Multidisciplinary Senior Design Conference
Kate Gleason College of Engineering
Rochester Institute of Technology
Rochester, New York 14623

Project Number P18393

Optimizing Traffic Flow Using Computer Modeling

Joe Curcie
Industrial Engineer

Bryan Healy
Computer Engineer

Justin Maggio
Electrical Engineer

Eilif Mikkelsen
Industrial Engineer

Samara Reddy
Electrical Engineer

Gaitz Soponski
Computer Engineer

ABSTRACT

The goal of this project was to create a user-friendly program to make the creation of an intersection in SUMO (Simulation of Urban Mobility) easier for new users. The purpose of this program is for new users to have an easier experience when first beginning to use SUMO for its simulation purposes. When using this program, a user will be able to download a package, open an Excel sheet and enter in its fields, and run a command prompt with documented steps which will allow the user to create a customizable intersection for their specific needs. The package will also allow for more advanced users to add more customization options to the baseline code in the future, which will give the user more customization to the user's need with the user-friendly Excel sheet entry. By using lane area detectors (cameras) in an intersection, the user should experience a more time and emissions efficient intersection. The final product successfully created very basic intersections and began to optimize the intersection through external traffic light control.

INTRODUCTION (BACKGROUND)

People spend a significant amount of their lives in cars. A large amount of this time can be spent in traffic resulting from backed up intersections. Many frustrating situations can arise, such as pulling up to an empty intersection and the traffic light turning red, because of outdated and inefficient designs. This causes extremely unnecessary waste in terms of emissions of harmful gases produced by cars. The aim of the TrafficSense team is to alleviate these situations by making it easier to define intersections in software for traffic researchers to use. Traditionally, traffic lights use induction loops in the ground to detect cars approaching an intersection. As opposed to the traditional method, our team hopes to incorporate lane area detectors into intersection modelling to bring earlier detection for intersection car detectors, which may potentially lead to more efficient intersections in the future.

THEORY

British Petroleum (BP) completed a study[1] in June 2016 and found that the U.S. accounted for approximately 20% of the global consumption of oil in 2015. According to the U.S. Energy Information Association[2], in 2016 the U.S. consumed approximately 143.37 billion gallons of gasoline, which works out to about 391.73 million gallons consumed daily. Approximately 20 lbs. of CO_2 are produced from burning just one gallon of gasoline. With every gallon of gasoline burned, there are harmful emissions released, such as the common greenhouse gas of carbon dioxide (CO_2). Based on those numbers, the U.S. produces around three trillion lbs. (1.36 billion metric tons) of CO_2[3]. This could have severe and long-lasting impacts on the environment.

Gasoline consumption will continue to be high as the U.S. continues to utilize technology that relies on it. This is a result of overreliance on outdated technology. It should be a goal to decrease gasoline consumption for the sake of the environment. An easy place to start would be to decrease the amount of time vehicles are stuck at traffic lights. Texas A&M conducted a study and concluded that Americans spent roughly 6.9 billion hours stuck at traffic lights in the year 2014 alone. This same study also found that three billion gallons of gasoline were wasted idling at intersections. This waste can partly be attributed to inefficiencies present in the current traffic light system. There are multiple opportunities to reduce waste resulting from traffic light inefficiencies.

PROCESS

In the starting phases of the project, the initial focus was to create a smarter intersection that would utilize various detection systems to identify cars approaching the intersection from all directions. A centralized processor would then use this information to determine which directions would create the most significant reduction in waiting time at traffic lights. The project would have used physical cameras to detect vehicles and adjust the traffic system to reduce wait time. While this goal still remained through the entire process, the end deliverable changed drastically as time went on. The focus eventually became a software deliverable that

would help make it easier for traffic researchers to model intersections in software, so more time could be spent researching rather than learning how to build models.

The TrafficSense team developed a tool using a program called SUMO to help researchers simulate the effects of a smart traffic intersection. SUMO is a very extensive program used to model traffic systems. Combining this with the customization capability provided by SUMO made it the perfect choice for creating the model[5]. The simulation is set in a traditional cross style intersection and utilizes the equivalent of a camera detection system. Once the cars are detected, a deep reinforcement learning algorithm is used to predict the best sequence of traffic control to optimize for specific variables of the simulation. In this case the variable is the vehicle emissions within the intersection. Not only does this work to reduce emissions which yields a more environmentally friendly system, but it also reduces the amount of time commuters spend waiting at the intersection.

Customer Requirements

There was some trouble with settling on a set design for this project. To begin, the goal was to reduce emissions associated with standard intersections by reducing acceleration and idling time. Initially the product would involve both hardware and software design. As time went on, the project shifted focus to strictly software design. Creating a customizable model to simulate intersections became the goal. Creating models for researchers to use often takes a significant amount of time on its own. This created an opportunity for the team to build and thrive. The team agreed to deliver a package program in which a user could download and have an easier time creating their customized intersection. The package would contain an Excel sheet that the user would modify and would create the necessary configuration files to operate generate a traffic demand to simulate the designed intersection.

Engineering Requirements

As mentioned previously, the project changed scope at multiple points until it was decided that a simulation tool would be the best way to satisfy the customer requirements. In order to more accurately measure the success of the project, a set of engineering requirements was created detailing the high-level goals. Table 1 details some of the requirements. The main requirement was to reduce emissions by 20% with supporting requirements necessary for a robust design.

Table 1. Engineering Requirements

Description	Initial Value
Smart Traffic Light must reduce emissions	20% emission reduction
Tool must support common intersections	80% of all intersections
Must be compatible with common technology platforms	Tools should be cross-platform between Windows, Mac, and Linux
Must Simulate all lanes of a given intersection (pass/fail)	Pass

Team

The team consisted of two electrical engineers, two industrial engineers, and two computer engineers. The electrical engineers focused on discovery and understanding the configuration side of SUMO. This consisted of providing examples and documenting features of SUMO. While SUMO had a large assortment of features the documentation on how to utilize those features was lacking, which made it necessary to appoint members to understand how SUMO worked in a more in depth. One of the industrial engineers was designated the project manager, who helped keep the team on track, set up meetings with the customer, and ran meetings to ensure tasks were completed by the team. The other industrial engineer worked with the computer engineers to design the software. Between them, work was split to complete the configuration side, which involved generating the SUMO input files, and retrieving data from the running simulation to be used in the optimization process.

RESULTS AND DISCUSSION

Beginning work on this project involved first picking a program to build around. There were a few types of programs out already that allowed for simulation of traffic. It was decided that SUMO offered the best customizability and had the most features available. Looking deeper into SUMO, it was found that some features of SUMO were implemented with Python coding, which aligned well with choosing Python for our coding language of our tool.

The development of the project can be broken down into work done for the configuration side and the runtime side. The combination of these two parts make up a fully functional tool for the users. Due to the high learning curve of SUMO, this simulation tool would be more user friendly for many types of engineers who would want to test their own types of algorithms before trying to do a physical implementation in the real world.

SUMO also allows an outside program to control the timing of the traffic lights. This is accomplished using TraCI (Traffic Control Interface). By using TraCI, the optimization side of the project will be easier to accomplish. TraCI is an “all-knowing” controller which can use the locations of cars in the simulation to time the traffic lights the most efficient way possible.

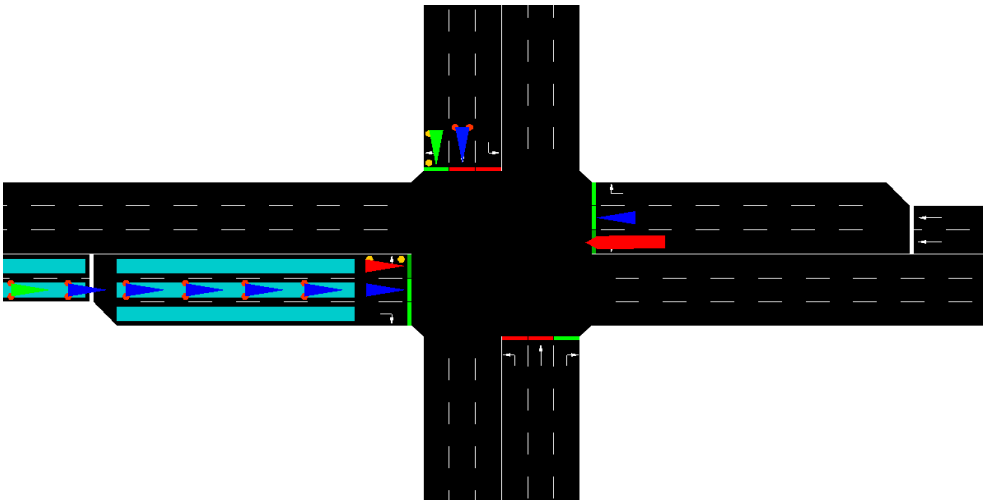

Figure 1. SUMO intersection including cars, lane splits, and lane area detectors shown on the SUMO’s GUI

Though the concept of an intersection is simple in principal, SUMO requires many definitions to create even a basic intersection, all of which would have to be created by the user. To create an intersection with cars that would run, a user needed at least five different configuration files made, and more if they wanted to have more customization and/or visuals. The first three files go hand in hand with each other, which are the node, edge, and connection files. The node file describes locations at which the roads used would be able to connect to. This is also where the placement of traffic lights would be determined. The edge file describes road characteristics, such as a road will connect between two different nodes, how many lanes, if there are any splits, and much more. With the edge file, you could also create one way only roads, or make the road lead cars to and away from the intersection. The connections file describes what lanes a car can travel to when it approaches the intersection via an edge. Though the connections file was not needed, it did allow for easy creation of left or right only turn lanes, or even straight and right turn only lanes. Without the connections file, SUMO would just place default connections, which at least would make sense for an intersection (i.e. no crisscross lane connections). Combining all of these files through a command using the command prompt, a user would create the net file, which really was just the contents of all the previous file expanded in this file. The net file is what SUMO reads to physically set up the intersection but does not place any cars. To place cars in the simulation, a route file needed to be created. These are the basic files needed to run an intersection.

If a user wanted to expand further, they could incorporate the use of detectors, which would be defined in additional files and would also create a logging file of the outputs of the detector as the simulation ran. Outputs of the detector include information such as what time the detector saw the collected information, the number of new vehicles entering the area, the average speed the vehicles were travelling, how much the area was occupied during the step, et cetera. All of this data can be used to create an effective optimization engine.

With the Excel sheet being the primary way to customize the intersection, the main design focus was to be very user friendly. As a result of this, some advanced customization had to be omitted. Customization was divided into different tabs within the Excel sheet. Each tab allowed the user to customize different parts of the system. One tab allows the user to modify some of the general simulation settings like setting a name for the configuration as well as how long the simulation will run. Another tab allows the user to modify the actual intersection. This includes incoming outgoing lanes as well as which inbound lanes are turning lanes and which direction they originate from. It also includes the speed limit for each road and the length of those roads. The final tab is used for some of the more advanced customization. This includes information about the age of the people going through the simulation, work hours of businesses, locations of schools, et cetera. This Excel sheet is then parsed by our program and the configuration files generated.

Category	Description	Units	Value	Value	Value	Value	Valid Data Type
			Branch 1	Branch 2	Branch 3	Branch 4	
Outbound Lanes	Number of Outbound Lanes	N/A	4	4	4	4	Positive Integer
Inbound Lanes	Left Turn Only Lanes	N/A	0	0	0	0	Positive Integer
	Left + Straight Lanes	N/A	1	1	1	1	
	Straight Only Lanes	N/A	2	2	2	2	Positive Integer
	Right + Straight Lanes	N/A	1	1	1	1	
	Right Only Lanes	N/A	0	0	0	0	Positive Integer
	Total number of Lanes	N/A	4	4	4	4	NO ENTRY
	Inbound Intersection ID		0	0	0	0	NO ENTRY
	Outbound Intersection ID		1	2	3	4	
	Angle away from intersection	Angle	N	E	S	W	
General	Speed Limit	Kilometers per Hour	20	20	20	20	Positive Integer
	Road Length	Meters	500	500	500	500	Positive Integer
	Branch ID		1	2	3	4	
	Priority	N/A	78	78	78	78	Unused
	Branch Type		Priority	Priority	Priority	Priority	

Figure 2. Example Template of Excel Sheet

Once the configuration files are created based on the user's specifications, the runtime program is started from the command prompt and is passed the location of the configuration files. The runtime program then starts the SUMO simulation using TraCI. The simulation can be run with or without the SUMO GUI visible.

Along with the runtime program, an interface is defined for an optimizer, which is trained after every tick of the simulation by the runtime program. For machine learning to work properly, there is a training period established. To facilitate the data collection storage and manipulation requirements of a machine learning traffic optimizer, the Rolodex class was created. The Rolodex provides an interface for the user to easily collect data for any object within the simulation using TraCI's built in subscription functions. By subscribing to a simulation variable, SUMO will automatically store the most recent value of the variable after every simulation tick.

The Rolodex builds upon this functionality by storing the desired data in domain-specific buffers. For example, if a user wanted to store the lane position and speed of a handful of vehicles, they could use the Rolodex to create a vehicle-domain buffer storing the specified data, the number of data points to be stored for each variable of each vehicle, and how many simulation ticks between each stored sample. The data can then be accessed by referencing a certain vehicle, or an attribute can be referenced to get the data for all vehicles of that attribute tracked by the Rolodex. Instead of listing a set of vehicle IDs, the user can also specify to have the Rolodex automatically update the objects being tracked. Specifically, whenever a new vehicle enters the simulation data will be collected for it, and when a vehicle exits the simulation the Rolodex will get rid of the data for it after the specified number of simulation ticks. Per the customer's requirements, the user will able to set up data collection using easy to remember

phrases such as “lane position” and “vehicle speed” rather than the more complex nomenclature used by SUMO and TraCI.

To control the traffic flow, a deep reinforcement learning approach was used. First, a Long Short-Term Memory (LSTM) based Recurrent Neural Network (RNN) was trained to take in several variables from each lane and vehicle in the simulation. Using back-propagation with a supervised-learning approach, it would then learn the best way to represent certain traffic scenarios that lead to factors such as aggressive driving, collisions, and long queueing times. Though collisions and driver-aggressiveness were not included during this project due to time-constraints, there is room for it to be added if the project gets renewed. A policy-gradient reinforcement learning algorithm was used to take in the representation of the current traffic scenario, produced by the RNN, and determine the best sequence of traffic light phases to minimize the amount of time spent waiting by vehicles, as well as potentially dangerous scenarios.

Budget

The team was able to complete the project with minimal use of external funding. Given the team used personal hardware, such as personal computers, the only part of the given $500 budget used was $100 to get JIRA for a full year for the whole team. JIRA is an online tool for managing projects and applies specifically well to software design. For managing projects, it is able to track new developments, software bugs, and link to many external applications. The team was able to use Slack for free, which allowed the team to communicate through computers and mobile phones. Slack also allows for easily shareable links and files.

CONCLUSIONS AND RECOMMENDATIONS

For many of the team members, this was their first experience working on a purely software project. The team was successful in creating an end-to-end product that created configuration files for SUMO, compiled them, ran a simulation, pulled data, and began to optimize the traffic light control. The team ran into many problems during development that affected the end product. Lane splits were documented for use but were unable to be included in the Excel sheet due to the difficulty of parsing lane splits and the timeline of the project. In addition, the individual lane connections were not included, as these heavily depended on the presence of lane splits. Given more time to complete the project, lane splits and customizable connections most likely would have been added.

The team believes there is much room for improvement on this project. Areas of improvement include allowing more customization of the intersection specifications, improving the user-friendliness of the Excel sheet, creating more realistic traffic demand, creating more realistic intersections, and making the Excel sheet able to handle multiple intersections. To make the product more useful, the previously listed improvements would need to be implemented.

During development, there was discussion of creating a GUI for users instead of the Excel sheet, which can dynamically update based on the selected options. A GUI tool would be more extensive and user friendly, especially to users without prior software development

experience. However, since most team members lacked familiarity with GUI development, an Excel sheet was made so more team members could understand it.

Looking back at the progress the team made over the course of two semesters, there were many lessons learned by the team. One of the biggest takeaways from the process was that in-person communication regarding the team's progress is extremely important. In addition to this, the team learned that documenting committed deliverables and decisions made is helpful for tracking progress. Many times, the team struggled to remember what was said weeks prior and promises to have work done were made and forgotten about.

ACKNOWLEDGEMENTS

The team would like to thank Kenneth Mihalyov for being the MSD guide throughout the project and putting in countless hours outside the class to aid in the team's design and execution. Ken was able to provide critical input and feedback that helped ensure the team's timeline and project goals were reasonable. The project did not have the same flow as one with a physical deliverable and Ken was able to adjust himself and the team to stay on track. The team would also like to thank Dr. Katie McConky, an ISE professor at RIT, for her inputs and feedback as the customer for the project. The team hopes the work completed this year will be useful to her and her graduate students in the future.

REFERENCES

[1]https://www.bp.com/content/dam/bp/pdf/energy-economics/statistical-review-2016/bp-statistical-review-of-world-energy-2016-full-report.pdf

[2]https://www.eia.gov/tools/faqs/faq.php?id=23&t=10

[3]https://www.eia.gov/tools/faqs/faq.php?id=307&t=10

[4]http://abcnews.go.com/US/time-americans-waste-traffic/story?id=33313765

[5]http://sumo.dlr.de/wiki/Simulation_of_Urban_MObility_-_Wiki

[6]https://github.com/TrafficSenseMSD/core