Using a Monte Carlo Simulation to Investigate Road Throughput

Harry P. Johnson [1]★
[1] *University of Southampton, University Road, Southampton SO17 1BJ, UK*

21 August 2025

ABSTRACT
In a world where many people journey using cars, it is important that we make sure as many people can travel as possible. We want to make sure that we can maximise the the amount of cars on the road whilst maximising the flow - the amount of cars passing through the road at any time, called throughput. This paper uses a Monte Carlo simulation to investigate how traffic flow varies with the number density of cars in a simple road model, where cars have a random chance to slow down to emulate errors in driving, or dangerous roads. By varying the simulations parameters such as the maximum car velocity, probability of slow down and car density, we discover that having a lowered slow down chance and specific values of car density allow for larger throughputs.

1 INTRODUCTION

Effective transport is important for our current day society, especially with many people travelling large distances to and from work every day. For these people, it is necessary that our roads allow for as many people to travel as efficiently as possible over these distances. Therefore, an important question to answer for engineers is how to move the maximum amount of cars through a given road in a certain time. We may investigate this question using Monte Carlo simulations. Monte Carlo experiments are computations that use random number generation to answer questions which may be deterministic. An example is numerical integration - say we have a curve of unknown area. Next, we draw a box of known area encompassing the curve. Next, we uniformly generate pairs of points within the box, and record whether or not these points land inside the curve. Using the percentage of points that land inside the curve compared to the total points generated, we can calculate an approximate area of the curve. Monte Carlo simulations are a modern invention - they require large amounts of random number generation, a task that requires the use of computers. The beginning of these simulations can be attributed to Stanislaw Ulam in the 1940's whilst he was working on nuclear weapon projects, an account of which can be read here Metropolis (1987). In this paper, I will use randomness to simulate the movement of cars. I will randomly generate positions for cars on a road, then allow them to move via a set of rules. One of these rules includes a random change for the cars to slow down, simulating a dangerous road where cars may slow down unexpectedly, or due to human error when driving. Many simulations will be done to investigate a variety of parameters of simulation, the main one being the amount of cars on the road so that I may investigate how car density effects the throughput of a road. Other parameters include the maximum speed of the road and the probability of cars randomly slowing down.

2 CREATING THE SIMULATION

2.1 Assumptions

There are two main approaches to modelling a traffic simulation. The first is treating the cars as if they were a liquid, in a hydrodynamic-esque model. In this paper, I will be using a model analogous to kinetic theory where each car is treated individually. The assumptions for the simulation are as follows -

- There is N cars on a long road with L spaces, each of which may contain 1 or 0 cars.
- The cars will have a maximum velocity V_{max}.
- The cars will have velocities 0 to V_{max} in integer steps.
- The road's ends will be joined, treating it as a circle.

At the start of the simulation, the road will be randomly populated with N cars, each starting with a velocity of 0. The velocities used in this simulation aren't related to any physical velocities and as such are only being used to comparatively to each other within the bounds of the simulation.

2.2 Car Logic

In order for the simulation to function, a set of rules for determining what a car does at each time step must be created. At each time step, the following rules will be used to update the cars position in the order specified.

(i) The car accelerates. It's velocity v will be increased to $v+1$ up to a maximum of V_{max}.
(ii) A check will be performed to see if the car will hit the car in front. If there is a car a distance d in front, and $v \geq d$ then $v = d - 1$. This ensures cars won't hit the car in front.
(iii) The car has a random chance p to slow down. This is used to emulate external events which may cause a driver to slow down or human error. A larger value of p may signify worse driving, or perhaps a more dangerous road. If a car is to randomly slow down its velocity is reduced to $v - 1$, provided its velocity is not already zero.
(iv) The car is moved forward v spaces.

Care must be taken when updating the positions of the cars. In code, I will be going through the road one slot at a time, in increasing order. This poses a problem - moving a car at the start of the road may cause a car at the end of the road to move to a space that it shouldn't, as I am treating the road as a circle. For example, take a road with 10 spaces. Now say that there is a car at position 1 with velocity 3, and a car at position 10 with velocity 2. Going through the positions in ascending order, we update the car in position 1 to position 5. This allows the car in position 10 to move into position 3. This produces different behaviour compared to having the cars in, for example, positions 2 and 3 where the car in position 2 wouldn't move. To solve this problem, at the start of every loop over the road I will create a copy of the list and use that list to update the positions, with the updated positions being stored in a different list. This effectively makes it such that all cars are updated at the same time, solving the problem.

2.3 Collecting Data

2.3.1 Number Density

The number density n is defined as $n = \frac{N}{L}$ where N is the amount of cars, and L is the length of the road. To change the number density, I alter the amount of cars on the road whilst keeping the length constant. The reasons for this are twofold -

- Altering length causes number density to scale as $n \propto \frac{1}{L}$. Since both N and L are integers, this leads to requiring a very large L for a reasonable amount of low number density values which would vastly increase computation time.
- In this simulation, the road has a single lane. Increasing the length of roads in reality would require large scale restructuring of roads which isn't feasible. Various other problems would also be caused, for example longer travel times, increased fuel consumption and others. I have chosen to use a road length of 400 and will vary the amount of cars from 10 to 390.

2.3.2 Calculating Throughput

In this simulation, I am looking to maximise throughput by by varying the number density of cars on the road. To do this, I will record whenever a car passes a specific place on the road for each time step. At the end, I will calculate the mean cars passed per time step, and save this value to a file.

For an accurate mean, a statistically steady state is required for each number density of cars. As a result, I will need to run the simulation for a large amount of time steps for each number density. I chose an amount of time steps equal to 100 times the length of the road, with a road length of 400.

2.3.3 Efficiency

As described in section 2.3.2, lots of data is needed for . This means lots of calculation time, so I have made efforts to improve the efficiency of data collection.

The main method I have used to reduce simulation time is to measure cars passing at multiple places throughout the road. This works well, as long as the places are well spaced apart. I used 4 places on the road of length 400 - at positions 100, 200, 300, and 400. This means I can quarter the amount of time steps. Using my chosen amount of time steps of 100 times the length results in 40000 time steps being reduced to 10000. This quarters the computation time. Even with this improvement, the simulation still took an hour to complete.

I also investigated an alternate method of reducing time, particularly for high number densities. My chosen method of filling the road with cars was beginning with an empty list, and randomly generating positions for cars, and rejecting if a position is already full. If the road is already very full, this results in many position generations until an empty one is found. It would be more efficient for the list to start as full and to remove cars for lists that are more than half full. After investigating the time taken to fill the lists for these methods I found the time difference to be negligible, so I chose not to implement this for simplicity.

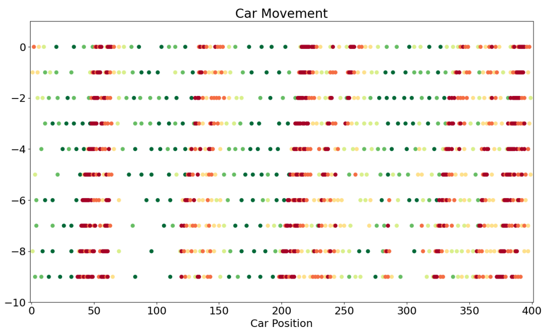

Figure 1. Still frames of the simulation. Time increases downward with each consecutive road being 3 time steps later than the last to allow the traffic jams to be more visible. The road is length 400, with 100 cars on the road, a slow down probability of 0.25, and a max velocity of 5. The velocities are signified via colours, with dark green representing the highest velocity, dark red 0 velocity, and yellow middling velocity.

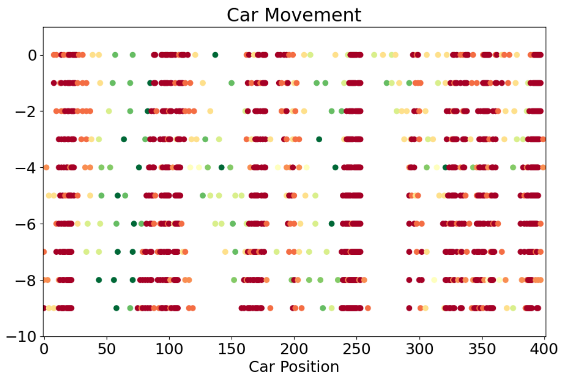

Figure 2. Still frames of the simulation. Time increases downward with each consecutive road being 3 time steps later than the last to allow the traffic jams to be more visible. The road is length 400, with 100 cars on the road, a slow down probability of 0.75, and a max velocity of 5. The velocities are signified via colours, with dark green representing the highest velocity, dark red 0 velocity, and yellow middling velocity.

3 RESULTS

3.1 Qualitative Properties of the Simulation

In figure 1, the movement of the cars over time can be clearly seen. The traffic jams slowly recess back along the road, which can be clearly seen at car position 225, and at the locations of the other red dots. A mini traffic jam being formed by a car slowing down is also clearly visible. From row 5 to 6, you can see an orange dot appear around car position 280. In the following rows, red and orange dots can be seen appearing in this location, signifying the start of a traffic

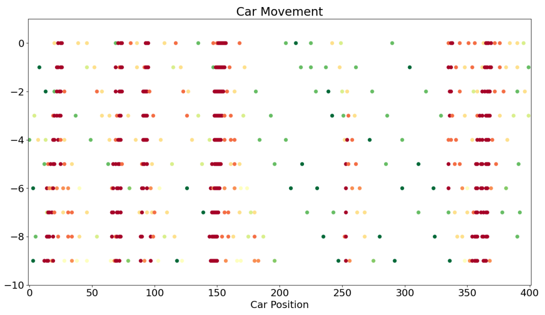

Figure 3. Still frames of the simulation. Time increases downward with each consecutive road being 3 time steps later than the last to allow the traffic jams to be more visible. The road is length 400, with 50 cars on the road, a slow down probability of 0.75, and a max velocity of 5. The velocities are signified via colours, with dark green representing the highest velocity, dark red 0 velocity, and yellow middling velocity.

jam. This behaviour mimics reality - traffic jams form around slower moving cars.

Increasing the slow down probability while keeping every other variable constant causes more traffic jams to occur, shown in figure 2, as would be expected in reality.

In figure 3 the beginnings of a traffic jam are shown. Here, I have decreased the amount of cars and increased the slow down probability. Around car position 250, and starting in row 3 you can clearly see cars of high velocity approaching the slower car and forming the start of a traffic jam.

3.2 Throughput Against Number Density

I have calculated the throughput against number density for a variety of different max velocities and slow down probabilities. The throughput was calculated at 4 individual points on the road so to convert it to that through a single point I have divided it by 4.

3.2.1 *P* = 0.25

In figures 4 and 5, you can see how throughput changes with number density. Here I have used a slow down probability of p=0.25 and maximum velocities of 3, 5, 7, and 9. All velocities have the same rough shape - a linear increase up to a peak value, then a slight downward curve after. Velocities $v = 7$ and $v = 9$ have a sharp peak in throughput - 0.6 and 0.64 cars per time step at densities 0.09 and 0.075 respectively. A small peak can be seen for $v = 5$ of 0.53 cars per time step at number density 0.12. $v = 3$ has a very different looking curve - there is no peak in sight. It has a relatively smooth transition between its linear increase to curved decrease. On close inspection all graphs appear slightly noisy on the descent of throughput after their respective peaks. At number densities of approximately 0.3 and higher the throughputs for each velocity converge.

3.2.2 *P* = 0.5

There are small differences in the shapes of the curves for p = 0.5 compared to p = 0.25, as shown in figures 6 and 7. In addition, the throughput for all velocities is lower than that of p=0.25. Both $v = 7$ and $v = 9$ have large peaks again (of 0.42 and 0.42 cars per time step). $v = 5$ has a slightly larger spike compared to p=0.25 as well.

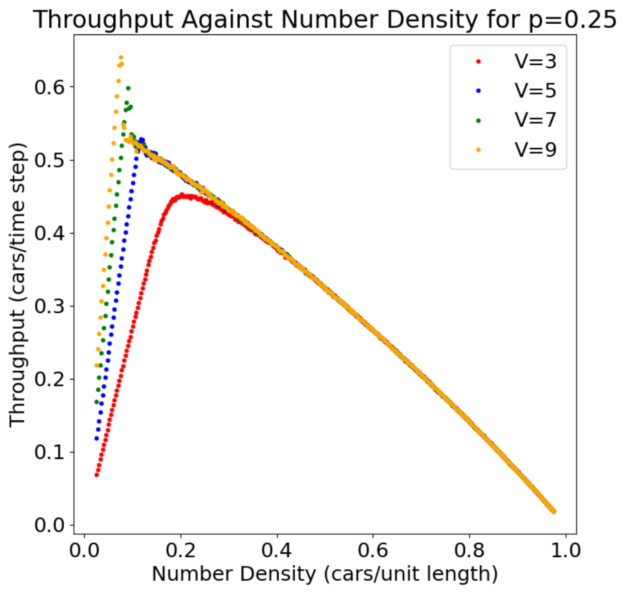

Figure 4. Graph of throughput against number density. The road is length 400, with 50 cars on the road, a slow down probability of 0.25, and max velocities of 3,5,7, and 9 with colours shown in the key.

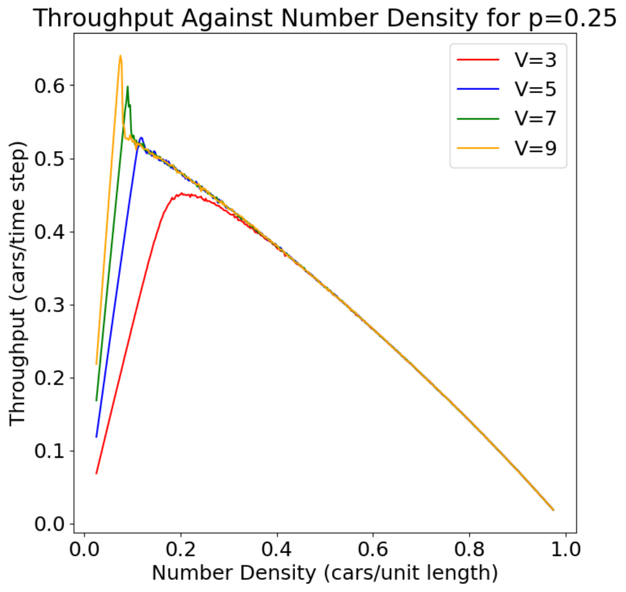

Figure 5. Graph of throughput against number density. The road is length 400, with 50 cars on the road, a slow down probability of 0.25, and max velocities of 3, 5, 7, and 9 with colours shown in the key. This plot is the same as that in 4 except it is a line graph.

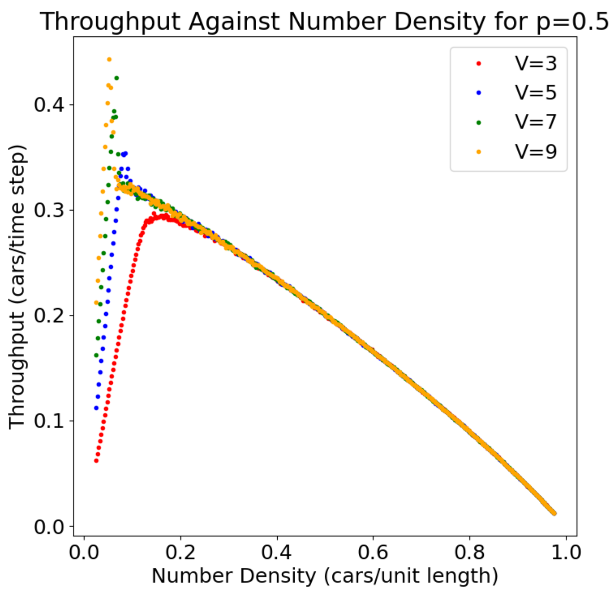

Figure 6. Graph of throughput against number density. The road is length 400, with 50 cars on the road, a slow down probability of 0.5, and max velocities of 3,5,7, and 9 with colours shown in the key.

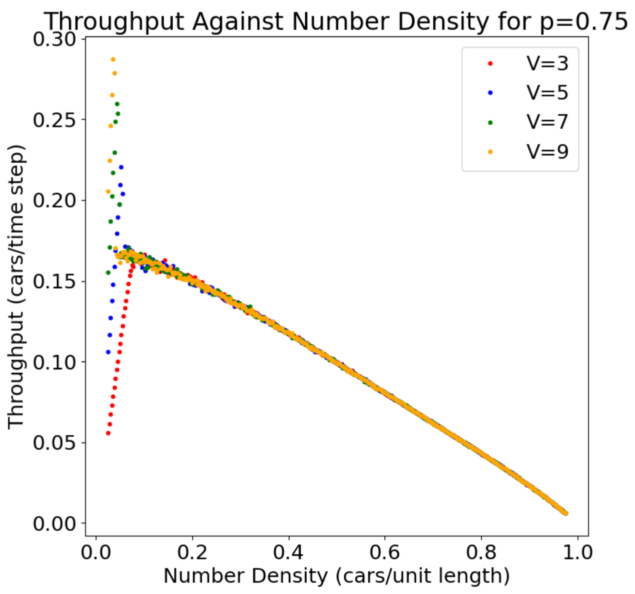

Figure 8. Graph of throughput against number density. The road is length 400, with 50 cars on the road, a slow down probability of 0.75, and max velocities of 3,5,7, and 9 with colours shown in the key.

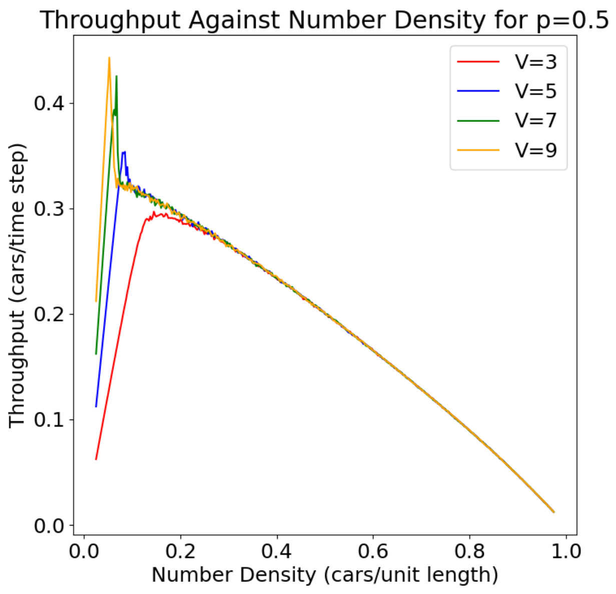

Figure 7. Graph of throughput against number density. The road is length 400, with 50 cars on the road, a slow down probability of 0.5, and max velocities of 3,5,7, and 9 with colours shown in the key. This plot is the same as that in 6 except it is a line graph.

The number densities of the peaks are much lower for this slow down probability too - 0.053, 0.068, 0.085, and 0.145 for $v = 9$, $v = 7$, $v = 5$, and $v = 3$ respectively. The point at which all velocities converge also happens at a lower number density - approximately 0.25. Another slight difference can be seen for $v = 3$, where the transition between increase and decrease is much less smooth. In fact, all curves appear more noisy than those for p=0.5.

3.2.3 P = 0.75

For p = 0.75 (figures 8 and 9), the changes are more notable, specifically for $v = 3$. The smooth transition no longer exists - instead, it has a sharp corner, shown in figure 10. It bears similarity to that of the peaked velocities, only it has no peak. Other than this, the same things that changed from p = 0.25 to p = 0.5 occur here as well. The peaks become more substantial and move to lower number densities, the overall throughput decreases, the curves become even more noisy, and the number density of convergence lowers.

3.3 Errors

I have excluded the standard deviation of throughput from the graphs. When calculating the throughput the range of possible values per time step is small - it can only take values of 0, 1, 2, 3, and 4. In addition, due to the random probability of slow downs it is not unlikely for each of these values to occur many times. As a result, I get large standard deviations which cover large amounts of the graph making the detail hard to resolve. Therefore when discussing the qualities of the graphs, it is useful for me to not include the standard deviation. The graphs including standard deviations are shown in figures 11, 12, and 13.

To reduce the errors, I could run the same simulations many times with different car starting positions. However, I haven't been able to

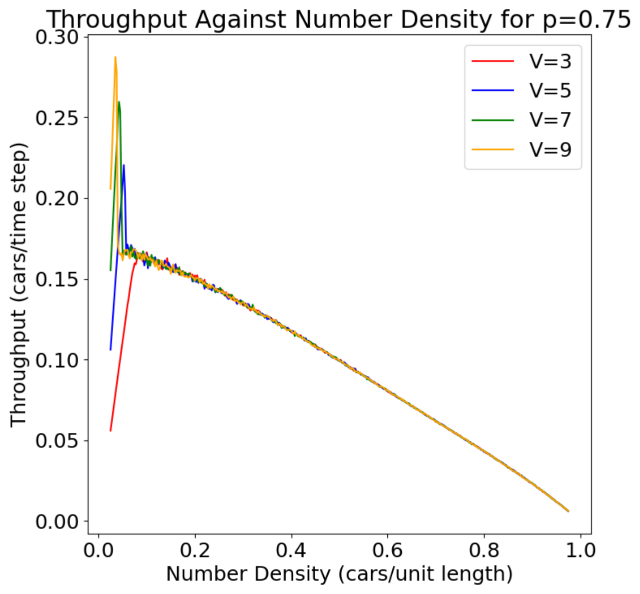

Figure 9. Graph of throughput against number density. The road is length 400, with 50 cars on the road, a slow down probability of 0.75, and max velocities of 3,5,7, and 9 with colours shown in the key. This plot is the same as that in 8 except it is a line graph.

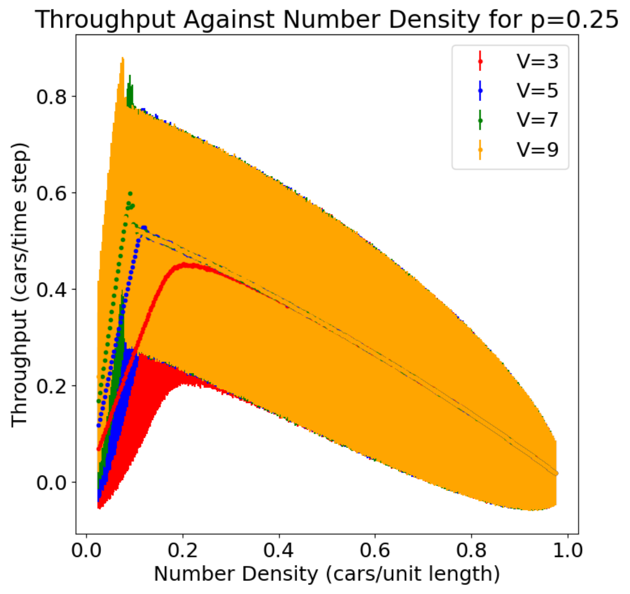

Figure 11. Graph of throughput against number density. The road is length 400, with 50 cars on the road, a slow down probability of 0.25, and max velocities of 3,5,7, and 9 with colours shown in the key. This graph includes the standard deviation.

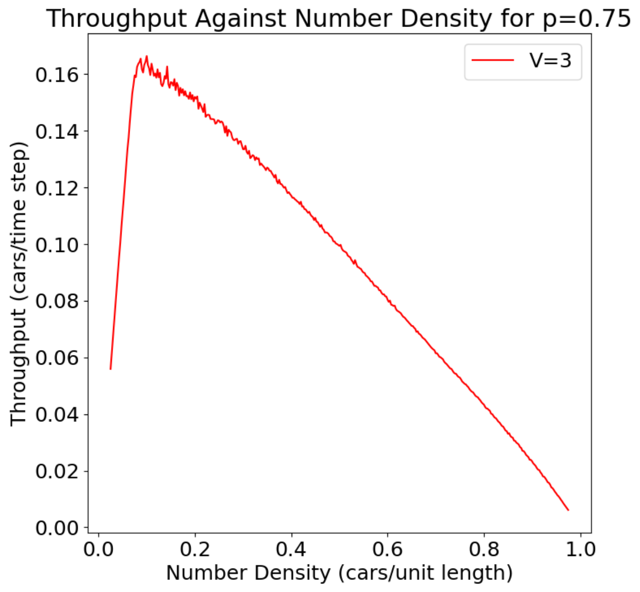

Figure 10. Graph of throughput against number density. The road is length 400, with 50 cars on the road, a slow down probability of 0.75, and a max velocity of v = 3.

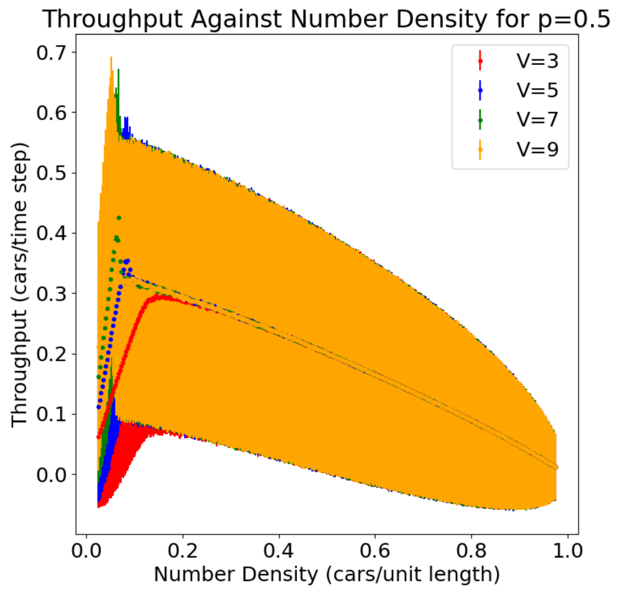

Figure 12. Graph of throughput against number density. The road is length 400, with 50 cars on the road, a slow down probability of 0.5, and max velocities of 3,5,7, and 9 with colours shown in the key. This graph includes the standard deviation.

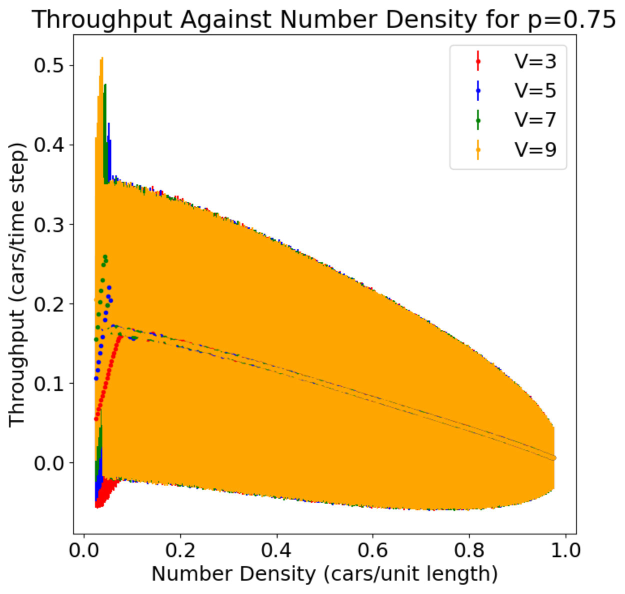

Figure 13. Graph of throughput against number density. The road is length 400, with 50 cars on the road, a slow down probability of 0.75, and max velocities of 3,5,7, and 9 with colours shown in the key. This graph includes the standard deviation.

Table 1. Table of peak throughputs for p=0.25.

Velocity	Peak Throughput	Number Density of Peak
9	0.64	0.075
7	0.60	0.090
5	0.53	0.12
3	0.45	0.22

Table 2. Table of peak throughputs for p=0.5.

Velocity	Peak Throughput	Number Density of Peak
9	0.44	0.053
7	0.42	0.068
5	0.35	0.085
3	0.30	0.145

do so due to time constraints. Each run takes approximately an hour to complete. Say I wanted a reasonable amount of data points for a good mean for each combination of slow down probability and velocity, for example 20 runs per combination. I have 12 combinations of slow down probability and velocity which would result in 240 hours of computation time. This is not reasonable for me to do. I would rather have the variety of runs I have so I can explore how different variable changes effect throughput opposed to one or two different combinations with small errors, not allowing any solid conclusions.

3.4 Table of Values

The following are tables with peak throughput values for each velocity and slow down probability, for future reference.

Table 3. Table of peak throughputs for p=0.75.

Velocity	Peak Throughput	Number Density of Peak
9	0.29	0.035
7	0.26	0.043
5	0.22	0.053
3	0.17	0.1

4 DISCUSSION

It is clear from the results that when maximising throughput it is very important to have a low number density, especially for high speed limits where there is a large spike in throughput. Another important factor is to reduce the probability of random slow downs where possible, as random slow downs significantly reduce overall throughput. Comparing the maximum throughputs for p=0.25 and p=0.5, we can see an overall decrease in throughput of the maxima of about 32%, as seen in table 4. There is little we can do to stop slow downs due to the roads themselves, eg: traffic lights, speed limits due to road works, and corners. There is one thing we may do in particular however, which is more stringent driving tests. If driving tests require a higher quality of driving, then we can reduce the probability of accidents and slow downs due to negligent driving.

For the number density, it is important to target a specific number density based on the speed limits of the road. From the graphs in section 3, we can see large spikes in throughput at precise number densities. A caveat to this is that at the point where throughput is highest, fluctuations are largest. A small change in the conditions can result in large changes to the results. The transition between free flowing traffic and traffic jams (seen in figures 5, 7, and the like, where the graph moves from a linear increase to the sloped decrease) is commonly regarded as analogous to a phase transition. Phase transitions are well known to house large fluctuations. Because of this the precise point where throughput is at a maximum cannot be well determined unless you have a large amount of number densities, more than what I have calculated.

Regarding these fluctuations, it seems like the low maximum velocity and low slow down probabilities largely avoid this problem, as can be seen in 5. A possible reason for this may be that because cars are so slow, they largely avoid getting into traffic jams. Another reason could be that the traffic jams are much smaller here compared to higher velocities, so they have a much smaller effect on the throughput. In figure 14, you can see this. I have chosen a number of cars at the approximate peak number density (80 cars, number density = 0.2). The jams are few and far between, and are also very small. This becomes more apparent when comparing it to figure 15. While there is only one traffic jam here, it contains many less cars (32 compared to 80) but has just as many cars in that single jam. This jam is clearly highly destructive to the stable traffic flow, compared to figure 14 where it is not. This could be a cause of the large fluctuations.

At low number densities we have a clear, linear increase in throughput. This can be explained by there being few cars on the road. At these low amounts of cars there aren't enough cars to form large traffic jams, so adding more cars simply increases the throughput proportional to the amount of cars on the road. As the peak throughput is reached, traffic jams will form. Even though there are jams the throughput is higher simply due to there being a larger amount of cars on the road. Eventually a point is reached where adding more cars causes too many jams and the throughput decreases.

For high number densities, the throughput appears the same regardless of maximum velocity. This has a simple explanation. When

Table 4. Table of peak throughputs for p=0.25 and p=0.5, along with the percentage decrease between the two.

Velocity	p = 0.25	p = 0.5	% Decrease
9	0.64	0.44	31.3
7	0.60	0.42	30.0
5	0.53	0.35	34.0
3	0.45	0.30	33.3

there are lots of cars, there are many cars close to each other. Since lots of cars will be directly behind another they have no chance to accelerate to a high velocity and thus stay at low velocities. Thus the maximum velocity they can reach is not limited by a speed limit, rather the amount of cars. Therefore at high densities and at all maximum velocities the traffic acts in the same manner so their throughputs converge. This is explicitly shown in figures 16, 17, and 18.

It should be noted that the exact values for velocity, road size, and the values calculated do not have direct real world equivalents (as in, what does a velocity of 5 in this simulation equal in the real world?). What is important however, is how changing these values changes the outcome of the situation as this is what would happen in reality.

The best way to increase the overall throughput is clear - reduce the amount of random slow downs drivers make. This could be done by making roads safer (for example improving road quality via the fixing of pot-holes and the like) or by imposing stricter driving tests to reduce human error. Other methods are possible, for example the one examined in Aldegheishem et al. (2018). Reducing the number density is also very important - this can be done by reducing the amount of cars on the road. Ways of doing this could include improving public transport so people are more likely to travel by bus or train, constructing more cycle lanes and providing benefits for cycling, and providing adequate foot paths to allow for easy access to nearby places of interest. Financial incentives could be used to promote more active travel methods as explored in Martin et al. (2012). Increasing the speed limit could also increase throughput - however this might not be suitable as increasing the speed limit could be liable to cause more road accidents. To increase the throughput for already set speed limits, we should instead reduce the amount of slow downs and number density as already outlined.

5 CONCLUSIONS

In this paper, I have simulated a simple road with cars moving to investigate how to maximise the the throughput while varying the road conditions. The cars try to accelerate whenever possible, slowing down to avoid crashing into cars in from and forming traffic jams. The cars also have a random chance to slow down slightly which emulates dangerous driving conditions, or perhaps incompetent driving. In a world where many people travel by car, it is important to reduce congestion and maximise throughput. By changing the amount of cars on the road, maximum velocity of cars, and the chance of random slow down I have created graphs showing how throughput changes with variations of these parameters. There is a large decrease in throughput overall when increasing the chance of slow down, which may be solved by improving overall driving conditions or improving driving quality. There appears to be a critical value for number density where throughput spikes, and this value is quite low. Methods for reducing the number of cars on the road to achieve this could include

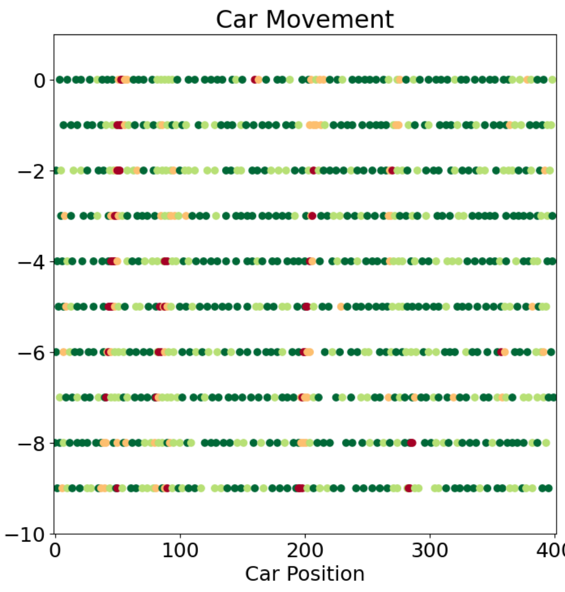

Figure 14. Still frames of the simulation. Time increases downward with each consecutive road being 3 time steps later than the last to allow the traffic jams to be more visible. The road is length 400, with 80 cars on the road, a slow down probability of 0.25, and a max velocity of 3. The velocities are signified via colours, with dark green representing the highest velocity, dark red 0 velocity, and yellow middling velocity.

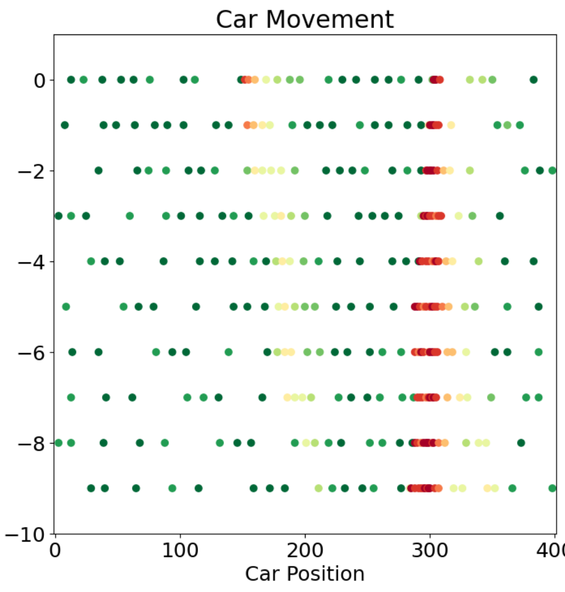

Figure 15. Still frames of the simulation. Time increases downward with each consecutive road being 3 time steps later than the last to allow the traffic jams to be more visible. The road is length 400, with 32 cars on the road, a slow down probability of 0.25, and a max velocity of 9. The velocities are signified via colours, with dark green representing the highest velocity, dark red 0 velocity, and yellow middling velocity.

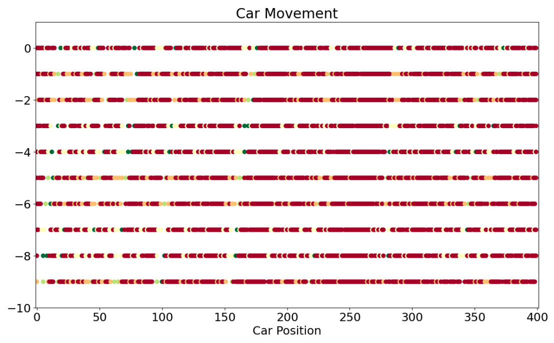

Figure 16. Still frames of the simulation. Time increases downward with each consecutive road being 3 time steps later than the last to allow the traffic jams to be more visible. The road is length 400, with 300 cars on the road, a slow down probability of 0.25, and a max velocity of 3. The velocities are signified via colours, with dark green representing the highest velocity, dark red 0 velocity, and yellow middling velocity.

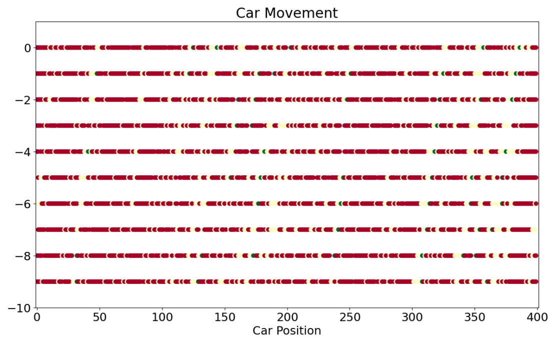

Figure 17. Still frames of the simulation. Time increases downward with each consecutive road being 3 time steps later than the last to allow the traffic jams to be more visible. The road is length 400, with 300 cars on the road, a slow down probability of 0.25, and a max velocity of 5. The velocities are signified via colours, with dark green representing the highest velocity, dark red 0 velocity, and yellow middling velocity.

promoting public transport, or providing benefits for using bikes or walking.

REFERENCES

Aldegheishem A., Yasmeen H., Maryam H., Shah M. A., Mehmood A., Alrajeh N., Song H., 2018, Sensors (Basel, Switzerland), 18, 1983
Jorgensen W. L., Chandrasekhar J., Madura J. D., Impey R. W., Klein M. L., 1983, J. Chem. Phys., 79, 926
Martin A., Suhrcke M., Ogilvie D., 2012, American Journal of Preventive Medicine, 43, e45
Metropolis N., 1987, Los Alamos Science, 1987 Special Issue Dedicated to Stanislaw Ulam, 125
Ohtaki H., Radnai T., 1993, Chemical Reviews, 93, 1157
The CMS Collaboration 2012, Journal of Instrumentation, 7, P10002
Yakowitz S., Krimmel J. E., Szidarovszky F., 1978, SIAM Journal on Numerical Analysis, 15, 1289

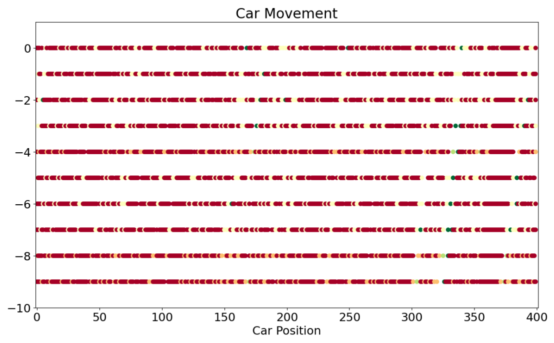

Figure 18. Still frames of the simulation. Time increases downward with each consecutive road being 3 time steps later than the last to allow the traffic jams to be more visible. The road is length 400, with 300 cars on the road, a slow down probability of 0.25, and a max velocity of 7. The velocities are signified via colours, with dark green representing the highest velocity, dark red 0 velocity, and yellow middling velocity.

APPENDIX A: MONTE CARLO SIMULATIONS

To create an accurate simulation I used a Monte Carlo Simulation. Monte Carlo Simulations use random number generation or random sampling to solve problems that are too complex to solve analytically. They can only provide approximate solutions to problems, and their accuracy is highly dependant on the amount of simulations performed. In academia , Monte Carlo simulations are commonly used in numerical integration (particularly for higher dimensional integrals, as the error scales the same in any dimensions whereas for other numerical methods errors decreases more slowly), mathematical optimisation (as explored in Yakowitz et al. (1978), where the use of Monte Carlo sampling is examined for a variety of mathematical problems), generating data from a probability distribution, and more. A slightly more detailed account of other uses of Monte Carlo simulations is given in the following sections.

A0.1 *Structure and Dynamics of Hydrated Ions*

In Ohtaki & Radnai (1993), Monte Carlo simulations were used to study hydrated ions. Ionic Hydration is the process by which water molecules are attracted to either positive or negative ions. In this paper, Monte Carlo simulations were used to generate random atomic configurations, which are then accepted or rejected based on the total energy of the system - if it decreases, the configuration is accepted otherwise it is not.

A0.2 *Comparison of Simple Potential Functions for Simulating Liquid Water*

In Jorgensen et al. (1983), different simple intermolecular potential functions used to quickly simulate liquid water are investigated for their accuracy compared to real water. Monte Carlo simulations were used to generate configurations for testing each of the potentials.

A0.3 *Performance of CMS Muon Reconstruction in pp Collision Events at $\sqrt{s} = 7$ TeV*

The CMS (or Compact Muon Solenoid) is a particle physics detector built at CERN. The paper The CMS Collaboration (2012) aims to measure the efficiency of the CMS in the detection of muons in pp (proton-proton) collisions. In this paper, Monte Carlo simulations are used to generate samples of events in order to compare the results obtained in experiment to our simulations and models.

APPENDIX B: MODULES USED

For the generation of random numbers for my Monte Carlo simulation I used numpy.random, a random number routine in the python module numpy. It uses PCG64 to provide bits for its pseudo-random generation. To place the cars, I used numpy.random.randint to generate random integers from a discrete uniform distribution. It takes an inclusive lower bound value, and exclusive upper bound value, along with optional arguments of the output shape (as in, what dimension of array should be created) and the data type of the numbers to be produced. For randomly slowing down the cars, I used numpy.random.rand. This function generates uniform random numbers from 0 (inclusive) to 1 (exclusive), and takes in optional arguments for the shape of the array to fill with random numbers.