated vehicles in the simulator. Other parameters of the autonomous vehicles are: the internal time buffer is $0s$, the edge time buffer is $0.25s$, and the maximum acceleration is $4\ m/s$. Then we vary the traffic level of each lane from 0.1 vehicles per second to 0.3 vehicles per second, and at each traffic level we run the simulator for one hour (simulated time) and compute the average delay of the vehicles. The result is shown in Figure 4. From the figure, we can see that when the traffic level is below 0.15 vehicles per seconds, most vehicles can get through the intersection without stopping (i.e., average delay is almost 0) and there is little difference between the performance of both driver agents. However, when the traffic level is more than 0.15 vehicles per seconds, the average delay of our planning-based driver agents is much lower than the average delay of the driver agents based on the optimistic/pessimistic heuristic. When the average delay of the heuristic-based driver agents levels off at the 0.25 traffic rate (which indicates that the throughput of the intersection has been saturated), the average delay of the planning-based driver agents remains low. Thus the use of our planning-based controller can increase the maximum throughput of the intersection and reduce the average delay, and this gives enough motivation to the autonomous vehicles to adopt our planning-based controller.

Planning for Faster Traversal in Intersections

The motion planning technique in the previous section can effectively prevent autonomous vehicles from unnecessary stopping before an intersection and in turn increase the initial velocity of the vehicles when they enter the intersection. The result is a reduction of the average traversal time and an increase in the maximum throughput according to the Little's law. Another way to reduce the average traversal time is to control the acceleration of vehicles inside an intersection such that the vehicles can pass through the intersection as quickly as possible.

According to the FCFS protocol, when the intersection manager grants a reservation to a vehicle, it will send an confirmation message to the vehicle. The confirmation message includes an piece of information called *acceleration profile*, which states how the vehicle should adjust its acceleration

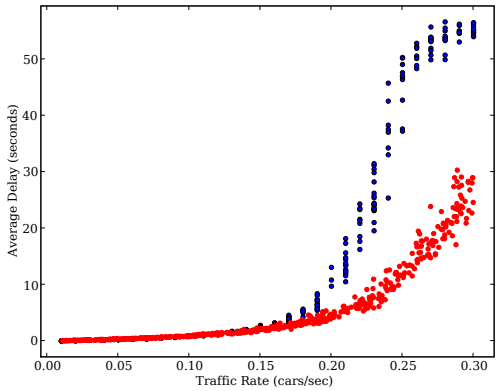


Figure 4: Comparison of the planning-based driver agent (red dots) with the driver agent based on the optimistic/pessimistic heuristic (purple dots).

during the traversal of the intersection. Formally, an acceleration profile is a list of pairs $\langle (a_1, t_1), (a_2, t_2), \ldots, (a_n, t_2) \rangle$, such that the vehicle, upon entering the intersection, should maintain an acceleration of a_1 for t_2 second, and then an acceleration of a_2 for t_2 second, and so on. But in the current implementation of the AIM simulator, an acceleration profile contains at most two pairs, namely $\langle (a_1, t_1), (a_2, t_2) \rangle$, where a_1 is the maximum acceleration of the vehicles, and a_2 is zero. Hence the vehicles in the simulator accelerates as much as possible for a certain period of time, and then maintain the speed when it is ready to leave the intersection. First of all, let us examine how the maximum acceleration of a vehicle affects intersection throughput in this setting.

We conducted an experiment using our simulator to see the change in average delay and throughput as the maximum acceleration of the vehicle increases. In the experiment, we again set the traffic level of each lane to be 0.16 vehicles per second, the speed limit of the roads to be $25\ m/s$, the static buffer size to be $0.25m$, the internal time buffer to be zero seconds and the edge time buffer to be $0.25s$. The maximum acceleration of the vehicles is then varied from 0 to 6 m/s^2, with the results shown in Figure 5 and Figure 6. We can see that as the maximum acceleration increases the average delay drops to zero quickly and the throughput increases to a level that is identical to the incoming traffic level.

To see why the throughput can be increased quickly by a small increase in the maximum acceleration, let us consider the relationship between the lower bound of the traversal time and the maximum acceleration. Let v_0 be the arrival (initial) velocity of the vehicle when it enters an intersection, d is the length of its trajectory in the intersection, v_1 be the velocity when the vehicle leaves the intersection, W be the time the vehicle spends in the intersection, and a be the acceleration, assuming that the vehicle accelerates at a constant rate during the entire traversal. If v_0 is almost equal to the maximum velocity of the vehicle and the speed limit of the road, the acceleration has little effect on the traversal time W; otherwise, we have $W = \frac{\sqrt{v_0^2 + 2ad} - v_0}{a}$ by solving the following two equations: $v_1 = v_0 + aW$ and $W = (v_1 - v_0)/a$. If v_0 is 0, then $W = \frac{\sqrt{2d}}{\sqrt{a}}$, and the throughput λ, which is inversely proportional to W, is proportional to $\sqrt{a}$. If v_0 is not zero, we can check that λ is approximately proportional to $\sqrt{a}$. This explains the shape of the line in Figure 6. If the traffic level is smaller than the maximum throughput an intersection can sustain, almost all vehicles can pass through the intersection without stopping, and it is indeed the case when the maximum acceleration is larger than 2 m/s^2 in Figure 5. In short, the maximum throughput is approximately proportional to the square root of the maximum acceleration of vehicles, and we can increase the throughput by increasing the acceleration.

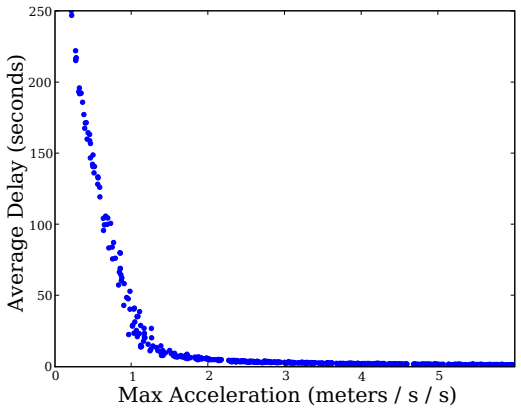


Figure 5: Maximum acceleration versus average delay.

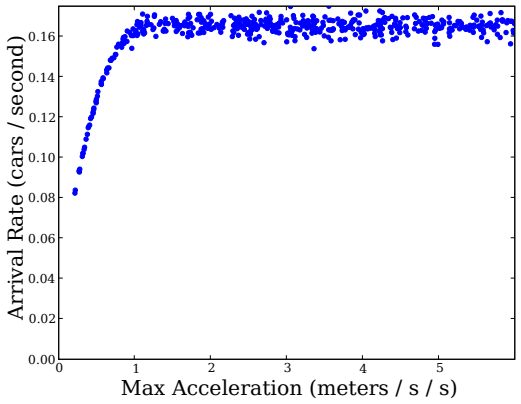


Figure 6: Maximum acceleration versus arrival rate.

The above experiment shows that acceleration profiles issued by an intersection manager can have a great impact on the maximum throughput of an intersection. An interesting question is how to find the best acceleration profile for vehicles to go through an intersection as quick as possible. Since the acceleration profile determines the trajectory of the vehicles, the question becomes how to compute the fastest trajectory for the vehicle. In the future, we intend to apply planning techniques to compute acceleration profiles for fast traversal in intersections.

Reality Check: AIM in Practice

Simulations inevitably will approximate critical aspects of reality. For example, real cars may not be as precisely controllable as the simulated cars in simulations, nor may GPS provide as reliable location estimates. The true test of the intersection management scheme will be how well it works in the real-world, with real autonomous vehicles. In other words, in order to do realistic action planning for autonomous vehicles at intersections, we need to ground it with realistic motion planning, and that needs to be verified in the real world.

Completely testing the AIM system on real hardware would require a fleet of autonomous vehicles, which is currently infeasible and potentially unsafe given the risk of collisions during testing. Instead, we take the first step and implement a mixed reality system that will allow us to test the system using a single autonomous vehicle (Figure 7(a)) and many virtual (or simulated) vehicles.

Mixed Reality Experiments

To facilitate the mixed reality experiments, modifications were required to both the AIM simulator and our existing autonomous vehicle software. While the details of these are mainly engineering, the key aspect was the addition of *proxy vehicles* to the AIM software. A proxy vehicle acts as a gateway for a real autonomous vehicle to interact with the simulated environment. As far as the Intersection Manager and the FCFS policy are concerned the proxy vehicle is identical to the virtual vehicles. The difference exists only in the actual simulation aspect. Instead of the vehicle location being modelled by motion equations, the location of the proxy vehicle is now updated via UDP packets containing GPS locations from a real robot.

In our setup, we ran the AIM software on a computer located in the autonomous vehicle. As the autonomous vehicle approaches an intersection, it sends a "Request Message" over a specified UDP port to the AIM software. Meanwhile AIM has been simulating potentially 100's of other vehicles. AIM will then grant (or reject) a reservation to the robot based on this 'virtual traffic' (Figure 7(b)). Once a reservation is granted the vehicle will progress through the intersection in both the real-world and the simulated world. [1]

Implementing the Protocol on an Autonomous Vehicle

Our vehicle has a layered control system with different modules controlling different aspects of the vehicles movements. In this section we will present a brief overview of the modules required to explain the current implementation of the AIM protocol. The *navigator* module is a hierarchical state machine that defines what behavior the vehicle executes. Its main goal is to plan a path between way-points on a map, and to create velocity and heading commands that will get the vehicle to the destination while remaining on the road.

[1] An example of these experiments can be seen in the video at `http://www.cs.utexas.edu/~mquinlan/MixedReality.wmv`

(a) Autonomous Test Vehicle (b) Mixed-Reality Simulation

Figure 7: a) The full sized autonomous vehicle used in the mixed-reality experiments. b) A picture of the AIM simulator running a mixed-reality experiment. The green box (in the middle of the intersection) is a *proxy vehicle* that represents the real world location of the autonomous vehicle. The while boxes are virtual vehicles with reservations, while the yellow boxes are virtual vehicles without reservations.

The *pilot* module takes velocity and heading commands and determines what throttle, brake and steering actuator movements are required to achieve the desired motion.

There were two important changes made to the vehicles code. First is the addition of a new module, an *Autonomous Intersection Observer*. This observer handles all aspects of sending and receiving AIM protocol messages with an Intersection Manager. The observer also constantly sends a message to the simulator that details its current state, i.e., velocity, acceleration, heading, and position so that AIM can represent the real vehicle in the simulator. Navigator queries the observer to discover if the vehicle has been granted a reservation and if so, what velocity, accelerations are required to go through the intersection safely.

The arrival time to the intersection is calculated as the distance to the intersection divided by the current velocity. In addition we have a known arrival velocity of 3 m/s as this was the stop-approach speed dictated by the DARPA Urban Challenge rules, which our vehicle is designed to adhere too.

Adjusting AIM Parameters

The initial real-world experiments identified areas in which the simulation failed to capture the errors present in our autonomous vehicle. In this section, we will discuss how we modified the AIM parameters to facilitate the safe traversal of the intersection.

First, the experiments were undertaken on a road with a legal speed limit of 20 mph, therefore the maximum speed (*Max Speed*) of both the autonomous vehicle and the virtual traffic was set to 7.5 m/s (17 mph).

Second, AIM requires us to define the operating characteristics of the vehicle. Originally we simply transposed the known vehicle specifications for length, width, acceleration (0 to 60 mph) and deceleration etc. However it became clear that AIM interpreted 'max' to mean 'expected', which presented several problems. While our vehicle is technically capable of accelerating at 2.5 m/s^2 if never does, in fact near intersections our controller rarely exceeds 0.5 m/s^2. Similarly the speed limit is treated as an upper bound by our controller; the vehicle may drift marginally below this limit

for safety reasons and to facilitate smoother braking when approaching a stop.

To better understand the true impact of these parameter modifications we ran simulations to measure the effect on average delay. Figure 8 plots average delay across a range of traffic densities, with each color representing a different set of parameters. You can observe that the default FCFS parameters [2] result in little delay and in fact the intersection never becomes congested even at higher traffic rates. In comparison the parameters that are reflective of the vehicle in reality result in significant average delay above a traffic rate of 0.06 vehicles per second per lane.

We then plotted the result of changing a single parameter. As you can see reducing the *Max Speed* actually reduced the average delay. However, the reduction in *Max Acceleration* caused a significant increase in average delay. This due each vehicle taking longer on average to traverse the intersection.

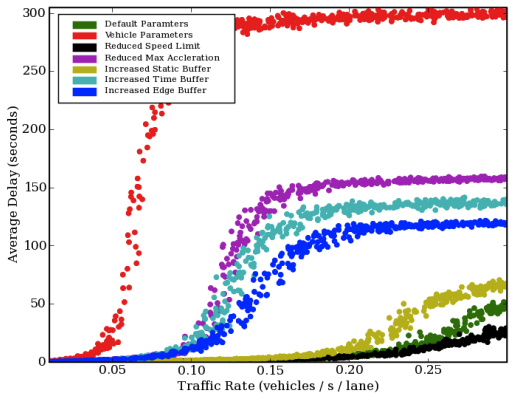


Figure 8: The effect of modifying FCFS parameters for use with the autonomous vehicle. The green points represent the default FCFS parameters. The red points are the parameters shown to work with our autonomous vehicle. The other points demonstrate the effect of changing a single parameter, for example 'Reduced Max Acceleration' is the default parameter set with *Max Acceleration* changed from 4.0 m/s^2 to 0.5 m/s^2.

The remaining three parameters: *Static Buffer Size*, *Internal Time Buffer* and *Edge Time Buffer* allow us to account for errors in the vehicles movement. In particular we can now factor in both GPS error and the velocity controller error (due to sensor noise and/or road conditions).

Based on preliminary experiments we set the buffers as follows: *Static Buffer Size* = 1.0 m, *Internal Time Buffer* = 2 seconds and *Edge Time Buffer* = 4 seconds. These values are set to be large enough to guarantee no accidents, which is more important than optimizing performance. The effect of the parameters can again be observed in Figure 8. As expected increasing any of these buffers does indeed increase the average delay of a vehicle. Increasing a buffer results in a vehicle reserving a higher percentage of the space-time tiles available inside the intersection, therefore reducing the likelihood of another reservation being granted.

Figure 9 shows that even with heavily reduced FCFS performance we still do better then the four-way stop sign currently in place at our test intersection.

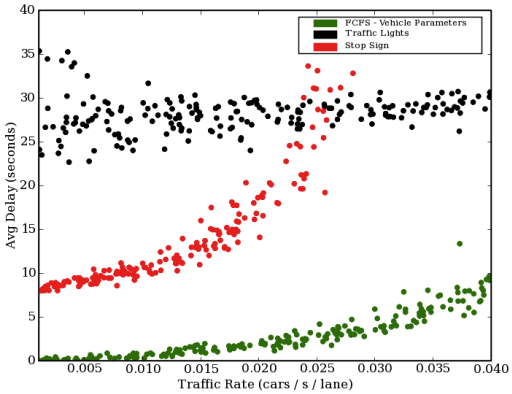


Figure 9: Comparison of the modified vehicle FCFS parameters with that of traffic signals and four-way stop signs.

The Effect of Large Buffer Sizes on Average Delay and Maximum Throughput

The mixed-reality experiments indicate that autonomous vehicles in the real world need a relatively large buffer size in order to compensate noise and errors in the vehicle's sensors and controls. An increase in static buffer size can reduce the average number of vehicles in intersections at any moment of time, thus reduce the maximum throughput of an intersection according to the Little's law. To study the effect of buffer sizes on average delay and maximum throughput, we ran an experiment with our AIM simulator with an implementation of FCFS. In the experiment, we set the traffic level of each lane to be 0.16 vehicles per second, the speed limit of the roads be 25 m/s, the maximum acceleration of the vehicles be 4 m/s^2, the internal time buffer be zero seconds and the edge time buffer be $0.25s$. We then increased the size of the static buffers from 0 to 6 meters to observe how it affects the average delay and the throughput. Figure 10 and Figure 11 show the results of these experiments.

From these figures, we can see that the average delay of the vehicles does increases as the static buffer size increases. Interestingly when the static buffer size is larger than 2 meters, the average delay suddenly increases by a factor of two. This transition point is due to the fact that a static buffer larger than 2 meters causes the width of trajectories to be larger than the width of a lane, and this prevents other vehicles from using the adjacent trajectories. In general, the average delay increases linearly with the static buffer size, expect at the transition point. The throughput (i.e., the arrival rate of the vehicles at the intersection), however, de-

[2]FCFS default parameters are: *Max Speed*, *Max Acceleration*,*Static Buffer Size*, *Internal Time Buffer*, *Edge Time Buffer* = 25.0 , 4.0, 0.25, 0.0, 0.5

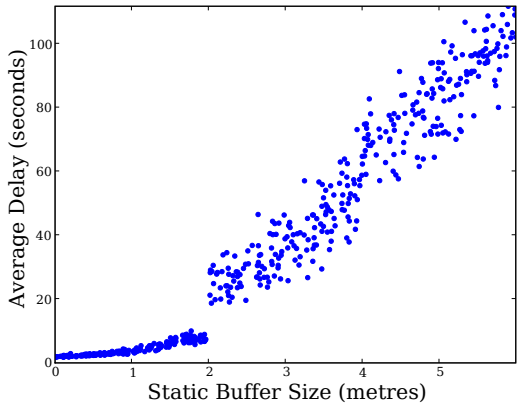


Figure 10: Static buffer size versus average delay.

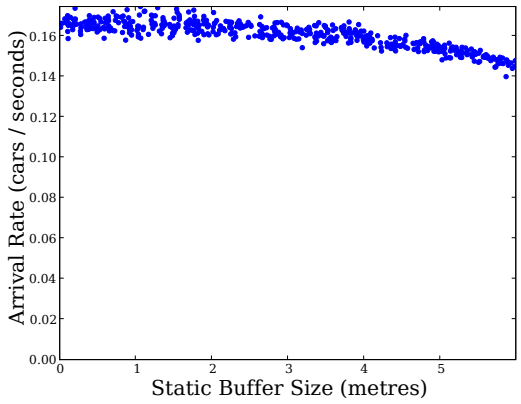


Figure 11: Static buffer size versus arrival rate.

creases moderately, because most of the vehicles eventually enter the intersection even though they have longer delays.

The results of the experiment shows that autonomous vehicles in the real world need a large buffer size, but a larger buffer size can increase the average delays and reduce the throughput of an intersection tremendously, causing traffic congestion. To compensate the effect of the large buffer size, we need to find ways to increase throughput of an intersection. The planning-based controller proposed in this paper can be used to alleviate the negative effects caused by the large buffer size.

Related Work

Intelligent Transportation Systems (ITS) is a multidisciplinary field concerns with advancing modern transportation systems with information technology (Bishop 2005). A noticeable research project on ITS is the Berkeley PATH project, which proposed a fully-automated highway system (Alvarez and Horowitz 1997). But most of the existing work on ITS focus on how to assist human drivers in the existing transportation infrastructure, and do not assume vehicles are driven autonomously by computer. Hence, most of the tools developed by transportation engineering (e.g., TRANSYT (Robertson 1969) and SCOOT (Hunt et al. 1981)) aim to optimize traffic signals rather than substitute them with a better mechanism. For intersection management, there are many work on the problem of intersection collision avoidance (Lindner, Kressel, and Kaelberer 2004; Naumann and Rasche 1997; Rasche et al. 1997; Naumann, Rasche, and Tacken 1998; Reynolds 1999; USDOT 2003). But none of these work concerns with autonomous vehicles. Balan and Luke presented a history-based traffic control (Balan and Luke 2006) that is potentially applicable to autonomous vehicles. Queueing Theory has been widely used in traffic analysis (Mannering, Washburn, and Kilareski 2008). Our analysis emphasizes how microscopic control of autonomous vehicles (via planning techniques) could affect the throughput of an intersection.

Conclusions and Future Work

The DARPA Urban Challenge in 2007 showed that fully autonomous vehicles are technologically feasible with current intelligent vehicle technology (DARPA 2007). Some researchers predict that within 5–20 years there will be autonomous vehicles for sale on the automobile market. Therefore the time is right to rethink our current transportation infrastructure, which is designed solely for human drivers. Dresner and Stone proposed to substitute traffic signals and stop signs for a new intersection control mechanism, namely FCFS, that takes advantages of the capability of autonomous vehicles, and demonstrated its effectiveness in simulation (Dresner and Stone 2008). In this paper we evaluated FCFS in a mixed reality scenario, identified assumptions in the simulation that need relaxing, and determine FCFS parameters that enable an autonomous vehicle to traverse an intersection. We explicated the relationship between the throughput of an intersection and various parameters of the intersection and vehicles via Little's law, and propose planning-based techniques to increase the throughput of an intersection. These findings allow us to implement specific improvements to our autonomous vehicle with the goal of achieving better FCFS performance. In the future, we intend to expand the mixed reality experiments to include multiple real autonomous vehicles and evaluate the system with noisy communications.

Acknowledgments

This work has taken place in the Learning Agents Research Group (LARG) at the Artificial Intelligence Laboratory, The University of Texas at Austin. LARG research is supported in part by grants from the National Science Foundation (CNS-0615104 and IIS-0917122), ONR (N00014-09-1-0658), DARPA (FA8650-08-C-7812), and the Federal Highway Administration (DTFH61-07-H-00030).

References

Alvarez, L., and Horowitz, R. 1997. Traffic flow control in automated highway systems. Technical Report UCB-ITS-PRR-97-47, University of California, Berkeley, Berkeley, California, USA.

Balan, G., and Luke, S. 2006. History-based traffic control. In *Proceedings of the International Joint Conferenceon Au-*

tonomous Agents and Multi Agent Systems (AAMAS), 616–621.

Bishop, R. 2005. *Intelligent Vehicle Technology and Trends*. Artech House.

DARPA. 2007. DARPA Urban Challenge. `http://www.darpa.mil/grandchallenge/index.asp`.

Dresner, K., and Stone, P. 2008. A multiagent approach to autonomous intersection management. *Journal of Artificial Intelligence Research (JAIR)*.

Dresner, K. 2009. *Autonomous Intersection Management*. Ph.D. Dissertation, The University of Texas at Austin.

Hunt, P. B.; Robertson, D. I.; Bretherton, R. D.; and Winton, R. I. 1981. SCOOT - a traffic responsive method of co-ordinating signals. Technical Report TRRL-LR-1014, Transport and Road Research Laboratory.

Lindner, F.; Kressel, U.; and Kaelberer, S. 2004. Robust recognition of traffic signals. In *Proceedings of the IEEE Intelligent Vehicles Symposium (IV2004)*.

Little, J. D. C. 1961. A Proof for the Queuing Formula: $L = \lambda W$. *Operations Research* 9(3):383–387.

Mannering, F. L.; Washburn, S. S.; and Kilareski, W. P. 2008. *Principles of Highway Engineering and Traffic Analysis*. Wiley, 4 edition.

Naumann, R., and Rasche, R. 1997. Intersection collision avoidance by means of decentralized security and communication management of autonomous vehicles. In *Proceedings of the 30th ISATA - ATT/IST Conference*.

Naumann, R.; Rasche, R.; and Tacken, J. 1998. Managing autonomous vehicles at intersections. *IEEE Intelligent Systems* 13(3):82–86.

Rasche, R.; Naumann, R.; Tacken, J.; and Tahedl, C. 1997. Validation and simulation of decentralized intersection collision avoidance algorithm. In *Proceedings of IEEE Conference on Intelligent Transportation Systems (ITSC 97)*.

Reynolds, C. W. 1999. Steering behaviors for autonomous characters. In *Proceedings of the Game Developers Conference*, 763–782.

Robertson, D. I. 1969. TRANSYT — a traffic network study tool. Technical Report TRRL-LR-253, Transport and Road Research Laboratory.

Squatriglia, C. 2010. Audi's Robotic Car Drives Better Than You Do. `http://www.wired.com/autopia/2010/03/audi-autonomous-tts-pikes-peak`.

USDOT. 2003. Inside the USDOT's 'intelligent intersection' test facility. Newsletter of the ITS Cooperative Deployment Network. Accessed online 17 May 2006 at `http://www.ntoctalks.com/icdn/intell_intersection.php`.