Rush Hour Traffic School

Team 112

Saturday Science and Math Academy

Daniel Washington

Mentor: Wayne Witzel

4/6/11

Introduction

The problem that is being addressed by this project is the continued onset of poor driving behaviors in today's society. This year over 33,800 people died of car crashes. The total damage cost of cars involved in crashes, both fatal and non-fatal, this year was estimated to be $230.6 Billion! By creating a simulation that demonstrates exactly how important safe driving is, we can then research ways to reduce these horrible numbers. Therefore, the main problems with current driving behavior must be isolated, and then an instructional program to teach people how to drive in a safer manner should be made.

Reckless and careless drivers are a serious danger to society. Traffic patterns are always changing based on the behaviors of the individuals behind the wheel. Some people cause traffic jams while others obey the rules of the road and get where they need to be safely and quickly. In this project a simulated highway system was created with which to test exactly which driver behaviors have the most adverse effect on traffic flow. This simulation can test trailing time and distance, gradual and sudden stops, and aggressive driving patterns. By programming cars to use a delayed reaction based on human reaction time to respond to the other vehicles of the road, numerous potential scenarios, both good and bad were tested. While most drivers are aware of the common rules of driving such as the "two second rule," most do not actually follow them. This project attempts to show exactly what kind of effect these behaviors can have on overall traffic flow.

Through the Netlogo program and the included models library, we were able to create a mathematically correct simulation of standard traffic patterns. Throughout the project we tested the effects of congestion, trailing time, reaction time, trailing distance, and speed restrictions. By using a system of phase transition diagrams I was able to denote the traffic behavior as "solid," "liquid," or in transition.

The Project

When I began this project I was new to both programming as a whole and the Netlogo system. Therefore I chose to base my simulation on a preexisting model. Then I created a mathematical model to effectively match traffic patterns.

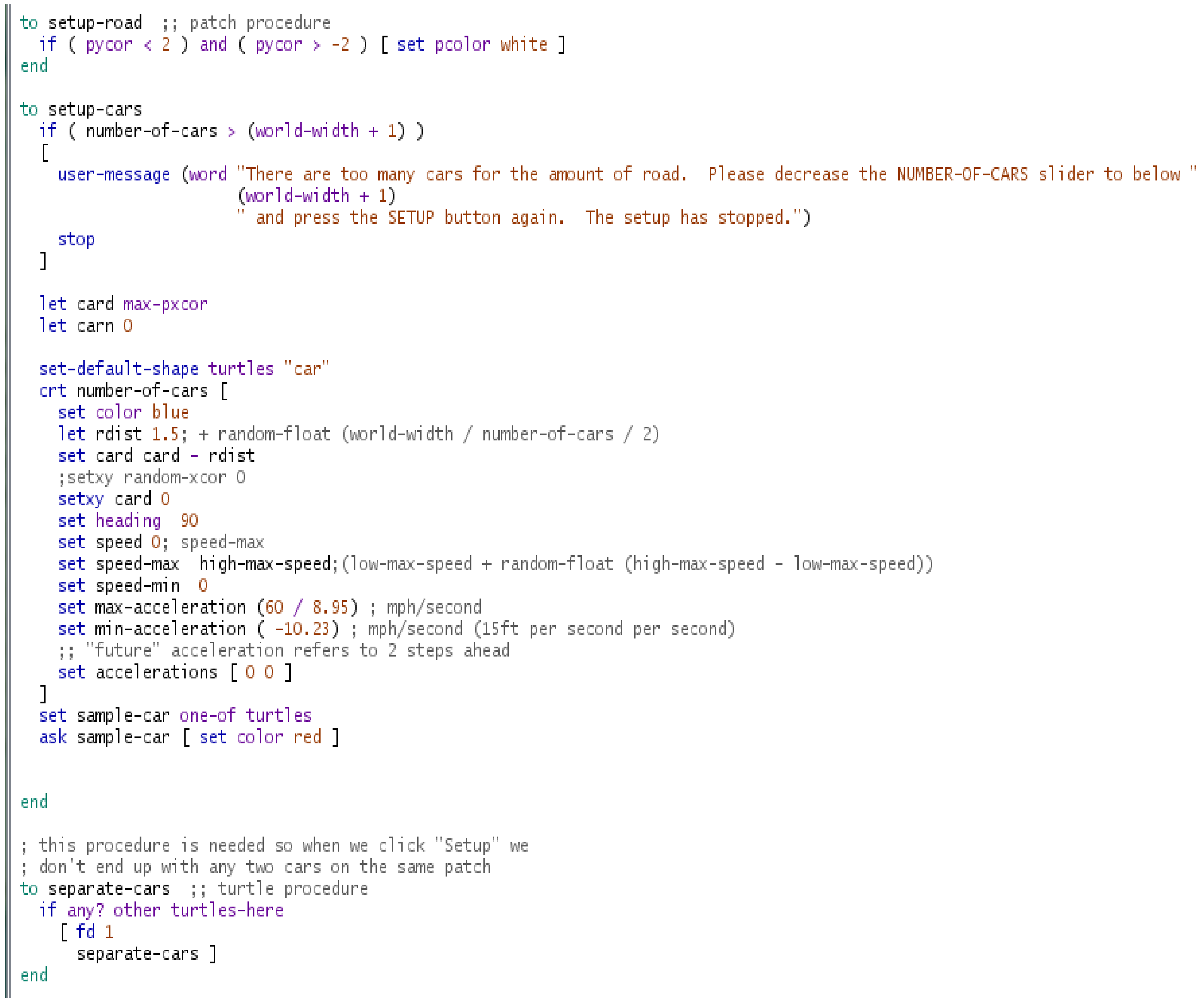
Behavior Patterns

I configured the cars to behave in a human-like manner. I incorporated functions for reaction time and gradual acceleration and deceleration. Next I researched the average acceleration of cars as well as their deceleration or maximum braking potential. These values were then incorporated these as a function of velocity and an alternating function of distance and time. We did this because it was the most effective way to implement the “two second rule” of driving. The cars are effectively programmed to take their current velocity and determine how long it will take to reach the car ahead. If that time is less then their distance away then they will slow down and conversely, they will speed up if distance is greater. With this behavior they will effectively resolve into a regularly spaced line when everything is moving correctly. However we have created sliders to control the ideal trailing time and distance as well as the anticipation or ability to notice changes in the forward car. Finally we added sliders to control the rate of acceleration and deceleration to help test wether gradual or choppy speed changes are better

The Code

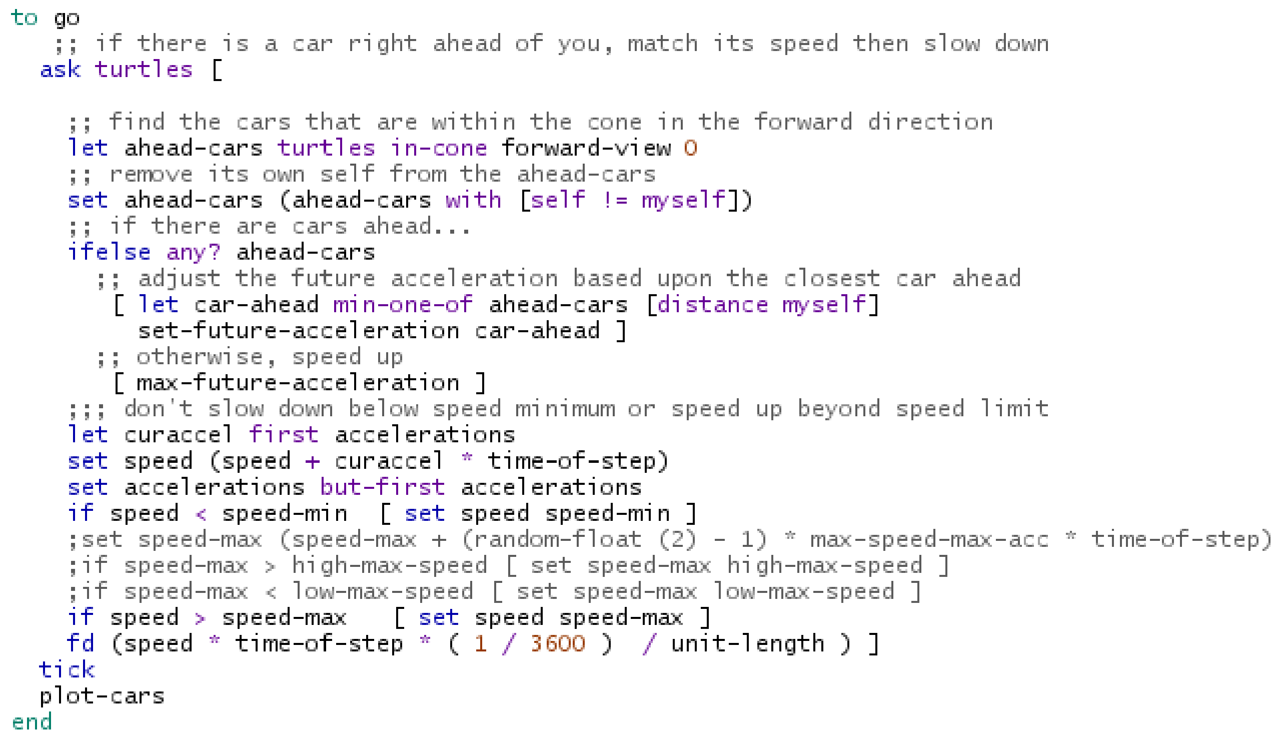
The programming code has four major portions: the setup, the movement functions, the acceleration and reaction time functions and the data monitoring.

The first part of the code is a simple setup function that creates a simple road with a specific number of cars as determined by a slider. It randomly denotes a single car as a "sample" to allow me to track an individual driver behavior. This portion of the code is where the cars are given their parameters including maximum and minimum acceleration and maximum and minimum speed.

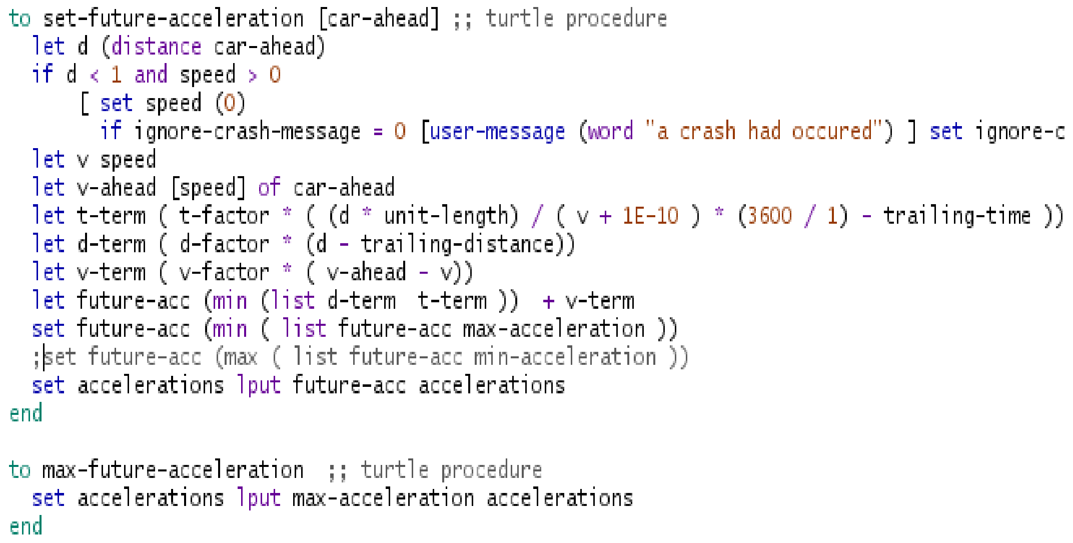


```
to setup-road  ;; patch procedure
  if ( pycor < 2 ) and ( pycor > -2 ) [ set pcolor white ]
end

to setup-cars
  if ( number-of-cars > (world-width + 1) )
  [
    user-message (word "There are too many cars for the amount of road.  Please decrease the NUMBER-OF-CARS slider to below "
                       (world-width + 1)
                       " and press the SETUP button again.  The setup has stopped.")
    stop
  ]

  let card max-pxcor
  let carn 0

  set-default-shape turtles "car"
  crt number-of-cars [
    set color blue
    let rdist 1.5; + random-float (world-width / number-of-cars / 2)
    set card card - rdist
    ;setxy random-xcor 0
    setxy card 0
    set heading  90
    set speed 0; speed-max
    set speed-max  high-max-speed;(low-max-speed + random-float (high-max-speed - low-max-speed))
    set speed-min  0
    set max-acceleration (60 / 8.95) ; mph/second
    set min-acceleration ( -10.23) ; mph/second (15ft per second per second)
    ;; "future" acceleration refers to 2 steps ahead
    set accelerations [ 0 0 ]
  ]
  set sample-car one-of turtles
  ask sample-car [ set color red ]


end

; this procedure is needed so when we click "Setup" we
; don't end up with any two cars on the same patch
to separate-cars  ;; turtle procedure
  if any? other turtles-here
    [ fd 1
      separate-cars ]
end
```

The second part of the code is where the cars are given their exact run functions. Here we set each car to a speed as determined by the system. The cars are made to first detect the location of the cars in front of it. If it has the appropriate amount of space in front, it will begin to accelerate to the maximum speed. They are also made to constantly check to see the position of other cars which directly affects their motion.



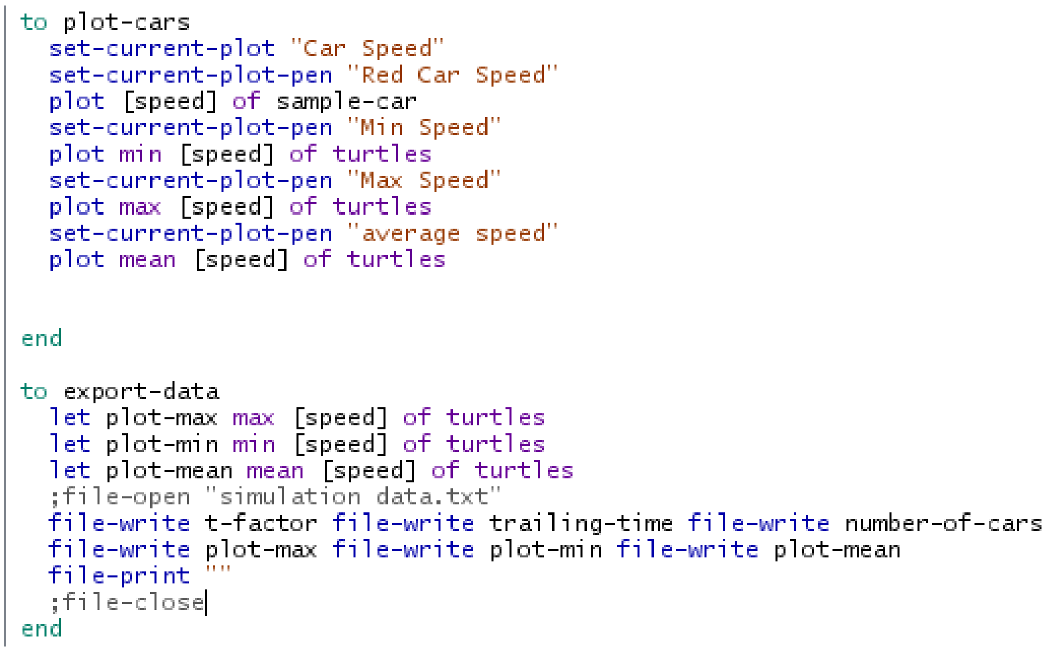
```
to go
   ;; if there is a car right ahead of you, match its speed then slow down
  ask turtles [

    ;; find the cars that are within the cone in the forward direction
    let ahead-cars turtles in-cone forward-view 0
    ;; remove its own self from the ahead-cars
    set ahead-cars (ahead-cars with [self != myself])
    ;; if there are cars ahead...
    ifelse any? ahead-cars
      ;; adjust the future acceleration based upon the closest car ahead
       [ let car-ahead min-one-of ahead-cars [distance myself]
         set-future-acceleration car-ahead ]
      ;; otherwise, speed up
       [ max-future-acceleration ]
    ;;; don't slow down below speed minimum or speed up beyond speed limit
    let curaccel first accelerations
    set speed (speed + curaccel * time-of-step)
    set accelerations but-first accelerations
    if speed < speed-min  [ set speed speed-min ]
    ;set speed-max (speed-max + (random-float (2) - 1) * max-speed-max-acc * time-of-step)
    ;if speed-max > high-max-speed [ set speed-max high-max-speed ]
    ;if speed-max < low-max-speed [ set speed-max low-max-speed ]
    if speed > speed-max   [ set speed speed-max ]
    fd (speed * time-of-step * ( 1 / 3600 )  / unit-length ) ]
  tick
  plot-cars
end
```



```
to set-future-acceleration [car-ahead] ;; turtle procedure
  let d (distance car-ahead)
  if d < 1 and speed > 0
     [ set speed (0)
       if ignore-crash-message = 0 [user-message (word "a crash had occured") ] set ignore-c
  let v speed
  let v-ahead [speed] of car-ahead
  let t-term ( t-factor * ( (d * unit-length) / ( v + 1E-10 ) * (3600 / 1) - trailing-time ))
  let d-term ( d-factor * (d - trailing-distance))
  let v-term ( v-factor * ( v-ahead - v))
  let future-acc (min (list d-term  t-term ))  + v-term
  set future-acc (min ( list future-acc max-acceleration ))
  ;set future-acc (max ( list future-acc min-acceleration ))
  set accelerations lput future-acc accelerations
end

to max-future-acceleration  ;; turtle procedure
  set accelerations lput max-acceleration accelerations
end
```

The third portion of the code is the most mathematically intensive portion. It is where the functions for acceleration and reaction time are generated. We use a system of five sliders to affect these variables. They are “trailing time” “trailing distance” “d-factor” “t-factor” and V-factor.” We created a maximum acceleration and deceleration which govern the entire equations. Each car is constantly checking for a minimum value between time and distance relative to the cars ahead. It will treat the lowest value as the active acceleration modifier. Similar to humans, if you are moving faster, you will notice a higher distance until impact, but a shorter time. The cars in the simulation behave in the same manner.

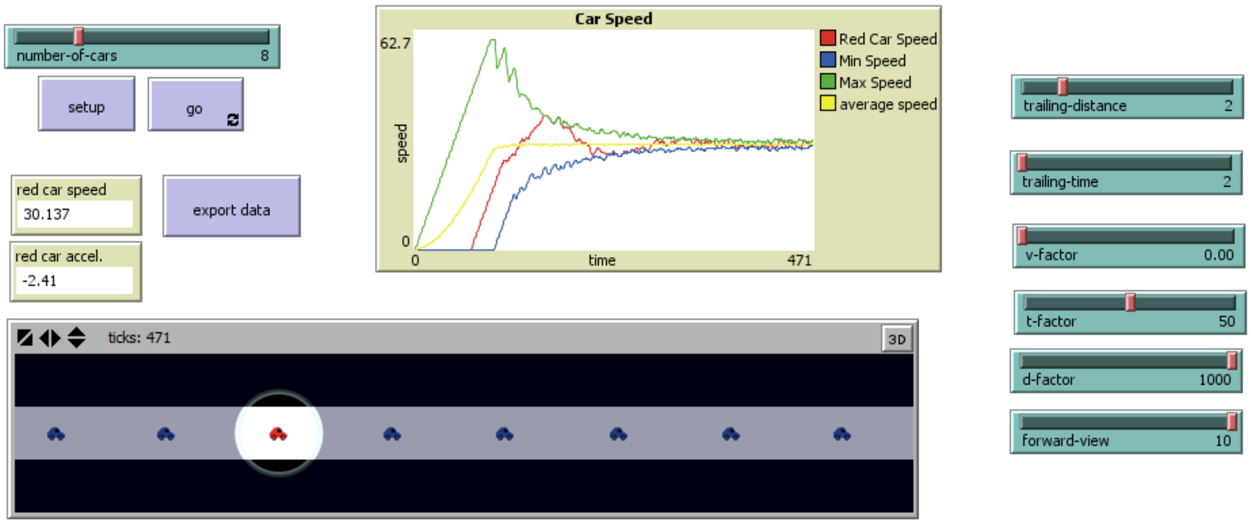
The final portion of the code is centered on data monitoring. It creates a line graph that updates at every time step of the simulation. This graph provides the global maximum speed, the global minimum speed, global average, and the speed of the sample car at any given time. This allows the user to see the relationship between the entire system’s speed and that of a single car.



```
to plot-cars
  set-current-plot "Car Speed"
  set-current-plot-pen "Red Car Speed"
  plot [speed] of sample-car
  set-current-plot-pen "Min Speed"
  plot min [speed] of turtles
  set-current-plot-pen "Max Speed"
  plot max [speed] of turtles
  set-current-plot-pen "average speed"
  plot mean [speed] of turtles


end

to export-data
  let plot-max max [speed] of turtles
  let plot-min min [speed] of turtles
  let plot-mean mean [speed] of turtles
  ;file-open "simulation data.txt"
  file-write t-factor file-write trailing-time file-write number-of-cars
  file-write plot-max file-write plot-min file-write plot-mean
  file-print ""
  ;file-close
end
```

The interface

The interface of this program is designed to be very user friendly and very adjustable. It contains the simulation window which gives a real-time display of the simulation as it would appear in real life. Above this is the graph of simulation speeds. To the left are the setup and run buttons as well as the slider affecting the number of cars in the system. Finally, on the right hand side is the behavior sliders. The trailing time and distance affect the “ideal” values which the cars will attempt to emulate. The T-factor and D-factor sliders modify the cars perception of time and distance in their behaviors. Finally, the V-factor slider adjusts the reaction time of the cars. This slider creates an effect similar to a damping effect seen on springs. This factor creates a form of damping between each of the cars. When it is high, the cars will behave as if there is a stiff spring between them. When it is low, they will behave as if there is no spring separating them at all.



number-of-cars 8

red car speed 30.137

red car accel. -2.41

ticks: 471

trailing-distance 2

trailing-time 2

v-factor 0.00

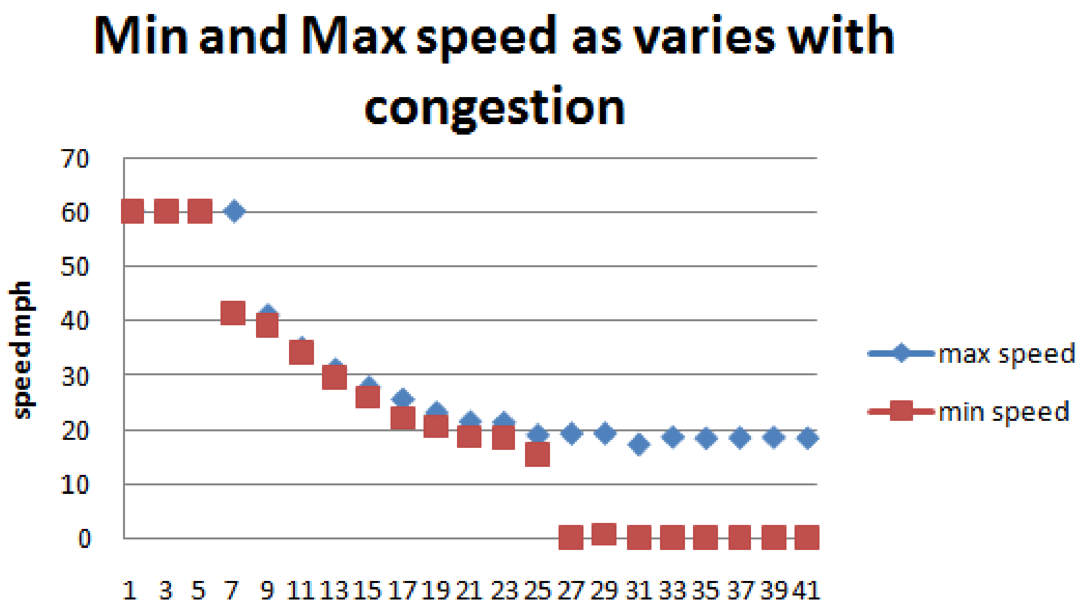
t-factor 50

d-factor 1000

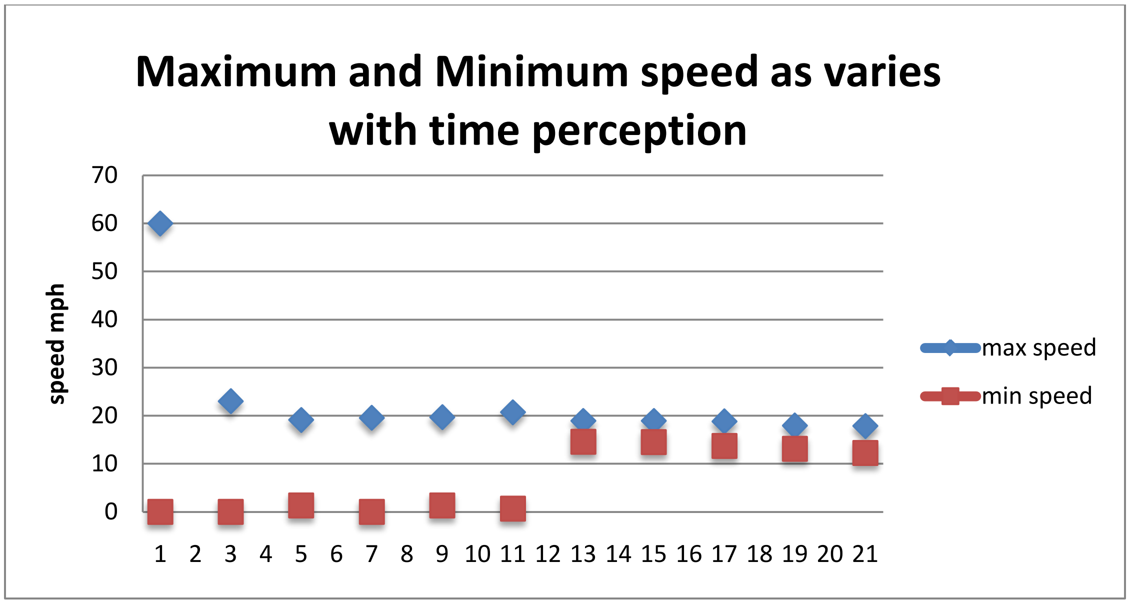
forward-view 10

Results

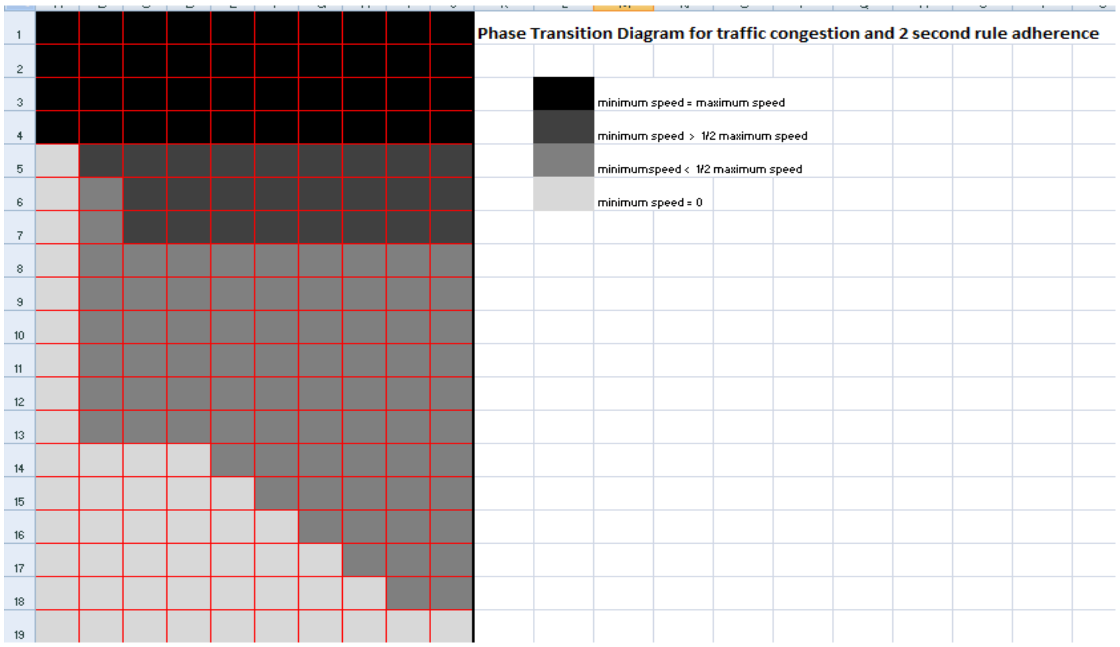
Through numerous trials and tests using my simulation I was able to create a series of graphs that demonstrates a very compelling correlation between perception of time, adherence to the 2 second trailing rule, and the general congestion of a given road system. The most compelling findings were the correlation between T-factor and traffic flow. The higher the T-factor or perception of the “2-second rule,” the more cars were able to remain in motion in a closed highway system. I also found the obvious correlation between congestion (the percentage of cars relative to the maximum capacity of the road) and traffic flow. As the number of cars in the system decreases, overall speed increases.



This graph is the function of the minimum and maximum speeds of the entire system with relation to the average congestion on the road. Two cars is the representation of absolute minimum traffic, while 44 represents the absolute maximum capacity of the given road. This graph demonstrates the most obvious, yet fundamental aspect of traffic. It cannot flow if it is overpopulated. The graph takes the form of a piece-wise function because at such extreme high and low traffic levels, the cars will either max out their speed, or remain stopped. Were it not for the continued minimum and maximum values, this function would form a perfect parabola.



This graph represents the maximum and minimum speeds of the system as compared to a variable called "T-factor." The T-factor is a representation of how closely the drivers follow the "two second rule." When this value is low, the cars ignore the rule completely. This causes them to behave with sporadic behavior, including sudden stops and wild acceleration. Inversely, when the value is high, the cars will follow the rule precisely which causes them to behave with more smooth motions including gradual stops and accelerations. As you can see, when T-factor is unreasonably low, one car will always be going the maximum speed while the others will remain at or near a standstill. However, when T-factor is high, the cars behave in the most controlled manner. This allows each and every car to move together at a slower pace. If the maximum and minimum repetition is removed, this graph would form a negative cubic representation. After averaging the speeds together, a high T-factor results in a much lower overall travel time for the entire system instead of just for one car.



Conclusion

In my findings, I discovered a relatively positive trend between good driving and traffic flow. While sporadic and dangerous behaviors moves a single car faster, it will slow down the entire system and ultimately prevent traffic from moving as a whole. Therefore, if drivers would stop thinking in the sense of "me" and start thinking in terms of the entire road, everyone would reach their destination faster and safer. The ideal driving patterns are using gradual stops and starts, high perception of surrounding cars. I have also found a direct correlation between congestion and average speed of the system which cannot be changed by any driver behavior. Therefore, if drivers behave in a more controlled and safe manner, the roads will be safer and traffic jams will be less frequent resulting in faster and safer travel.

Future Plans

The eventual goal for this project is to expand it into a piece of software that acts as an interactive simulation. This simulation would have the user control a car from a bird eye view and see how the surrounding traffic is affected by their actions. It will direct the driver's attention to the change in road conditions as a result of their decisions and take focus away from just their car. This system has immense potential for implementation in the public school system, or private driving schools. Ideally, this software would develop into a complete copyrighted program which could be utilized by driving schools. This would certainly contribute to the reduction of reckless driving and the massive car crash rate.

Works Cited

Coexisting phases and lattice dependence of a cellular automaton model for traffic flow
Raissa M. D'Souza. Web. March 4, 2011
http://mae.ucdavis.edu/dsouza/Pubs/bml-pre.pdf

Statistics On Traffic Violations.
Web. March 6, 2011
http://www.trafficviolationlawfirms.com/Statistics.cfm

NetLogo User Manual- Netlogo program.
Web. Feb. 21, 2011
http://ccl.northwestern.edu/netlogo/

Wikipedia- The Free Encyclopedia
Wikimedia corporation. Web. January 15, 2011
http://en.wikipedia.org/