Smart Traffic Light Using YOLO Based Camera with Deep Reinforcement Learning Algorithm

Mochammad Sahal
Department of Electrical Engineering
Institut Teknologi Sepuluh Nopember
Surabaya, Indonesia
sahal@ee.its.ac.id

Zulkifli Hidayat
Department of Electrical Engineering
Institut Teknologi Sepuluh Nopember
Surabaya, Indonesia
zulkifli@ee.its.ac.id

Yusuf Bilfaqih
Department of Electrical Engineering
Institut Teknologi Sepuluh Nopember
Surabaya, Indonesia
bilfaqih@ee.its.ac.id

Mohamad Abdul Hady
Department of Electrical Engineering
Institut Teknologi Sepuluh Nopember
Surabaya, Indonesia
hady@bme.its.ac.id

Yosua Marthin Hawila Tampubolon
Department of Electrical Engineering
Institut Teknologi Sepuluh Nopember
Surabaya, Indonesia
marthinhawila32000@gmail.com

***Abstract*—Congestion causes many losses, such as in terms of time, economy, and the psychology of road users. One of the causes of congestion is that traffic lights are not adaptive to the dynamics of traffic flow. This paper tries to solve this problem by proposing a smart traffic light system that can adapt to traffic conditions. The method used is reinforcement learning applied to the Simulation of Urban Mobility (SUMO) traffic simulator. The data used is real video data from the Kedung Cowek intersection in Surabaya. The video data is processed using the You Only Look Once (YOLO) algorithm, which will detect and count vehicles. The output of the video processing will be used in reinforcement learning. The result is that traffic lights can be adapted to the number of existing vehicles, and there is a reduction in the length of traffic queues from 06.00 to 09.00 from an average of 110 vehicles to 106 vehicles.**

Keywords—reinforcement learning, smart traffic light, YOLO

I. INTRODUCTION

Traffic jams are a common problem that often occurs in big cities. This congestion causes a lot of losses, such as in terms of time, the economy, and even the psychology of road users. Based on research conducted by INRIX in 2021 [1], the city of London (England) became the most congested city with a total time loss of 148 hours, followed by Paris (France) with a time loss of 140 hours. Meanwhile, in Indonesia, Surabaya became the most congested city with a time loss of 62 hours, followed by Jakarta with a time loss of 28 hours. In 2020 [2], in New York, congestion caused 100 hours of lost time with a loss of $1,486 per driver and a total city loss of $7.7 billion. From a psychological point of view, congestion can cause stress and aggressive behavior.

A road is said to be congested if the flow of traffic passing through it exceeds the planned capacity of the road, which results in the speed of the freeway approaching 0 km/hour or even being 0 km/hour, causing a queue [3]. In describing traffic, it can be done by knowing the characteristics of the traffic [4],[5]. Traffic characteristics are based on three things: flow or volume, speed, and density. An intersection is an area where two or more roads merge or intersect, including roads and roadside facilities for traffic movement in them [6]. From this explanation, it can be concluded that the causes of congestion include road narrowing, traffic density, and traffic lights. Traffic lights are considered one of the causes of congestion due to their inability to adapt their duration to the dynamics of traffic flow. Most existing traffic light cycles are fixed cycles, i.e., the traffic light duration is fixed in all conditions. Meanwhile, the number of riders continues to grow every year. The inability of traffic lights to adapt causes vehicle encounters on roads and causes congestion.

For this reason, a technology called "smart traffic lights" is proposed. Smart traffic lights are a technology where traffic lights can intelligently adapt according to traffic conditions. In [7], smart traffic light innovations are divided into 3 main parts, namely, to reduce congestion, for emergency vehicles, and for pedestrians. This system will be able to determine the duration of the green light for each road segment so as to reduce congestion. To be able to do this, the system must be able to calculate the volume and density of traffic. Then the system learns from the input volume of the vehicle to produce the duration of the traffic light. In calculating the volume of vehicles, the camera is used as a medium, and the YOLO algorithm is used for automatic identification and calculation. And for learning, deep reinforcement learning algorithms are used.

YOLO is an algorithm that uses a neural network to detect and recognize various objects in an image. YOLO divides the image into fixed-sized squares. For every object in the image, there will be a box that is responsible for predicting it [8], [9]. Reinforcement learning (RL) is part of machine learning that optimizes decision-making based on stages [10]. Reinforcement learning uses rewards and punishments as signals for positive and negative actions. The goal of RL is to find a suitable action model that will maximize the total cumulative reward of the agent. The interaction will continue until a terminal state is reached or the agent meets the termination conditions [11], [12].

II. SYSTEM DESIGN

A. Data Used

The data to be used is real traffic data from one of the intersections in Surabaya requested from SITS (Surabaya Intelligent Transport System) by Surabaya City Transportation Service. The data used is in the form of traffic CCTV videos. The intersection that is used as the data is the Kenjeran Kedung Cowek intersection. The intersection

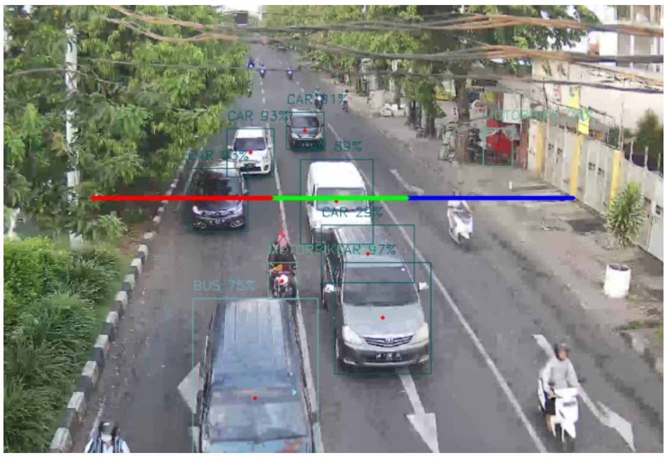

Fig. 1. Auxiliary line for calculating and determining routes.

consists of four roads, namely Jalan Kedung Cowek on the north side, Jalan Putro Agung on the south side, and Jalan Kenjeran on the east and west sides. The road in the north consists of four lanes, where the left lane is not affected by traffic lights (turn left directly). The road to the east consists of three lanes, with the left lane marked ‘turn left straight away’. The road in the south consists of three lanes, with all lanes affected by traffic lights. The road in the west consists of three lanes, with the left lane marked ‘turn left straight’. Through the data received from SITS, the duration of the traffic light phase at the Kedung Cowek intersection is as follows:

- The north side has a green light duration of 70 seconds.
- The south side has a green light duration of 48 seconds.
- The east side has a green light duration of 48 seconds.
- The west side has a green light duration of 44 seconds.

So the total time for one cycle is 210 seconds.

B. Video Processing Program Design

Vehicle counting is needed to determine traffic characteristics, namely, to determine traffic flow and volume. In general, vehicle counting is done manually, namely by directly observing traffic conditions and counting vehicles using a counter. In this paper, vehicle counting is not done manually. The vehicle count is carried out by utilizing the traffic CCTV at the crossroads. The CCTV videos that have been obtained will later be processed into a program. The program will be tasked with identifying, calculating, determining routes, and calculating the time the vehicle is detected. Programs are created using the Python programming language.

1. Vehicle identification

For identification, the YOLO algorithm is used. YOLO uses a model that has been trained beforehand. This training model can be made yourself or by using an existing one. If you create your own model, you can determine how many classifications to use by creating a dataset based on the desired classification and training it so that at the end you get a model that can be used for classification. However, we do not create our own model but use an existing model (pretrained). The model we use is a pretrained YOLOV4 model made by Alexey Bochkovskiy [13]. This model can classify 80 classes. Of the 80 classes, only 4 were used, namely cars, motorcycles, buses, and trucks. This class was chosen because, in general, the streets of Indonesia are filled with these four classes.

When the program is executed, the video will run frame by frame. Each frame will be input into the YOLO model. YOLO will produce an image that has been identified based on the existing class and its bounding box. Each object that has been identified is then chosen based on its accuracy. This choice also reduces the existing bounding box. After this process, an image is generated that has a bounding box on each detected vehicle object that has a level of accuracy exceeding a set limit. In this case, the object's identification accuracy limit is 20%.

2. Vehicle count

Objects that have been identified will be determined at the midpoint based on the bounding box. This midpoint will be used to calculate the vehicles. Previously, from the detection results to the object identification, the bounding box and its midpoint were displayed as output for the current frame. For the next frame, the same thing will be done until identification is obtained along with the bounding box and the midpoint. For different frames, there may be the same image, so another program is used to perform tracking. This track is based on the Pythagorean theory. The distance calculation is done using the midpoint value that has been obtained previously. If the distance obtained is less than a certain value — which in this case is 25 pixels — then the object in that frame is the same as the object from the previous frame. If YOLO can detect an object well, then the object will be detected as the same object for each frame until the object is no longer visible in the video.

By tracking the object, the object or vehicle can be counted. To calculate the vehicle used auxiliary line. This guideline will serve as a differentiator between the vehicles that have been counted and those that have not. Because all CCTV videos are traffic videos where the vehicle moves closer to the CCTV, the vehicle count is as follows: As shown in Fig. 1, the auxiliary line is placed in the center of the video dimension. If the position of the object is detected above the auxiliary line, then it is tracked and passed the guide line, or more specifically, the midpoint of the bounding box past the guide line, then the object will be counted. The calculated object will be stored in a list in the program.

3. Route determination

The route of the vehicle is also one of the variables that will be used in the future in the system. At the intersection used as data, there are a total of 12 vehicle routes. The list of routes is as follows:

- Route 0: route from West to East;
- Route 1: route from West to South;
- Route 2: route from West to North;
- Route 3: route from South to North;
- Route 4: route from East to West;
- Route 5: route from South to East;
- Route 6: route from East to West;
- Route 7: route from East to South;
- Route 8: route from East to North;
- Route 9: route from North to South;
- Route 10: route from North to West;
- Route 11: route from North to East.

Each road segment has three routes. The routes are routes to the left, right, and straight. Each route on each segment is represented by each lane on that road segment. Except on the north road, which has four lanes. On the north road, two lanes are used for routes to the right, and the remaining two lanes are for straight and left-turn routes.

To determine the route is not much different from the calculation of vehicles, namely using a guide line. The auxiliary line used is the same auxiliary line as the vehicle counting auxiliary line. It can be seen in Fig. 1 that the auxiliary lines have three different colors. The colored line is used to determine the route of the vehicle. The red line defines the route to the right, the green line defines the straight route, and the blue line defines the route to the left. If the object above the auxiliary line passes through the auxiliary line, and the coordinates of the object's center point are between one of the three colored lines, for example, between the red lines, then the object is determined to have a route to the right.

4. Detection timer

Time is also one of the variables that will be used. In this case, the timing is the time when the vehicle crosses the auxiliary line. This time will later be used in the simulation to determine when a vehicle appears in the simulation environment. To calculate the time, we used the library timeit. This library is used to calculate the execution time of a program. This library will serve as a timer when the program is executed. At the start of program execution, the timer starts counting. Then if there is a vehicle that crosses the auxiliary line, then the program will save the number of seconds the vehicle passed the help line. The timer will continue to run until the program execution is complete. And during that time, the program will save the time of each vehicle that crosses the auxiliary line.

5. The final result of video processing

Required data such as vehicle type, vehicle route, and vehicle time is stored in different list variables. After the video is finished executing, these variables will be saved in a .csv (comma separated value) file format. Later, this video processing data will be used as input in the simulation and will be trained with reinforcement learning.

C. Simulation Design

At this stage, a traffic simulation is designed that approximates the existing traffic conditions in Indonesia, especially the observed traffic intersection conditions. The simulation uses the Simulation of Urban Mobility (SUMO) application developed by the Eclipse Foundation [14].

1. Traffic network design

When installing the SUMO application, a special application will also be installed for designing traffic networks and passing vehicles. The application is called Netedit. In Netedit, you can create various traffic networks, from simple to complex. Because this paper only simulates intersections, the appropriate traffic network is created. The traffic network created has an intersection that has four roads. Where each road segment has a two-way lane. And each lane has three lanes. The length of each road segment is 100 m. Generally, the traffic created using Netedit is right-hand driving traffic. Meanwhile, Indonesia applies to left lane driving traffic. So, in Netedit, the settings are made to be the left path.

In this design, travel routes that can be taken by vehicles that will pass are also arranged. Each lane on the road can be set to indicate where and from which direction the vehicle is moving. The route in this simulation is made as close as possible to the real condition of the observed traffic. It can be seen in Fig. 2, where each lane on each segment has a direction where the vehicle will go.

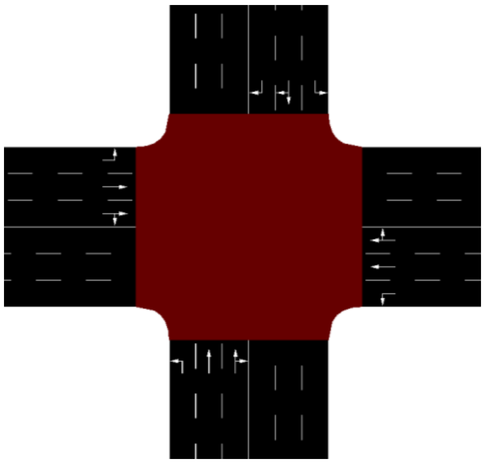
Fig. 2. Route design for each lane on each road segment.

2. Passing vehicle design

In Netedit, you can design which vehicles will pass through that traffic. There are four classes of detected vehicle objects, the vehicle design is designed according to the existing classes. The dimensions of each vehicle class are taken from the average value of various types of vehicles in the same class. The dimensions of each designed vehicle are:

- Motorcycle: 2.0 m long, 0.8 m wide;
- Car: 4.2 m long, 1.8 m wide;
- Bus: 9.0 m long, 2.1 m wide;
- Truck: 7.0 m long, 2.0 m wide.

In this design, the number of vehicles that will be simulated is also designed with the route of each vehicle and the time the vehicle appears in the simulation. This is where the data that has been obtained previously from video processing is used. The previously obtained.csv format data will be converted into data that can be understood by the simulator application. The vehicle that will be simulated on the simulator is a video processing vehicle, which is the original data.

The vehicle route will be the same as the previously described route. In this design, assumptions are also made to facilitate the simulation. Information regarding these assumptions cannot be obtained from the previous program. The assumption is that the speed of each vehicle when entering the intersection is "10". From the previous program, it is not possible to determine the speed of the vehicle. With the option "10", which means 10 m/s in the simulation, the speed of each vehicle when entering the simulation is 10 m/s. Another assumption is that the vehicle arrival path when entering the simulation is "best". The meaning of this assumption is that the vehicle will enter the simulation according to its route path. If the vehicle has a route to the left, then the path taken is the left lane; if the vehicle has a straight route, then the path taken is the middle lane; and if the vehicle has a route to the right, the path taken is the right lane.

D. Design of Reinforcement Learning

Things that are designed when doing reinforcement learning are state, action, reward, and algorithm design [15].

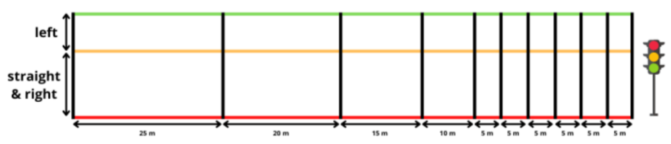

Fig. 3. Cell division on each road segment

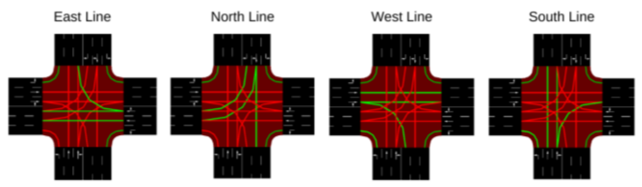

Fig. 4. Actions that can be taken

1. State design

In the system that will be made, the state that will be used as an input is the position of the vehicle. In the simulation, the length of each road segment is 100 meters. To find out the position of the vehicle, each road segment is divided into cells based on the length and route of the vehicle. In one, each road segment is divided into 20 cells. Ten cells for the left turn route, and the remaining ten cells for the straight and right turn routes. The cells are divided based on the length of the road segment. So, the total cells in the intersection are 80 cells. The division of cells in roads can be seen in Fig. 3.

2. Action design

The action to be taken is the possible green light that can be given at the intersection. Specifically, at this intersection, there are four actions that can be taken, namely:

- East line: green light;
- North line: green light;
- West line: green light;
- South line: green light.

Actions that can be taken can be seen in Fig. 4.

The duration of the lights is set as follows: the duration of the green light is set for 10 seconds, the yellow light is set for 2 seconds, and there is a light duration where all roads are red for 2 seconds. The agent will choose a possible action from the four actions. So that later the traffic lights do not take place sequentially as in the initial state. If the action taken by the agent at time t is the same as the action taken at time $t - 1$, then there is no yellow light phase and the green light phase continues. However, if the action taken at time t is different from the action taken at time $t - 1$, then between the two actions the yellow light phase will begin, followed by the red-light phase on all roads. For convenience, an example of the process of determining the duration of a traffic light can be seen in Fig. 5.

3. Reward design

Reward is the feedback that is obtained when the agent chooses an action. To be able to understand the results of the actions or actions taken and improve the model for further action. Therefore, rewards are an important aspect of the learning process. Prizes usually have two possible values: positive and negative. Positive rewards result from good actions, and negative rewards result from bad actions. In this paper, the aim is to maximize the flow of traffic from time to time to avoid congestion. The metric function used is the accumulated total waiting time for vehicles specified in (1).

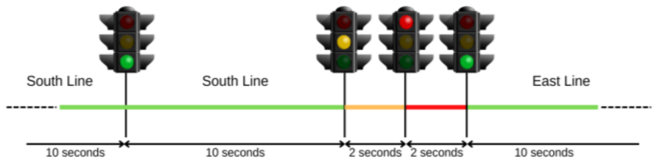

Fig. 5. An example of determining the duration of a traffic light

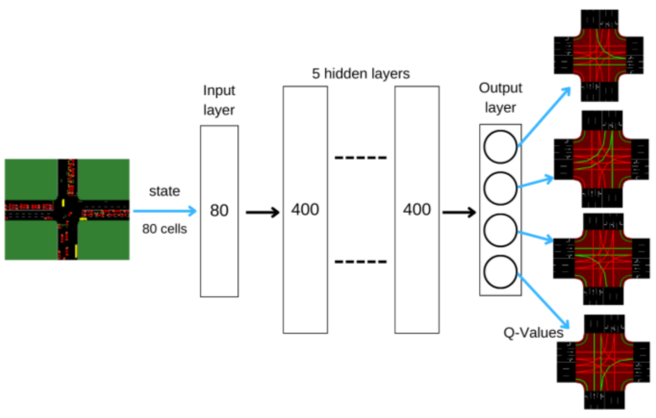

Fig. 6. Actions that can be taken

$$t_t^T = \sum_{v=1}^{N} t_{v,t} \tag{1}$$

Where $t_{v,t}$ is the amount of time in seconds since a vehicle v was called in the simulation environment that has a speed of less than 0.1 m/s at time t. N represents the number of vehicles in the neighborhood at t. Therefore, t_t^T is the accumulated total waiting time for vehicles at time t. Once the metric is defined, the reward function is defined as in (2).

$$r_t = t_{t-1}^T - t_t^T \tag{2}$$

Where r_t is the time t prize, t_t^T is the total accumulated waiting time at time t, and t_{t-1}^T is the total accumulated waiting time at time $t - 1$.

4. Deep Q-learning

The equation is used as in (3).

$$Q(s_t, a_t) = r_{t+1} + \gamma . max_A Q'(s_{t+1}, a_{t+1}) \tag{3}$$

The reward received after performing action a_t in state s_t is denoted by r_{t+1}. The value of Q associated with taking action a_{t+1} in state s_{t+1}, i.e., the next state after taking action a_t in state s_t, is denoted by the term $Q'(s_{t+1}, a_{t+1})$.

In this paper, a fully connected deep neural network is used [16], with the input layer, which is the number of states, as many as 80 neurons; 5 hidden layers, which each have 400 neurons; and the output layer, which is the number of actions of 4 neurons. A representation of a deep neural network can be seen in Fig. 6.

III. Results and Discussion

A. Video Processing Results

The video used is a video of the Kedung Cowek crossing on May 18, 2022, which was taken from the SITS (Surabaya Intelligent Transport System) Surabaya City Transportation

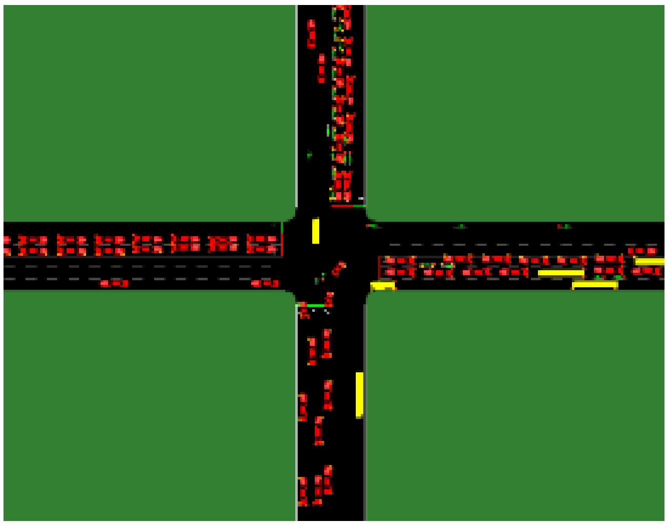
Fig. 7. Traffic simulation with SUMO

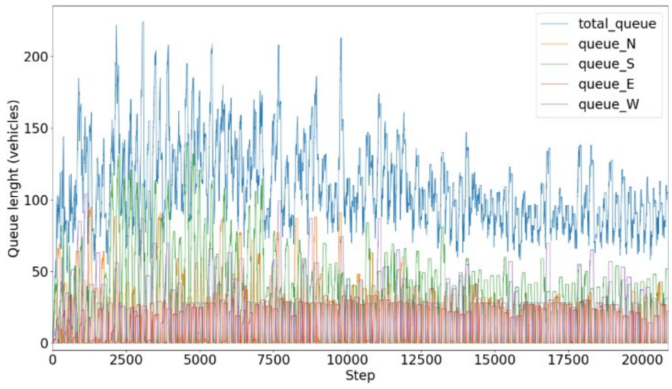

Fig. 8. The queue length of vehicles without RL model at 06 – 09

Service. In this paper, the mention of each road segment is based on the cardinal directions. The North Road Section refers to the Kedung Cowek Road section. The "South Road" refers to the Jalan Putro Agung section. The east road section refers to Jalan Kenjeran. The west road section refers to Jalan Kenjeran. The intersection has four CCTVs, so four video distributions are obtained based on each road segment. The duration of the video to be processed is from 06.00 am to 06.00 pm. The duration of the video is further divided into three hours. So, there are four videos that will be processed for each segment, and there are a total of sixteen videos in total.

The calculated vehicles will be saved in a.csv format file. Based on the total processed videos, there are also sixteen csv files containing data. However, to enter the simulation, the files must be merged based on the clock. For example, northern data at 06.00–09.00 is applied to southern data, eastern data, and western data at the same time. So there will be four large files in.csv format. The data stored is name, class, direction, and time. The name containing the vehicle id serves to distinguish one vehicle from another. A class is a type of vehicle consisting of four types of vehicles, namely cars (cars), motorcycles (motorcycles), buses (buses), and trucks (trucks). "Direction" is the direction of movement in which the vehicle will go. The Direction contains the route for each vehicle. Time is the time when the vehicle is counted.

B. Results of Reinforcement Learning

The vehicle count data obtained previously was used as input in the simulation. The appearance of the simulation

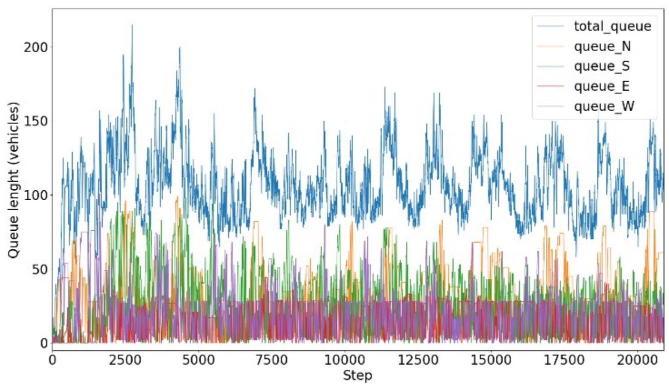

Fig. 9. The queue length of vehicles using RL model at 06-09

carried out with the SUMO application can be seen in Fig. 7. The reinforcement learning training was carried out four times. This is done because there are four data points. Later, the model generated from the training with the first data will be used in the training with the second data. And the model generated from the training with the second data will be used in the next model training. And so on until the training with the latest data. This is done so that the model is more trained for the various existing datasets. For each model trained with data, there are 25 episodes. And each episode consists of 800 epochs. In the results of the training, there was an increase in the reward from the first training of -700000 up to the last training of -400000.

Furthermore, the model that has been obtained from the training will be tested. The test is carried out using one of the datasets that has been obtained, namely data at 06.00-09.00. Previously, the data would be simulated first without using the model that had been obtained. This is done to see how the queue length graph works before using the model. After that, the same data will be simulated using the model that has been obtained. The results of the queue length without a model and the results of testing using the model can be seen in Fig. 8 and Fig. 9. It can be seen in the graph without a model that the queue length in each segment tends to go up to a point, then stay at that point for a few seconds, and then go down again. This happens over and over again. This is due to the constant cycle of traffic lights. So, when the traffic light is red, the vehicles begin to pile up, increasing the density of vehicles and causing congestion. However, if you look at the test graph of the model that has been trained, the accumulation of vehicles can be minimized. It can be seen that if the queue of vehicles is high enough, then the green light will be given on the lane immediately, so the queue of vehicles does not last long enough. From the data, it is also found that the overall average queue length of the simulation without using the RL

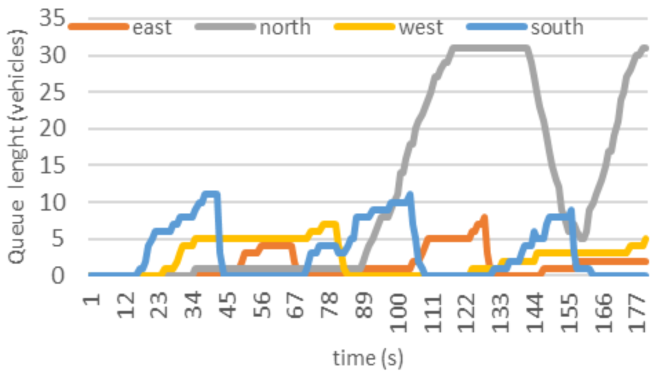

Fig. 10. The queue length in the first three minutes

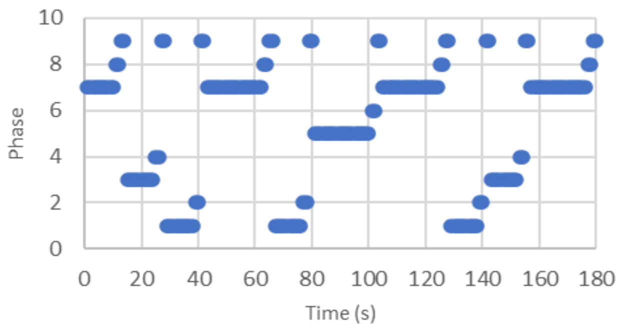

Fig. 11. Light phase in the first 3 minutes. Each number represents a phase in traffic. (1) for the green phase in the east. (2) for the yellow phase in the east. (3) for the green phase in the north. (4) for the yellow phase in the north. (5) for the green phase in the west. (6) for the yellow phase in the west. (7) for the green phase in the south. (8) for the yellow phase in the south. (9) for the red phase on all sides.

model is 110 vehicles, and when using the RL model it is 106 vehicles.

To determine the duration of traffic lights, samples were taken from the first 3 minutes. As explained in the previous chapter, traffic lights do not occur sequentially. The system will choose according to the results of observations that have been made in the previous training. The graph of the queue length by vehicle unit can be seen in Fig. 10. Also, the graph of the given traffic light phase can be seen in Fig. 11.

It can be seen that the model has been able to determine the duration of traffic lights quite well. Although it can be seen that there is still a significant increase in queues on one side of the intersection. This is due to the lack of training data, so that the training results have rewards that are still far from positive.

IV. Conclusion

In this paper, the vehicles in the video can be calculated using the YOLO algorithm. These vehicles are calculated based on four classes, namely cars, motorcycles, buses, and trucks. and can be tracked so that it can determine the route of each vehicle. Then the results of the video processing are used as input in the simulation to be trained with the reinforcement learning algorithm.

The results of the reinforcement learning training coupled with the SUMO simulation show that the system can be adaptive in determining the duration of traffic lights based on the number of vehicles on the road. Thus, it can reduce the length of the queue, such as on road conditions at 06.00–09.00, where a queue length of 110 vehicles before using the RL was reduced to 106 vehicles after using the RL.

In the future, it is recommended to use an algorithm that can detect and identify vehicles more quickly and accurately. And go deeper into the SUMO simulation application so that the simulation that is carried out is closer to real traffic conditions. It is also recommended to have more traffic data so that training and testing can be done better and the reward value in the RL model can be increased.

References

[1] B. Pishue, “2021 INRIX Global Traffic Scorecard,” p. 21, 2021.
[2] B. Pishue, “2020 INRIX Global Traffic Scorecard,” p. 23, 2020.
[3] G. Nurinda Abdi, S. Priyanto, and S. Malkamah, “HUBUNGAN VOLUME, KECEPATAN DAN KEPADATAN LALU LINTAS PADA RUAS JALAN PADJAJARAN (RING ROAD UTARA), SLEMAN,” *Teknisia*, vol. XXIV, no. 1, pp. 55–64, May 2019, doi: 10.20885/teknisia.vol24.iss1.art6.
[4] E. N. Julianto, “HUBUNGAN ANTARA KECEPATAN, VOLUME DAN KEPADATAN LALU LINTAS RUAS JALAN SILIWANGI SEMARANG,” p. 10, 2010.
[5] Dharmawan -, Rahmadwati Rahmadwati, Hadi Suyono, " Implementation of Background Subtraction and Fuzzy Logic Control for Green-Timing Optimization on 3-Junction Traffic Light, JAREE (Journal on Advanced Research in Electrical Engineering), Vol.3 No.1, 2019.
[6] J. D. Ansusanto and S. Tanggu, “ANALISIS KINERJA DAN MANAJEMEN PADA SIMPANG DENGAN DERAJAT KEJENUHAN TINGGI,” vol. 12, no. 2, p. 8, 2016.
[7] A. N. Aulia Yusuf, A. Setyo Arifin, and F. Yuli Zulkifli, “Recent development of smart traffic lights,” *IAES Int. J. Artif. Intell. IJ-AI*, vol. 10, no. 1, p. 224, Mar. 2021, doi: 10.11591/ijai.v10.i1.pp224-233.
[8] T.-N. Doan and M.-T. Truong, “Real-time vehicle detection and counting based on YOLO and DeepSORT,” in *2020 12th International Conference on Knowledge and Systems Engineering (KSE)*, Can Tho, Vietnam, Nov. 2020, pp. 67–72. doi: 10.1109/KSE50997.2020.9287483.
[9] A. Gupta, A. Verma, A. Yadav, and A. M, “YOLO OBJECT DETECTION USING OPENCV,” *Int. J. Eng. Appl. Sci. Technol.*, vol. 5, no. 10, Feb. 2021, doi: 10.33564/IJEAST.2021.v05i10.036.
[10] J. D. Martín-Guerrero and L. Lamata, “Reinforcement Learning and Physics,” *Appl. Sci.*, vol. 11, no. 18, p. 8589, Sep. 2021, doi: 10.3390/app11188589.
[11] S. S. Mousavi, M. Schukat, and E. Howley, “Traffic Light Control Using Deep Policy-Gradient and Value-Function Based Reinforcement Learning.” arXiv, May 27, 2017. Accessed: Jun. 12, 2022. [Online]. Available: http://arxiv.org/abs/1704.08883
[12] X. Liang, X. Du, G. Wang, and Z. Han, “Deep Reinforcement Learning for Traffic Light Control in Vehicular Networks,” *IEEE Trans. Veh. Technol.*, vol. 68, no. 2, pp. 1243–1253, Feb. 2019, doi: 10.1109/TVT.2018.2890726.
[13] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, “YOLOv4: Optimal Speed and Accuracy of Object Detection.” arXiv, Apr. 22, 2020. Accessed: Jun. 13, 2022. [Online]. Available: http://arxiv.org/abs/2004.10934
[14] M. Semrau, J. Erdmann, B. Friedrich, and R. Waldmann, “1 Simulation framework for testing ADAS in Chinese traffic situations,” p. 11, 2016.
[15] A. Vidali, L. Crociani, G. Vizzari, and S. Bandini, “A Deep Reinforcement Learning Approach to Adaptive Traffic Lights Management,” p. 9, 2019.
[16] K. G. Kim, “Book Review: Deep Learning,” *Healthc. Inform. Res.*, vol. 22, no. 4, p. 351, 2016, doi: 10.4258/hir.2016.22.4.351.

MOCHAMMAD SAHAL received the B.S. degree in electrical engineering from Institut Teknologi Sepuluh Nopember (ITS), Indonesia, in 1997, and the M.S. double degree in electrical engineering from Institut Teknologi Sepuluh Nopember (ITS), Indonesia, and University of Applied Science Darmstadt, Germany, in 2007. His research interest includes signal processing, multi agent system, and control system engineering. He is currently a Lecture in the control and system engineering with the Department of Electrical Engineering, ITS. And he is currently an academic staff in the Department of Electrical Engineering, ITS.

ZULKIFLI HIDAYAT received the B.S. degree in electrical engineering from Institut Teknologi Sepuluh Nopember (ITS), Indonesia, in 1997, and the M.Sc. degree in control system engineering from the University of Twente, Netherlands, in 2004. His research interest includes data-driven system modeling and simulation. He is currently an academic staff in the Department of Electrical Engineering, ITS.

YUSUF BILFAQIH received a Bachelor's degree in Electrical Engineering from Institut Teknologi Sepuluh Nopember (ITS), Indonesia, in 1997, and a Master's degree in Control Systems

Engineering from ITS in 2008. His research interests are in system design and development including system modeling and simulation. Currently, he is head of the System and Cybernetics laboratory in the Department of Electrical Engineering, ITS.

MOHAMAD ABDUL HADY received the B.Eng. degree in electrical engineering from Institut Teknologi Sepuluh Nopember (ITS), Indonesia, in 2011, and the M.Eng. degree in control system engineering also from ITS Surabaya, in 2014. His research interest includes intelligent adaptive control. He is currently an academic staff in the Department of Electrical Engineering, ITS.

YOSUA MARTHIN HAWILA TAMPUBOLON received the B.S. degree in electrical engineering from Institut Teknologi Sepuluh Nopember (ITS), Indonesia, in 2022. He has an interest in machine learning, artificial intelligence, and data analysis.