Implementation of Dynamic Traffic Light Controllers Using Artificial Neural Networks to Diminish Traffic Ordeals

Mohammad Samin Yasar and Md. Tahmid Rashid
School of Engineering and Computer Science
BRAC University
Dhaka, Bangladesh
{yasar.samin@gmail.com, tahmidrashid} @gmail.com

Abstract—Since the advent of modern transportation, the necessity to orderly move huge volumes of vehicles on the road has heightened while the techniques to conduct the orderly movements have grown ever challenging. This paper focuses on a novel approach for handling road traffic by incorporating an intelligent traffic light controlling system with the aid of artificial neural networks algorithm. As our experimental model, we have selected Dhaka city to be the center of our research as it is infested with traffic jam, ranking it towards the top among least livable cities of the world. Our endeavor is to make use of 24/7 Dynamic Traffic Light Controllers (DTLCs) based on artificial neural networks. As part of the solution, we have developed a program using a popular programming platform that would calculate sets of drive orders for traffic signal lights and we have optimized it to run on low power hardware. The output from the application is then verified within a road traffic simulation software. The cumulative results from the experiment portray the efficiency of applying an adaptive algorithm that dynamically calculates the order of signal lights. The decision making algorithm is designed to replicate, in a meager form, the human brain with the system trained to learn to respond to certain traffic situations. Static Traffic Light Controllers (STLCs) handle traffic flow with predetermined traffic conditions. In contrast, our approach would be a fully dynamic and spontaneous solution to mitigate the ongoing traffic crisis.

Keywords— traffic light controllers; artificial neural networks; DTLC; STLC

I. INTRODUCTION
Traffic predicament is never a new matter of discussion and centuries of research has been contributed towards reducing traffic congestion. Once upon a time, controlling traffic on roads was a daunting task for traffic police officers as they had to use hand signals or hold signed placards to route vehicles on streets. Those days are long gone and today, we have electronic traffic lights to direct the vehicles on the roads in orderly fashions. In spite of the progressions of modern technology, traffic ordeals are still endless in developing nations like Bangladesh. In Dhaka city where 131 million people dwell [1], traffic congestion has deteriorated drastically in the recent years in spite of the introduction of automated traffic lights. Periodic gridlocks are inevitable and traffic congestion gas garnered the title ‘menacing’. Measures such as the construction of flyovers, the construction of alternate roadway projects have been taken up to counter traffic congestion [1], [2]. However, as yet there has not been significant progress in combatting the problem in context.

Traffic in Bangladesh is seen as one of the major vices that is stifling the country’s economic progress, with an average of 2 hours being spent waiting on the roads [2]. This has caused innumerable sufferings and a decline in public welfare and has resulted in massive loss of productivity with the resulting effect being felt on the GDP. At present, BRTA employs static traffic light control (STLC), which has fixed cycle of green and red lights depending on the historical flow of cars along certain routes. The STLCs, which are based on fixed counters and do not have any sort of provision to consider the present traffic flow, have failed to significantly improve the situation with people still suffering from traffic jams as they used to. In severe cases it is seen that these STLCs are deemed inadequate to handle the traffic with the officer in site manually overriding them [3].

The proposed DTLC will have real time updates on the present traffic situation by means of sensors placed on each road. These sensor inputs would be fed into the neural networks in place, which will evaluate them to provide an optimal drive order. The artificial neural networks are used as they avoid the risk of being locked into local minima and will provide a reasonable robust solution to any condition of the intersection. Because of the computational load that these neural networks impose, we have used a small form factor microcomputer, the Intel NUC which powered by an Intel Atom microprocessor to handle the processing and memory loads of the algorithm.

The DTLC is part of the broader vision of an Intelligent Traffic System (ITS), which is in line with the vision of intelligent computer systems. The aim of is to record and maintain all sorts of transport activities under the digital canopy. It is an integrated system which can handle and predict all sorts of traffic challenges that may present itself on a regular day. DTLC is viewed as a stepping stone to that vision. With DTLC, there will be an improvement on the traffic regulation by eliminating the redundancy which is normally associated with the STLC. If there is a circumstance where one part of the road is congested while
the other part have almost no traffic, DTLC will help to bring stability and improve the situation and allow the congested part to clear out by prolonging the duration of the green light on that route. When the traffic load is low on all sides of the intersection, the duration of green light will be small and there will be a more rapid transition.

Alongside DTLC, which, at its very best, is still a passive and a reactive way of reducing traffic jam, this research also endorses another solution in the form assigning routes to cars based on the estimated travelling time. These times are computed using sensors that are placed on the road. They measure the traffic flow and send the readings to a centralized server which then uses these values to calculate the estimated arrival times for a vehicle travelling at a certain speed from a certain point A to point B. However the scope of this paper does not permit an elaborate elucidation of that approach.

II. EXISTING IMPLEMENTATIONS

It is indeed fair to assume that this traffic congestion mitigation is a well-researched topic. There have been several attempts at applying the concept of artificial intelligence to route or guide vehicles on the roads. According to the research carries out by Anfilets et al. [4] machine learning in the form of artificial neural networks can be used solve the traffic light cycle paradigm. Their results showed a significant improvement in the traffic queuing at intersections when their algorithm was applied with the queue size decreasing to a satisfactory level. This paper uses a similar approach to theirs to tackle the queuing up of vehicles at these intersections. In our case, we used floating car detection method for taking the input of the traffic compared to the image processing that was used in their case. Jansson [3] used genetic algorithm to solve the traffic intersection problem by assigning optimal drive orders depending on the present traffic parameters. His work took into consideration the pedestrians’ requests as well as trams’ request. Our work also involves assigning specific drive orders for clearing of queues but is centered on vehicular requests only. Oliveira et al. [6] used the Environment Observation Method based on Artificial Neural Networks Controller, EOM-ANN controller to optimize traffic lights cycle depending on dynamic inputs. Their simulation was conducted in SUMO (Simulation of Urban Mobility) whereas our work was simulated in PTV VISSIM. Their results also showed statistical improvements in the traffic conditions when their algorithm was applied. Nanayakkara et al. [7] developed their own GA solution to the route planning for the map of Singapore with their proposed GA using only distance information. Their algorithm is tested to find the shortest distance between any two nodes with a search space of over 1000 nodes, and is compared to ant-based route planning algorithms. Our proposed solution package aims to provide a real-time guess of the possible drive order than can be selected with the aim to end up with the least number of cars queuing up in a road intersection or end.

To supplement our algorithm, we used real time feed of the traffic which is provided by the Floating Car Detection (FCD) technology. This is a non-intrusive technique of gathering car volume data. The process involves triangulation of the vehicular position using signals pinged from the user’s phone. On this issue our work has some similarity with Eichler [8] and Fabritiis et al. [9]. The latter involved testing on the Rome’s Ring Road and used artificial neural networks and pattern matching to predict short-term arrival times of vehicles. Their work used the existing embedded Telematics platform (developed by OCTOTelematics) present in most cars in Europe to gather information about the present traffic. On Board Units (OBO) were installed in cars which relayed information to a central data system, thus providing the system with the requisite information. Due to the lack of such infrastructure in Bangladesh, we have to develop our own FCD system using a simple GPS receiver and a GSM transmitter. Our proposed version of FCD uses the existing telecommunication network to relay information to our central data base. The system analyses the density of mobile traffic to estimate traffic conditions at any point of time.

III. SYSTEM ARCHITECTURE AND IMPLEMENTATION

Our solution approach is concerned with the collection of traffic data which allows our system to understand traffic situation at any point of time and provide users with the best possible route. In modern times there has been an advent of floating cellular data (FCD) or floating car data which is gradually becoming a top choice to collect vehicle densities on roads. Our approach focuses on using this technique due to its simplicity and the fact that it doesn’t need many additional components [10], [11]. This method uses cellular network data (CDMA, GSM, UMTS, and GPRS) and thus no special devices are necessary. Instead of installing beacons or tags onto the cars, this method uses the cars themselves as floating sensors to generate information on vehicle flow. Virtually every switched-on mobile phone turns into a traffic probe and is as such an anonymous source of information. Using either GPS triangulation or the hand-over data stored by the network operators’ towers, the location of the mobile phone is determined [11]. The structure of our entire system is illustrated in Figure 1 and
The subsequent mechanism of operation is discussed in the following section namely System Implementation.

A. Algorithm and Route Calculation System

Artificial Neural Networks falls in the category of AI that deals with machine learning, i.e., the study of systems that can learn and adjust itself from the data provided. They are complex adaptive systems, meaning they can change their internal structure based on the information flowing through it [4]. The aim is to replicate the functions of a human brain by making the algorithm learn and adjust to the changing conditions. At its very elementary stage, neural networks are composed of artificial neurons, which receive inputs, process these inputs and generate outputs [6], [12]. There is an activation stage for each neuron which determines whether the said neuron will fire its output to the next layer of neurons or not. For our work, the activation stage involved a sigmoid curve. The choice for using the sigmoid curve is because it has a continuous output over the region of \((0, 1)\) non-inclusive and satisfies a property between the derivative and itself such that it is computationally easy to perform.

\[
\frac{d}{dt} \text{sig}(t) = \text{sig}(t)(1-\text{sig}(t))
\]

where, \(\text{sig}(t)\) is the sigmoid function.

Figure 2 demonstrates the hierarchies of our algorithm. Each layer of neurons fire an output depending on the activation function and the outputs are propagated through to the final node. The inputs and outputs are normalized so that they are continuous in the range of 0 to 1.

There are weights associated to each neuron such that the inputs of these neurons are multiplied by the weight before being propagated forward. These weights are what give our neural network its adaptive ability as it is not feasible to change the internal structure of the network. For the initial run of the system, the weights of each node are randomly generated in the range of \([-1, +1]\). These weights are then adjusted so that the output is closer to the expected output.

For this to happen, we implemented the supervised learning technique on our network.

Supervised learning is essentially a strategy that involves the user to be smarter than the network that he is teaching [12]. The user has to provide the set of sample inputs for the system to run through, with the outputs being known by the user. The network will make its guesses which will be compared with the outputs of the user and the difference between them will allow the network to make the adjustments. The above process is known as back propagation, where the outputs of the network are generated in the same pattern as a neuron [13]. The difference here is that the error, that is the difference between the actual output and the expected, correct output, is passed through additional layer of neurons and are hence propagated backwards. This results in the weights being adjusted to produce something close to our expected output. For each set of inputs, the network was allowed to undergo 500 iterations before approximating towards the expected output. The output of the network was normalized and thus came in the region of 0 to 1. The output was interpreted to provide the drive order, which gave the duration of the green light cycle.

Fig. 2 The developed neural network algorithm hierarchies

![Fig. 2 The developed neural network algorithm hierarchies](image)

In our neural network, we used one hidden layer, one input layer and one output layer. There were four input nodes corresponding to the four levels of threshold (to be discussed later) in each roads. These inputs are connected to the nodes of the hidden layers. The first hidden layer contains 4 nodes while the second layer contains two nodes. The final layer has the singular node which will provide the drive order for the traffic intersection. To ensure that our algorithm does not fall prey of being bottled in a local minima, we used biased weights in each of the hidden layers. The reason behind using the 2 strata hidden layer was justified by to make the network more adaptable to the changing conditions of the roads. These strata can be increased along with the nodes they contain depending on the response of the system. For our data, this model worked aptly [13]. Normally, for static traffic light controllers, STLCs, there is rarely an optimal green cycle meaning that there are large queues of cars to be cleared. This can be mathematically defined as:

\[
QL = Q_{tot} - Q_{clc}
\]

Where, \(Q_{tot}\) is the total number of cars in queue,
\(Q_{clc}\) is the number of cars cleared and
\(QL\) is the number of cars remaining.

Fig. 3 Typical traffic light cycle

![Fig. 3 Typical traffic light cycle](image)
Thus an additional length of the green light cycle is required to clear the QL amount of cars [14]. By the time of the next cycle, this value is slated to increase meaning that it is likely that QL >0 even after the next cycle. However, with the deployment of dynamic traffic light controllers, DTLCs, the value of QL is expected to fall with the system actively assigning drive orders to clear the queues. In our system, we issued drive orders to clear out the excess traffic at the intersections.

B. System Implementation

Our desired outputs will thus be a combination of drive orders for each intersection. From the output which was in the range of 0 to 1 non-inclusive, we decoded the drive order. In our case, it is simply taking the output and multiplying it by a factor of 15 and rounding the value to provide the drive order. This particular network is expected to take inputs from the sensors place on the road and provide the optimal drive order at any time of the day. Figure 4 illustrates a list of possible drive orders [5]. Since the traffic is dynamic and prone anomalous behavior from time to time, our network has several drive orders. Since it is an AI, it is expected to learn and improve its decision making and even provide newer drive orders from the ones it has learned already. However, that would only happen after prolonged exposure to the traffic and through the accumulation of several traffic data. Thus, it is one that is not expected instantaneously.

The proposed system takes in car flow rate or in other words, volume of cars displacing a certain point at a given time using Floating Car Detection (FCD), which is a modern vehicle count method [10], [11], [15]. Data is fed to the system as vehicle volume indices that denotes the traffic jam condition at a specific intersection or node, on a scale of 0 to 4 with 0 meaning a ‘no jam’ situation and 4 classifying a total deadlock. Data would be sent to a centralized server through 3G networks for further processing. The centralized server would process the vehicle volume indices and provide the result of the route requiring shortest time to the end user by the means of an overhead display on the car dashboard and again via 3G.

The drive order defining program was implemented in Eclipse IDE using the popular programming platform, Java. The inputs to the software are the sensor readings corresponding to volumes of traffic at road intersections and the outputs are the drive orders that would determine the state of the traffic signal lights. The inputs can be generic user inputs or can be dynamic real-time sensor inputs from roads (FCD).

![Fig. 5 The developed drive order generator application](image1)

![Fig. 6 Illustration of dynamic real time vehicular data (FCD)](image2)
IV. RESULTS AND ANALYSIS

Due to the constraints of carrying out a large scale research, we tested our system by working with a small scale model. We tried to replicate real-time road conditions as much as possible to evaluate the integrity and functionality of our system. We carried out a twofold approach to test the viability of the developed system. First, we attempted to carry out a vehicle simulation using a road traffic simulation software and tested the effectiveness the developed application. In order to assess the efficiency of our system further, we have tried to demonstrate the possible behavior of road traffic because of the selection of drive orders on a PVC board with LED strips to show the limited scale viability of the approach.

A. Road Traffic Simulation Results

The simulation was carried out in PTV VISSIM software which is used worldwide for vehicular road network modeling. The software has capabilities of placing road traffic detectors for FCD and incorporates the ability to manipulate traffic signals on the fly making it very suitable for our simulation. Figure 7 shows a screenshot of the application.

![Fig. 7 Simulation running in PTV VISSIM (Student Version)](image)

We chose a section of Dhaka city to be the subject of our experiment and the regions include Banasree, Hatirjheel, Gulshan 1, Gulshan 2, Mohakhali, Kakoli, Badda, Pragati Sharani, Mirpur Matikata, Mirpur Cantonment, Jahangir Gate and Tejgao. We ran the simulation and observed the typical vehicle flows. Next, we fed the traffic volume data into the application we had developed earlier and generated a set of drive orders. Having completed that, we halted the ongoing simulation and manipulated the traffic signals with the generated drive order. We observed the ready change and found the reduction of vehicle travel times. 10 samples were taken from 10 separate runs and averaged to lower discrepancies. Table I provides the vehicle travel times extracted from the PTV VISSIM.

<table>
<thead>
<tr>
<th>Route</th>
<th>Before applying the generated drive order</th>
<th>After applying the generated drive order</th>
</tr>
</thead>
<tbody>
<tr>
<td>Banasree-Gulshan1-Mohakhali-Mirpur</td>
<td>19.1s</td>
<td>6.0s</td>
</tr>
<tr>
<td>Banasree-Gulshan1-Mohakhali-Mirpur Matikata</td>
<td>35.4s</td>
<td>21.7s</td>
</tr>
<tr>
<td>Banasree-Gulshan2-Mirpur</td>
<td>84.0s</td>
<td>56.0s</td>
</tr>
</tbody>
</table>

B. Demonstration of drive orders on PVC board

For our demonstration purpose, we imitated the road traffic on a PVC board by consisting of 200 white LEDs. A set of red, yellow and green LEDs were used to represent the traffic lights. Sensor readings were emulated using sets of LDRs placed over traffic stops. The readings correspond to pre-calibrated list of vehicle volume index on a scale of 0 to 4 with 0 indicating the most relaxed state and 4 denoting a jam-packed situation. The intermediary device used to interface the drive order determining application in the computer with the LDRs on the PVC was an Arduino Mega microcontroller development board.

![Fig. 8 PVC board with LED strips to imitate the roads](image)

Now that we had made an interface with the computer application, we generated a random flow of vehicles with the help of the microcontroller. The LDRs gave readings emulating FCD sensors and these were dynamically fed to our application. Drive orders were generated which were then fed back to the microcontroller to light the corresponding traffic signal LEDs and block or allow vehicles to pass intersections. What we found was that 5 out of 7 intersections had vehicle volume indices of less than or equal to 2 while the rest two had 3. When we took out the drive order generating application and used some manual traffic light sequences, 7 intersections had vehicle volume indices of 3 while the last one had 4. The results here are self-conclusive and from our observations, we can conclude that our system possesses adequate level of efficiency in pushing road traffic in real-time applications.
V. CONCLUSION

Curtailing road traffic congestions requires extensible effort from all ends and it is never a viable deal to stick to only a one single approach to achieve this goal. What we have presented was a small demonstration of how introducing a drive order system can help to reorder traffic flows and in the process bring down the magnitude of traffic jams significantly across developing countries like Bangladesh. In the short scale, our solution helps in removing the unnecessary clogging that frequently bogs roads and on the broader perspective the process offloads the pressure from the jam packed road channels and hence reduces the overall traffic ordeal. Situations like gridlocks can also be avoided as roads are being constantly monitored by sensors. It is also apparently clear that the much of the work of traffic police have also been reduced in this package as traffic light sequences are fully automated in this system and the traffic light controllers no longer need to be manipulated manually to cater to abnormal traffic situations. We are hopeful and feel passionate to extend our work further and also see a practical implementation of our system in the near future.

ACKNOWLEDGEMENT

The evaluation of the developed system was carried out in the Student Version of PTV VISSIM which is a microscopic multi-modal traffic flow simulation software package developed by PTV (Planung Transport Verkehr) AG in Karlsruhe, Germany. The application provided us an opportunity to test the efficiency and plausibility of our system. The authors would like to acknowledge PTV AG for providing us this application for our research purpose.

REFERENCES