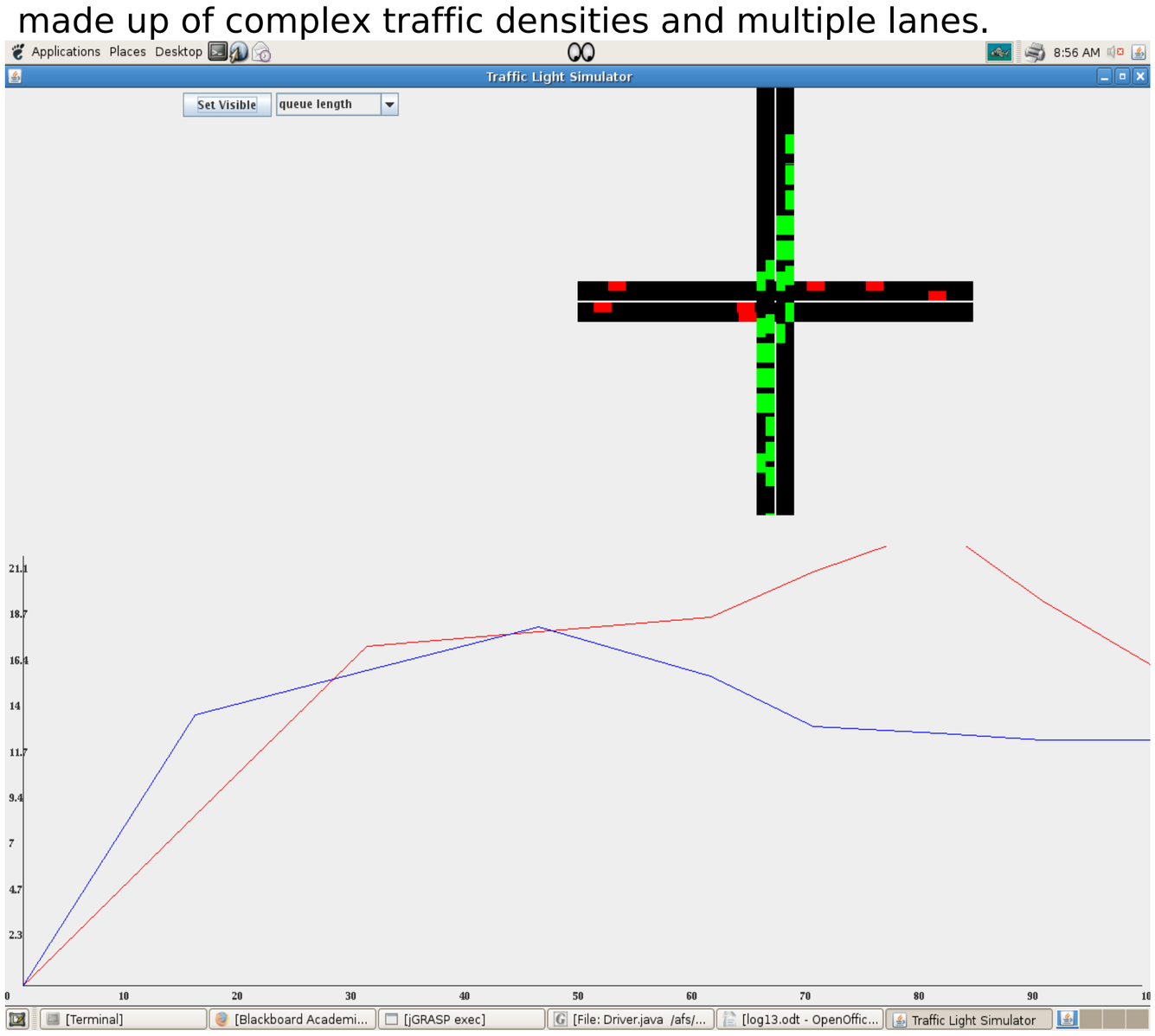
Traffic Light Simulation
Lynn Jepsen
TJHSST Computer Systems Lab 2005 - 2006

Abstract

This project is meant to simulate a busy traffic light. The program recognize patterns in the intersection and then uses that information to make the light as efficient as possible. These patterns could be related to the time of day and/or the day of the week. At first I would purposely input recognizable patterns and see if the program would catch it, but eventually the plan would be to possibly use this program at a real intersection. There are many variables that the program takes into account including traffic density, number of lanes, etc.

Simulation

The first section that I developed was a basic simulation of a 4 lane intersection. I programed in cars that move with realistic speeds and acceleration This is the visual component that shows the intersection. The cars stop behind each other, accelerate at reasonable speeds and obey other “rules of the road”. I set each intersection at a set traffic density that can be changed at the beginning of each experiment. Traffic density is the number of cars then drive onto the road in a minute. The traffic densities are still slightly random in that the average number of cars is close to the traffic density. I later added multiple lanes, so I could test my algorithm on intersections made up of complex traffic densities and multiple lanes.



Results

This program looks like a realistic model of a traffic light. The cars speed up and slow down realistically, obey lights, and don't run into each other. The light algorithm decreases the number of back ups in the intersection. The light algorithm is only so useful. Once the traffic densities get too high for the current number of lanes, then then no amount of changing cycle length or ratio will decrease queue length or wait time. However, if the intersection has a reasonable relationship between traffic density and number of lanes, then the light algorithm optimizes the efficiency variables, and increases traffic flow!

Background

Traffic lights have always caused problems with traffic flow. If the light has no information about the intersection it is controlling, then you can sit at a red light for what seems hours. Even worse is when the light is much to short and you have to wait through lots of cycles. We have all experienced this. There has been a lot of research done to try and automate cars, using GPS, so that they all pass through the intersection harmlessly. However, this technology can be expensive, and it would take a long time to install it into all cars in order for the project to work. I think a much cheaper and temporary solution would be to fix the traffic lights our society already has with brand new algorithms that would make the light efficient. In order to see if this algorithm is working, and gather lots of data very quickly, I designed a simulation of a traffic intersection to test the algorithm on. This way I can test pictorially, the aerial view of the intersection, but also graph the efficiency variables to see general trends.

Light Algorithm

The light algorithm is what decides when to change the lights from red to green in all four directions. In order to do this there are two variable that must be found. These are cycle length and ratio. Cycle length is how long it takes the intersection to go through an entire cycle with both sides having their chance to be green. Ratio is the ratio of green light time in the north/south direction compared to the east/west direction. I wanted to find the correct variables for the light cycle at each instant on an intersection that would maximize efficiency. I defined efficiency with three other variables. They are queue length, wait time, and green light usage. Each instant on the intersection has one main variable that I cannot effect with my light algorithm. This is the traffic density. So if we consider traffic density, cycle length, and ratio to be our independent variables and our efficiency variables are our dependent variables. The experiment is to change th cycle length and ratio (because you can't change traffic density) in order to optimize efficiency (all three variables). The trick comes is getting all three efficiency variables optimized at once. The light algorithm uses previous traffic densities, cycle lengths, and ratio to try and find which combination of best optimized all three efficiency variables. The algorithm must compromises to make sure that all three efficiency variables are good. The intersection continues to compromise and eventually begins to use similar cycle lengths and ratios. It approaches an optimization after enough time. Considering that an intersection receives tons of data every day, I feel like an algorithm that needs lots of information is appropriate for the situation.

I added another visual component to my program once I had finished my first draft of the light algorithm. It allows you to graph all three efficiency variables. This way you can see if the algorithm is really optimizing the intersection. I made is so that it graphed a north/south line and an east/west line. The closer the two lines are to each other, the better the optimization.