Simulation of Autonomous Vehicles: *An Analysis of Robustness to Random Stopping*

Alexander Rowden*
University of Maryland - College Park

Abstract

The simulation of traffic models is fundamental to both city planning and policy making. In order for informed decisions to be made about the inclusion of automated cars into our highways, we must learn how their inclusion will effect the safety and flow of traffic. In this work, I will be primarily focused on the safety and the ability of the models to recover from random stops. These random stops are meant to simulate unforeseeable scenarios such as pedestrians running into the road, or sudden environmental hazards.

1 Introduction

Autonomous vehicles are on their way [9]. That being said, we need to be prepared when they come. Autonomous cars have the potential to increase highway efficiency and safety but this must be validated before they can be allowed on real roads. A safe and effective way to do this would be with traffic simulations with vehicle trajectory models based on the real trajectory models of autonomous vehicles. In this work we implement a novel simulation paradigm where random obstacles are added to the road to act as environmental hazards. We then track the number of collisions, the mean waiting time and the mean travel time at varying levels of automated vehicle market penetration. The hope of this paradigm is to determine which vehicle trajectory models are the most robust to random hazards.

In this work we are dealing with a type of traffic simulation known as microsimulation [7]. Other techniques have been used to analyze traffic safety such as before-after studies [2], and Road Safety Audits/Inspections [4]. The issue with these methods is that they mainly rely on historical crash records and observation and thus cannot be done before the release of autonomous vehicles as our purpose intends. Microsimulation, however, is done by modeling the movement of individual vehicles travelling around road networks determined by car following, lane changing, and gap acceptance rules.

2 Background and Model Modification

2.1 Vehicle Following Models

The trajectory of a vehicle is decided by two factors in a simulation. It's route is determined by a traffic planner. This is typically implemented using some variant of an A-star algorithm to map each vehicle from its injection point to its target. Then, the vehicles motion, is decided by a vehicle following model. A vehicle which has departed, meaning it has been injected into the scene, determines its motion based on the nearest vehicle or obstacle in its path. Most vehicle following models put a limit on the perception ability of the vehicle. This is commonly to simulate the limited view of the driver but is also used in simulating automated vehicles to model the limitations of the vehicles sensors.

2.1.1 Modelling Human Drivers

Historically, modelling human drivers has followed the work of Krauss [5, 11] which is a stochastic variant of the Gipps model [3, 11]. The Gipps model was designed a model to reflect real vehicle conditions, with measurable driver characteristics, which would behave as expected when the interval between recalculations is the same as the driver reaction time. Due to these constraints the following formulation is made. Note, in the following equations b is defined as the deceleration constant, Δt is the constant reaction time of the driver and the minimum allowable gap is defined as s_0.

First, due to constant braking acceleration, we can define the *braking distance*, the distance necessary for the vehicle to come to a complete stop, as

$$\Delta x_l = \frac{v_l^2}{2b_l} \tag{1}$$

Where v_l and b_l are the current velocity and breaking constant of the leading vehicle. Then, the reaction time must be taken into account which means the minimum *stopping distance* is defined by

$$\Delta x = v\Delta t + \frac{v^2}{2b} \tag{2}$$

thus we must constantly maintain some distance s such that,

$$s \geq s_0 + v\Delta t + \frac{v^2}{2b} - \frac{v_l^2}{2b_l} \tag{3}$$

Thus the maximum safe speed is defined as,

$$v_{safe}(s, v_l) = -b\Delta t + \sqrt{b^2\Delta t^2 + v_l^2 + 2b(s - s_0)} \tag{4}$$

This gives us a constant solution for the velocity of a Gipps modeled vehicle as

$$v(t + \Delta t) = \min[v + a\Delta t, v_0, v_{safe}(s, v_l)] \tag{5}$$

Note that this model assumes a constant acceleration.

This, however is unsuited for our purpose as Krauss and Gipps models are mathematically collision free. Thus it is more used for vehicle flow simulation, than safety simulations. They also have fixed accelerations and thus act counter to how humans drive, as humans tend drive smoothly. This means we must adopt a new solution. For our simulation we use the Intelligent Driver Model (IDM) for modelling human driving behavior [10, 11]. We use this method as it has become the state of the art for most non-automated car simulation due to its ability to maintain realistic properties at the deterministic limit. This means it doesn't have the sharp edges that the minimum provides in Equation (5). In addition, this model is derived with the assumption that acceleration is done in most scenarios "softly" .

IDM is described by the following equation,

$$\dot{v} = a\left[1 - \left(\frac{v}{v_0}\right)^{\delta} - \left(\frac{s^*(v, \delta v)}{s}\right)^2\right] \tag{6}$$

*e-mail: alrowden@cs.umd.edu

The acceleration of IDM consists of two parts one comparing the current speed v to the desired speed v_0, and another which compares the current distance s to the desired distance s^*. The desired distance is calculated as,

$$s^*(v, \Delta v) = s_0 + \max\left[0, vT + \frac{v\Delta v}{2\sqrt{ab}}\right] \tag{7}$$

This model does have its faults. For instance, it assumes the vehicle knows the safe time gap T. This doesn't take into account the limited viewpoint of the driver. It also doesn't take into account the ability for humans to become distracted. We thus modify the vehicle trajectory model to reflect these human characteristics. Finally, some implementations, including the one in SUMO, take measures to ensure zero collisions. We thus, modify the network to bypass these characteristics. Specifically, the model will unnaturally decelerate to zero velocity if the gap is a certain size and switch to Krauss if it detects an imminent collision. We then call this model LIDM for Less Intelligent Driver Model.

2.1.2 Modelling Automated Vehicles

To simulate automated vehicles we use the Cooperative Automated Cruise Control model [1, 11]. This model is the current state of the art for simulating automated vehicles in academic works as true fully automated car algorithms are still being developed and are proprietary IP. CACC was derived from another model called Automated Cruise Control or ACC [8,11]. ACC is a system designed such that drivers maintain a desired cruise speed as long as the preceding vehicle maintains a specified distance away called the following gap. It uses the *Constant Acceleration Heuristic* or CAH, to predict the movements of the preceding vehicle which is defined as,

$$a_{CAH}(s, v, v_l, \dot{v_l}) = \begin{cases} \frac{v^2 \tilde{a}_l}{v_l^2 - 2s\tilde{a}_l} & \text{if } v_l(v - v_l) \leq -2s\tilde{a}_l \\ \tilde{a}_l - \frac{(v-v_l)^2 \theta(v-v_l)}{2s} & \text{else} \end{cases} \tag{8}$$

The ACC vehicle determines the distance to another vehicle by simulating a LIDAR or radar sensor. This model however, lacks the communicative ability of the fully autonomous vehicles we will be seeing emerge in the coming years. Due to the emergence of Dedicated Short Range Communication (DSRC), a vehicle can exchange information with surrounding vehicles and why the CACC model was developed to utilize the vehicle's access to the surrounding vehicle's intent, position, velocity and other such parameters. Thus the ACC model is defined as follows,

$$a_{ACC} = \begin{cases} a_{IDM} & \text{if } a_{IDM} \geq a_{CAH} \\ (1-c)a_{IDM} + & \\ c\left[a_{CAH} + b\tanh\left(\frac{a_{IDM} - a_{CAH}}{b}\right)\right] & \text{else} \end{cases} \tag{9}$$

This equation balances the CAH with the IDM model. This is done because the ”pure” ACC model, as it believes in perfect reaction time, would leave too small a gap and thus drive ”recklessly”. The c parameter in equation (9), is known as the *coolness term*. It is used to balance the two models and is typically chosen to equal .99.

CACC is built on top of ACC as you can see by Figure 1. This state machine defines the basic logic of CACC. The logic of CACC is defined by the identity of the preceding car. If the preceding car is also controlled by CACC, the logic is defined by the CACC model, however if it is following any other vehicle, the CACC vehicle falls back to an ACC model as it cannot communicate with the vehicle. Otherwise it uses its specialized CACC logic. The CACC vehicle determines its control mode by calculating the time gap to the vehicle it is following. Since model can communicate with car ahead of it, it can know not only the car's position, as an ACC model would use, but it can also get the vehicle's motion plan to make a better estimate of the vehicle's position in the next time-step. When the gap is calculated the CACC controller chooses a mode based off of a time-gap threshold. If the gap is under the threshold, the controller uses gap regularization, where they apply emergency breaking to avoid collision. If the gap is over the threshold, the controller applies gap closing, where the vehicle attempts to approach it's ideal cruising speed.

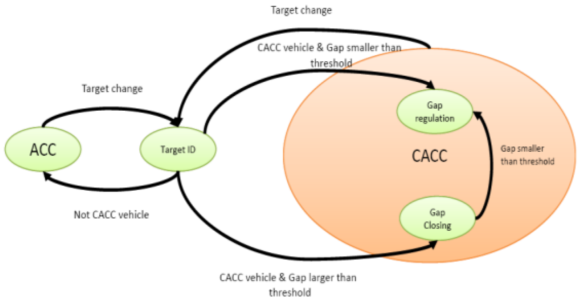
Target change
CACC vehicle & Gap smaller than threshold
Target change
ACC
Target ID
Gap regulation
CACC
Gap smaller than threshold
Gap Closing
Not CACC vehicle
CACC vehicle & Gap larger than threshold

Figure 1: The state machine of a CACC controller. From [1]

I chose Cooperative Automated Cruise Control as it can simulate communication between vehicles, through singular connections, and simulates smooth acceleration which aids in traffic alleviation and driver comfort which will be a priority for automated car manufacturers. This model contains an implicit ranging function. The base implementation of this model has the detection range of the vehicle set to a constant parameter. In other words,

$$\mathbf{x_p}(\mathbf{x}) = \begin{cases} x & \text{if } x < threshold \\ inf & \text{else} \end{cases} \tag{10}$$

This means that the vehicle will only be able to detect a vehicle up to a certain distance away. This is a physically-based constraint as LIDAR and radar can only sense so far accurately.

This model is also slightly modified for our purposes to simulate the variability of sensor data. For each time-step, a random penalty is placed on the visibility range of the cars sensors. This is done similarly to the variable sight implemented in 2.1.1 except instead of randomizing the sensor penalty when the car is loaded, we calculate the penalty every time-step. This is to simulate sensor error and is implemented as follows,

$$\mathbf{x_p}(\mathbf{x}) = \begin{cases} x & \text{if } x + \mathbf{r} < threshold \\ inf & \text{else} \end{cases} \tag{11}$$

where, $0 < \mathbf{r} < \varepsilon_{sensor}$ and ε_{sensor} represents the range of sensor error.

3 Experimental Design

3.1 Frameworks

For this project I used the Simulation of Urban MObility (SUMO) [6] framework for my simulations. SUMO provides a structure for a simulation paradigm and provides modules for each step of the simulation while allowing customization through various inputs, and overriding classes. For the purposes of my simulation I used SUMO's scheduler, and Junction handling modules as they were not the focus of this research. We however modify the vehicle following models as specified in 2.1.1 and 2.1.2. I also utilized their tool for generating SUMO traffic networks from map data called OSMWebWizard. I also utilized the open source extension to SUMO called TraCI, Traffic Controller Interface which allowed for real time input to SUMO. This is how I was able to add obstacles on the fly.

3.2 Experiments

The purpose of this simulation was to analyze two vehicle types; one representing driven cars, LIDM; and another representing automated cars with communication CACC with modification. I then wanted to analyze their robustness to sudden stops so I used TraCI to inject random obstacles (almost stationary non-CACC vehicles) to act as environmental hazards. For the control experiment I ran the simulation at different automated vehicle market penetrations with no stops to set a baseline. I then fixed the market penetration to 50-50 and changed the probability of random stops as another baseline. Finally, I ran another sweep over market penetrations with a fixed non-zero probability of generating obstacles at each time-step. I chose this to be a low probability, .01, so that the highway doesn't get backed up so much that no real results can be seen. For a clip depicting an example of the final experiment see here. As a note each simulation sweep ran 11 simulations of around 1700 vehicles with randomly generated routes. Each simulation ran for around half an hour (making each sweep about 6.5 hours long) on a Windows Computer running a Linux Subterminal on an Intel i7-6800K CPU running at 3.4Ghz.

4 Results

As can be seen from the first baseline provided in Figure 2 the automated cars are more prone to collision. This was an expected outcome as Automated cars don't encode a strict distance gap like the driver model and as such it is more prone to collisions in day to day traffic, traffic without sudden stops. This baseline also showed that the automated cars were in general faster in making it to their destination. This is likely due to the fact that they drive more aggressively in general and can make better following decisions when behind another self driving car to increase efficiency of not only their own trip but the trips of others as well.

From the second baseline shown in Figure 3 we see an obvious result and a bit less obvious of a result. Firstly, the number of collisions increased dramatically with the number of stops. This was not surprising as the obstacles were designed to cause collisions. These collisions are placed in locations that violate the stopping constraints of the vehicle models – they could be placed within the distance gap of LIDM or within the time gap specified by CACC. A less expected result however, was that as the probability of obstacles increased, the mean travel and wait times didn't increase or decrease. I attribute this to the network becoming fully congested and thus traffic entered it's worst case scenario.

For the experimental results in 4 we see something a bit unexpected. As a reminder this test was done with a set likelihood of adding an obstacle in each time step sweeping through market penetrations. As can be seen, the number of collisions increases as in the baselines. In addition, the mean wait time lost its clear downward trend from benchmark one. Going into this experiment I assumed the differences between models would be exaggerated but the data shows the opposite. I attribute this due the model getting fully congested and thus fewer collisions happen overall and wait times become uniform between market penetrations.

5 Limitations

The largest limitation of this work is in the vehicle following models. Specifically, IDM is a simplification of human driving. This is minimized by the introduction of randomness in the design and by our additional changes but this does not fully encapsulate human driving. Another limitation which was stated in 2.1.2 is that the true trajectory planning involved in autonomous vehicles is proprietary IP and thus not accessible for use in this simulation. Further, the randomness involved in these experiments was limited by simulating a long period of time however, it would be best to run the simulation multiple times and average out the runs.

6 Conclusion and Future Work

It is clear that improvements need to be made to automated cars before they can be allowed to enter the dynamic environment that a highway poses but we also must improve our methods of simulating these vehicles such that proper policies can be put in place. Future work should look into how an autonomous car would handle lane changing as both of these vehicle models are designed for single lane use and modified on the fly to change lanes which may be a source of error. Still, this framework can easily be developed further and modified to test future automated car models to develop confidence in these vehicles before mass market availability. If I were able to run this simulation suite one more time I would change the vehicle density. In my original tests, without random stops, the vehicles were so far apart they never collided and no discernible difference between models could be seen, but I appear to have gone too far. A future work could explore the effect of traffic density on these results as well.

Acknowledgments

This work was done for CMSC828X: Physically Based Modelling, Simulation and Animation under the direction of Professor Ming C. Lin.

References

[1] F. Bu, H. Tan, and J. Huang. Design and field testing of a cooperative adaptive cruise control system. In *Proceedings of the 2010 American Control Conference*, pp. 4616–4621, June 2010. doi: 10.1109/ACC.2010.5531155

[2] R. Elvik. The importance of confounding in observational before-and-after studies of road safety measures. *Accident Analysis & Prevention*, 34(5):631–635, 2002.

[3] P. Gipps. A behavioural car-following model for computer simulation. *Transportation Research Part B: Methodological*, 15(2):105 – 111, 1981. doi: 10.1016/0191-2615(81)90037-0

[4] M. Hoque. Road safety audit in developing countries. *Transportation Research Group*, 1997.

[5] S. Krauss. Microscopic modeling of traffic flow: investigation of collision free vehicle dynamics, Apr 1998.

[6] P. A. Lopez, M. Behrisch, L. Bieker-Walz, J. Erdmann, Y.-P. Flötteröd, R. Hilbrich, L. Lücken, J. Rummel, P. Wagner, and E. Wießner. Microscopic traffic simulation using sumo. In *The 21st IEEE International Conference on Intelligent Transportation Systems*. IEEE, 2018.

[7] S. S. Mahmud, L. Ferreira, M. S. Hoque, and A. Tavassoli. Micro-simulation modelling for traffic safety: A review and potential application to heterogeneous traffic environment. *IATSS Research*, 43(1):27 – 36, 2019. doi: 10.1016/j.iatssr.2018.07.002

[8] G. G. Naus, J. Ploeg, R. van de Molengraft, and M. Steinbuch. Explicit mpc design and performance-based tuning of an adaptive cruise control stop-&-go. *2008 IEEE Intelligent Vehicles Symposium*, pp. 434–439, 2008.

[9] S. L. Poczter and L. M. Jankovic. The Google Car: Driving Toward A Better Future? *Journal of Business Case Studies (JBCS)*, 10(1):7–14, 2014. doi: 10.19030/jbcs.v10i1.8324

[10] M. Treiber, A. Hennecke, and D. Helbing. Congested traffic states in empirical observations and microscopic simulations. *Physical Review E*, 62(2):1805–1824, Aug 2000. doi: 10.1103/physreve.62.1805

[11] M. Treiber and A. Kesting. Traffic flow dynamics. *Traffic Flow Dynamics: Data, Models and Simulation, Springer-Verlag Berlin Heidelberg*, 2013.

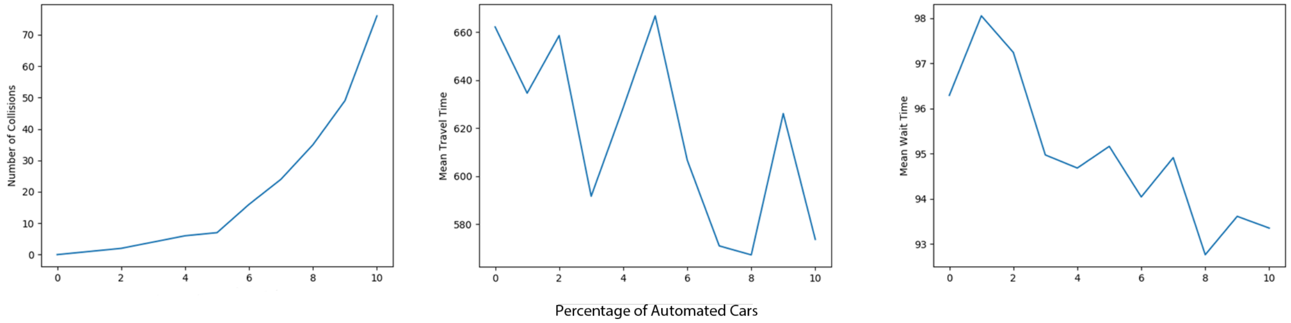

Figure 2: Results from baseline number one. Left: Number of collisions as the market penetration of automated cars increases. Middle: Average travel time as the market penetration of automated cars increases. Right: Average time waiting as the market penetration of automated cars increases.

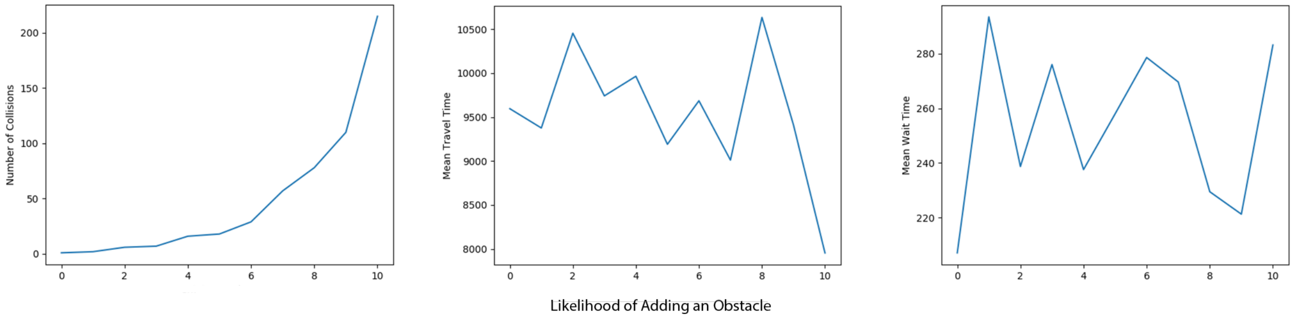

Figure 3: Results from baseline number two. Left: Number of collisions as the likelihood of adding a random stop each frame increases. Middle: Average travel time as the likelihood of adding a random stop each frame increases. Right: Average time waiting as the likelihood of adding a random stop each frame increases.

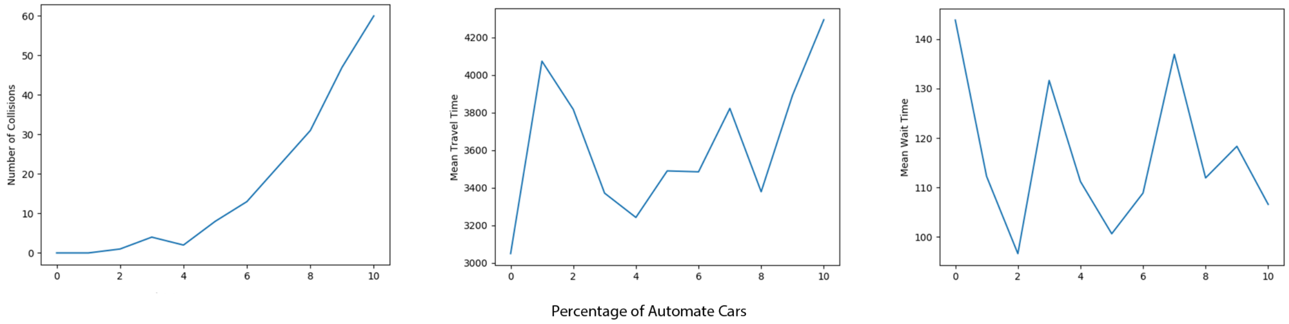

Figure 4: Results from the test sweep. Note this test was run with a constant probability of adding a random stop set to 1 percent. Left: Number of collisions as the market penetration of automated cars increases. Middle: Average travel time as the market penetration of automated cars increases. Right: Average time waiting as the market penetration of automated cars increases.