ITS Autonomous Car Maneuver at Roundabouts or U-Turn using Deep Reinforcement Learning

1st Prof. Dr. Ir. Mauridhi Hery Purnomo, M.Eng.
Department of Computer Engineering
Faculty of Intelligent Electrical
and Informatics Technology
Institut Teknologi Sepuluh Nopember
Surabaya, Indonesia 60111
hery@ee.its.ac.id

2nd Dr. I Ketut Eddy Purnama, ST., MT.
Department of Computer Engineering
Faculty of Intelligent Electrical
and Informatics Technology
Institut Teknologi Sepuluh Nopember
Surabaya, Indonesia 60111
ketut@ee.its.ac.id

3nd Muhtadin, ST., MT.
Department of Computer Engineering
Faculty of Intelligent Electrical
and Informatics Technology
Institut Teknologi Sepuluh Nopember
Surabaya, Indonesia 60111
muhtadin@te.its.ac.id

4rd Muhammad Roychan Meliaz
Department of Computer Engineering
Faculty of Intelligent Electrical
and Informatics Technology
Institut Teknologi Sepuluh Nopember
Surabaya, Indonesia 60111
meliaz.17072@mhs.its.ac.id

Abstract*—An autonomous car or autonomous vehicle is a vehicle which has the ability to drive independently as if controlled by humans using a series of artificial intelligence. In this study, we propose research on the development of an autonomous vehicle iCar ITS (Intelligent Car Institut Teknologi Sepuluh Nopember) by developing a maneuvering system for autonomous vehicles at roundabouts or u-turns in a simulated environment. In the simulation environment, the model used is a vehicle model adapted to iCar. The development of autonomous vehicle navigation and maneuvering systems is carried out using the Deep Reinforcement Learning method, a branch of Machine Learning. From this research, the results obtained are reinforcement learning models that are able to maneuver at roundabouts with intersections and roundabouts without intersections with a mean deviation value of the angle of the path of 27,011° and 30,068°, respectively, able to maneuver without collision for an average of 13.3 seconds and 7.9 seconds, and with an average speed of 27.0 kmph and 28.5 kmph.

***Index Terms*—Autonomous Vehicle, Reinforcement Learning, Deep Learning, Simulation**

I. Introduction

AUTONOMOUS vehicle technology has a long history. The first fully functional prototype was built in 1980. Using a camera, this prototype managed to cover 100km of empty roads without needing to be steered by humans. With this success came many projects in the 80s and 90s using similar systems used for driving through highways, in either light or no traffic. In its development, autonomous vehicles can solve the problem of driving safety and efficiency. Therefore, the main objective of the research is to prevent or reduce traffic accidents, reduce the time people drive, and reduce carbon emissions. [1]

One of the autonomous car products being developed is iCar ITS (Intelligent Car Institute of Technology Ten November). iCar ITS is a prototype car equipped with autonomous driving features as a result of collaborative research from ITS researchers with various fields of expertise. [2] iCar ITS is operated as a commuter car that serves passenger trips to various destinations within the ITS campus. [3]

A commonly used method for developing autonomous vehicles is reinforcement learning, a subset of machine learning that is also a subset of artificial intelligence. Reinforcement Learning is a learning method about what to do (implement actions into situations) on a problem/problem to get maximum results/reward. Agents are not given instructions on what actions to take. Agent will take action with the principle of Trial and Error, then make a decision based on the reward obtained (maximum reward).

In its application, iCar ITS still uses the traditional rule-based method without utilizing machine learning methods. This method is not good enough because the developer has to manually program every scenario that the iCar will face. This method will not last long given the number of real-world scenarios that are almost impossible to solve manually.

II. System Design and Implementation

This section describes the application of the Deep Reinforcement Learning method which aims to create an algorithm for an autonomous car capable of efficiently maneuvering roundabouts or u-turns.

A. Simulation Environment Preparation

Using the CARLA simulator, the Town03 CARLA map is used, which has a roundabout and a u-turn to facilitate research work. There are two types of roundabouts used in this study.

1) Roundabout with Intersections: The four-way roundabout is used as shown in Figure 1. In this environment, the agent will spawn from four spawn points around the roundabout.

Fig. 1: Roundabout with Intersections

2) Roundabout without Intersection: A roundabout without an intersection is used as shown in Figure 2. In this environment, the agent will spawn from an initial spawn point of the roundabout.

Fig. 2: Roundabout without Intersection

B. Sensors

In this study, a camera sensor is used which is physically placed on the lower front of the agent. The camera sensor used is a segmentation camera sensor. The segmentation camera sensor is a camera sensor that separates objects in the simulator into a variety of unique solid colors. The image generated from the segmentation camera sensor in Figure 4 is an advanced image that has been processed from the RGB image in Figure 3

C. Data Acquisition

Image data is taken from the sensor with a size of 480x270. The image used is a segmented image that has been provided by the segmentation camera sensor from the CARLA Simulator. Seen in Figure 3 is the initial image. Then segmentation is carried out on the image so that each object is presented with a solid color as shown in Figure 4

Fig. 3: RGB Image

Fig. 4: Segmented Image

There are two types of images used in this study, namely grayscale segmentation images and advanced segmentation images.

Grayscale segmentation image is a segmented image which is then given a grayscale filter in order to minimize the analyzed data but still maintain the existing features.

Fig. 5: Citra Segmentasi Grayscale

The advanced segmentation image is a segmented image, which is then re-segmented into two types of objects, namely drivable and non-drivable. Drivable objects are represented in white and non-drivable objects are represented in black.

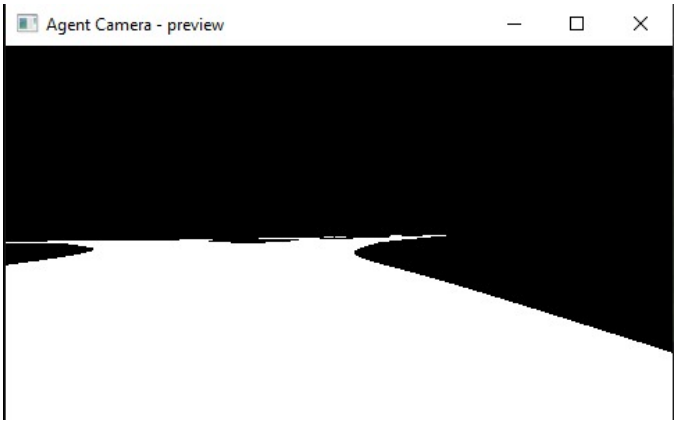

Fig. 6: Citra Segmentasi Lanjutan

Grayscale segmentation images and advanced segmentation images will be the final result of image acquisition which will then be given to the DQN algorithm to perform model training and/or inference models.

D. Action

There are 3 actions that the agent can take. Among them are:

1) `forward`
   Agent throttle with a value of 1 and steer with a value of 0. Thus the agent will perform a forward maneuver.
2) `forward_left`
   Agent throttle with a value of 1 and steer with a value of -1. Thus the agent will perform a forward left maneuver.
3) `forward_right`
   Agent throttle with a value of 1 and steer with a value of 1. Thus the agent will perform a forward right maneuver.

E. Reward Function

The reward function is designed to allow autonomous cars to move along the track quickly and safely.

1) Roundabout with Intersections: There are four rewards set. Each reward pays attention to the reference lane target. The target lane is an imaginary line that is the target of the autonomous car movement. Seen in Figure 7, the target lane is depicted with red dots around the roundabout.

Fig. 7: *Target lane*

a) *Reward* 1: Angle deviation from the target lane. The reward used in Reward 1 is 1/alpha. Alpha is the angle of the agent that deviates from the target lane. The alpha technique is described in Figure 8.

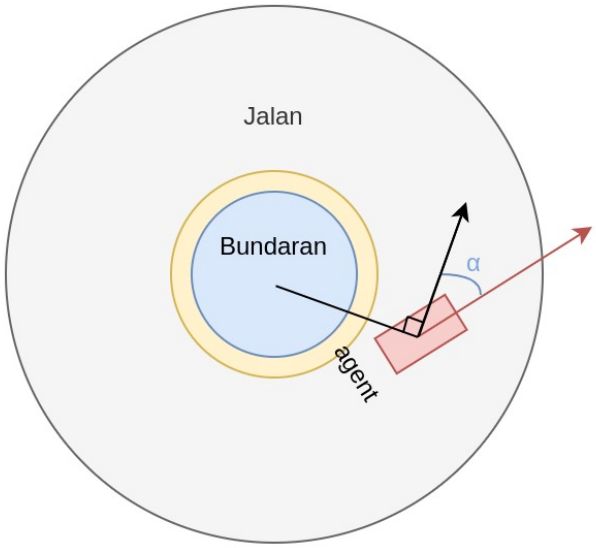

Fig. 8: Reward 1

The black line is a perpendicular vector from the center of the circle to the center of the agent. Then the red line is the direction vector of the car. From these two values, we get alpha which is the angle of the two vectors. Then the reward value is defined by 1/alpha. The highest reward value is limited to 1. This is done so that the fluctuation of the reward given to the agent is not too high for a small change in value.

b) *Reward* 2: Distance deviation from the target lane. The reward used in reward 2 is 1/distance_deviation*10. Dis-

tance deviation is the agent's smallest distance from the target lane in meters.

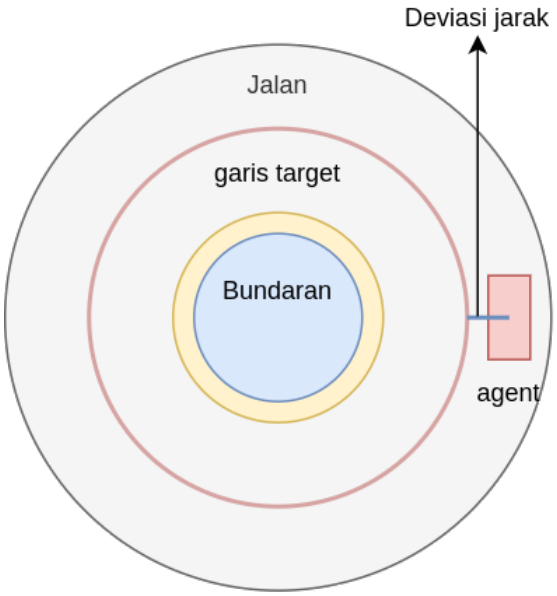

Fig. 9: Reward 2

The highest reward value is limited to 1. This is done so that the fluctuations in the reward given to the agent are not too high for a small change in value, as in Reward 1. In addition to positive rewards, negative rewards are also needed to reduce the opportunity for agents to do things that should not be done. There are two negative rewards given.

c) *Reward* 3: Collision with other objects. A reward of -1 will be given to the agent if the agent touches other objects other than roads, ground, and sidewalks.

d) *Reward* 4: Distance deviation too big. A reward of -0.5 will be given to the agent if the distance between the agent and the roundabout is greater than 30 meters. The distance of 30 meters from the roundabout can be seen in Figure 10

Fig. 10: Distance Limit

e) Total *Reward* The total reward value is defined as Reward 1 + Reward 2 + Reward 3 + Reward 4.

2) Roundabout without Intersection: At roundabouts without intersections, the reward function given is different. In this case, path points are provided as waypoints that the agent must follow.

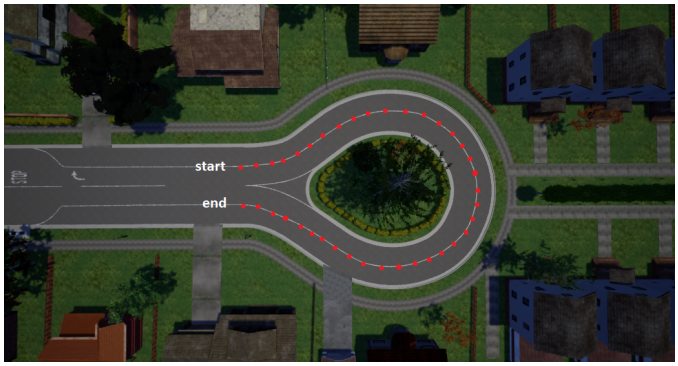

Fig. 11: *Waypoint*

Waypoints are points on the map as the target destination for the agent. Seen in Figure 11, the red dots are illustrations of waypoints. Waypoints are 100 physical points where each point is one meter apart.

a) *Reward* 1: Angle deviation from *waypoint*. Rewards are defined by 1/alpha. Alpha is the angle deviation of the agent direction to the direction of the nearest waypoint from the agent + 5.

Fig. 12: *Waypoint* from *Agent's* Side

Figure 12 shows that the closest +5 waypoint from the agent is six meters from the agent's coordinate center.

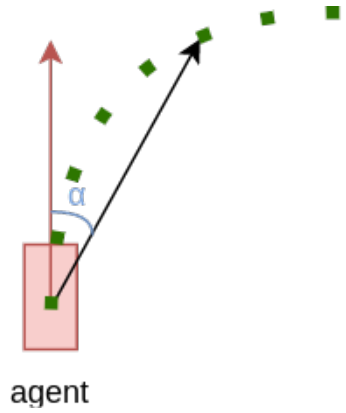

Fig. 13: Alpha Reward

The illustration of the reward function can be seen in Figure 13, where the green dots are waypoints, the red line is the direction of the agent, the black line is the direction from the agent to the waypoint+5 of the agent, and alpha is the angle difference between the two angle values. The highest reward value is limited to 1. This is done so that the fluctuations in the reward given to the agent are not too high for small changes in value, such as in Reward 1 and Reward 2 at the four-way roundabout.

b) *Reward* 3: Collision with other objects A reward of -1 will be given to the agent if the agent touches other objects other than roads, ground, and sidewalks.

c) Total *Reward* The total reward value is defined as Reward 1 + Reward 2.

F. End Episode

The maximum time given to the agent to carry out the learning process for each episode is 10 seconds.

The episode will end if the maximum time of the episode ends or the agent touches other objects other than roads, dirt, and sidewalks.

G. DQN Parameters

Determining the right hyperparameter value is one of the important steps taken to get a good machine learning model. In machine learning, hyperparameters are parameters used to regulate the course of the machine learning process. In contrast to model parameters whose values change as machine learning progresses, hyperparameters need to be defined up front and are generally constant throughout the learning process. In this study, the hyperparameters of the DQN algorithm are defined as follows:

Hyperparameter	Value	Description
MINIBATCH_SIZE	16	Jumlah sampel pembelajaran yang diproses oleh perhitungan SGD (stochastic gradient) algoritma DQN
PREDICTION_BATCH_SIZE	1	Jumlah sampel yang di prediksi di saat yang bersamaan
TRAINING_BATCH_SIZE	8	Jumlah sampel yang di fit di saat yang bersamaan (lebih besar lebih cepat)
TRAINER_MEMORY_FRACTION	0.6	

TABLE I: Hyperparameter model.

1) Model:

Hyperparameter	Value	Description
DISCOUNT	0.99	Nilai faktor discount dalam perhitungan reward algoritma DQN
REPLAY_MEMORY_SIZE	20_000	Berapa banyak step terakhir yang disimpan untuk training model
MIN_REPLAY_MEMORY_SIZE	5_000	Jumlah step minimum dalam memori untuk memulai training
OPTIMIZER_LEARNING_RATE	0.001	Laju pembelajaran yang digunakan pada optimizer
OPTIMIZER_DECAY	0.0	Pengurangan laju pembelajaran pada optimizer setiap episodenya

TABLE II: Hyperparameter DQN.

2) DQN:

Hyperparameter	Value	Description
START_EPSILON	1	Nilai epsilon saat pertamakali mulai training
EPSILON_DECAY	0.9995	Penurunan nilai epsilon di setiap episodenya
MIN_EPSILON	0.1	Epsilon terendah yang diperbolehkan

TABLE III: Hyperparameter epsilon.

3) Epsilon: The determination of the epsilon parameters is determined by the following conditions. Epsilon will be worth 1 when you first start learning or at episode 0. Every time a new episode starts, the epsilon value will be multiplied by 0.9995 until it will finally be 0.1 after 4700 episodes. The reduction in the epsilon value will stop after reaching the value of 0.1.

The determination of the epsilon-greedy parameter is carried out so that the agent is able to carry out exploration and exploitation activities appropriately. So that the results obtained will be good in a short time.

H. Arsitektur Model

Here is the architecture used:

Layer (type)	Output Shape	Param #
conv2d_1_input (InputLayer)	(None, 270, 480, 1)	0
conv2d_1 (Conv2D)	(None, 270, 480, 64)	640
activation_1 (Activation)	(None, 270, 480, 64)	0
average_pooling2d_1 (AveragePoo	(None, 90, 160, 64)	0
conv2d_2 (Conv2D)	(None, 90, 160, 64)	36928
activation_2 (Activation)	(None, 90, 160, 64)	0
average_pooling2d_2 (AveragePoo	(None, 30, 54, 64)	0
conv2d_3 (Conv2D)	(None, 30, 54, 64)	36928
activation_3 (Activation)	(None, 30, 54, 64)	0
average_pooling2d_3 (AveragePoo	(None, 10, 18, 64)	0
flatten_1 (Flatten)	(None, 11520)	0
kmh_input (InputLayer)	(None, 1)	0
concatenate_1 (Concatenate)	(None, 11521)	0
dense_1 (Dense)	(None, 256)	2949632
dense_2 (Dense)	(None, 3)	771

TABLE IV: Arsitektur model.

Convolution is given to the images 3 times. Each convolution is done by average pooling. The next process after convolution is adding agent speed data. Then at the end of the process there will be 3 output actions.

I. Training

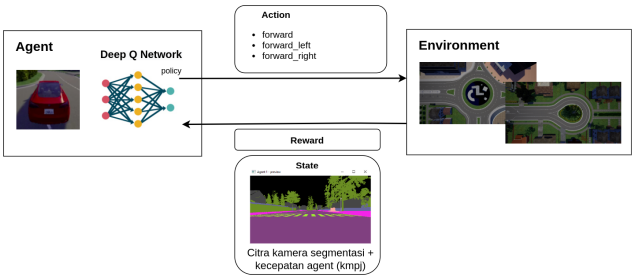

Fig. 14: Diagram Blok Metodologi

The training process is carried out after all configurations are completed. The training methodology of this research is shown in Figure 14. The agent in the simulation performs an action that causes the state to change. The state in the form of a segmentation image and agent speed (kmpj) as well as a reward from the agent is then sent to DQN for training. The result of the training in the form of a policy will determine the action which will then be carried out by the agent. Then the process will repeat itself into an endless cycle.

III. Testing and Result

A. Training DQN

DQN algorithm training is carried out using the following hardware:

CPU	Intel Core i5
GPU	Nvidia GTX 1060 - 6GB VRAM
RAM	8GB
Disk	256GB SSD M.2
OS	Windows 10

TABLE V: Hardware used for training.

Training is done on a local machine using a GPU. The machine learning software environment used in training is:

Software	**Version**
tensorflow-gpu	1.13.1
keras	2.2.5
h5py	3.1
python	3.7.9

TABLE VI: Software environment used for training.

The result of the training process is a DQN network model that represents an autonomous car movement planning system. The model is stored in a .model file format which can then be visualized with a tensorboard to see the results of the DQN algorithm training.

1) DQN training at Roundabout with Intersections using Grayscale Segmentation: The training of the DAN algorithm at the four-way roundabout with grayscale segmentation lasted for 20 hours. Here are the results of the DQN algorithm training visualization:

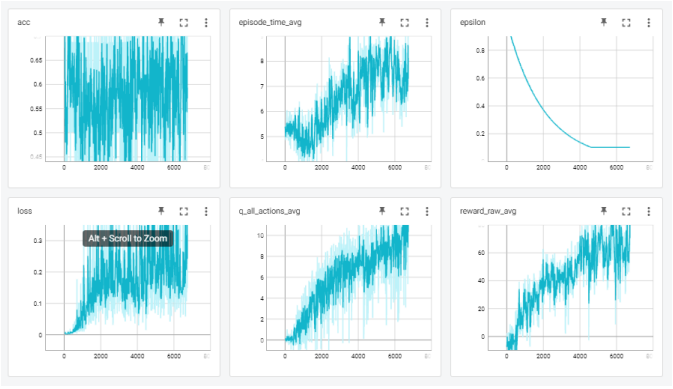

Fig. 15: Tensorboard result on Roundabout with Intersections using Grayscale Segmentation

From the results of the DQN training visualization in Figure 15, it can be seen that the DQN algorithm training process has been running well. This can be seen from the value of accuracy, episode time, and rewards for driving autonomous cars which have an increasing trend as the training process progresses.

2) DQN training at Roundabout with Intersections using Advanced Segmentation: The DQN algorithm training at the four-way roundabout with advanced segmentation lasted for 13 hours, 30 minutes, and 48 seconds. Here are the results of the DQN algorithm training visualization:

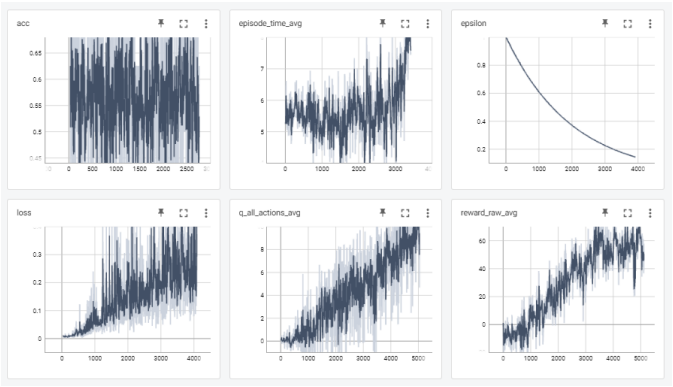

Fig. 16: Tensorboard result on Roundabout with Intersections using Advanced Segmentation

From the results of the DQN training visualization in Figure 17, it can be seen that the DQN algorithm training process has been running well. This can be seen from the value of accuracy, episode time, and rewards for driving autonomous cars which have an increasing trend as the training process progresses.

3) DQN training at Roundabout without Intersections using Advanced Segmentation: DQN algorithm training on roundabouts without intersections with advanced segmentation lasted 26 hours and 25 seconds. The following are the results of the DQN algorithm training visualization:

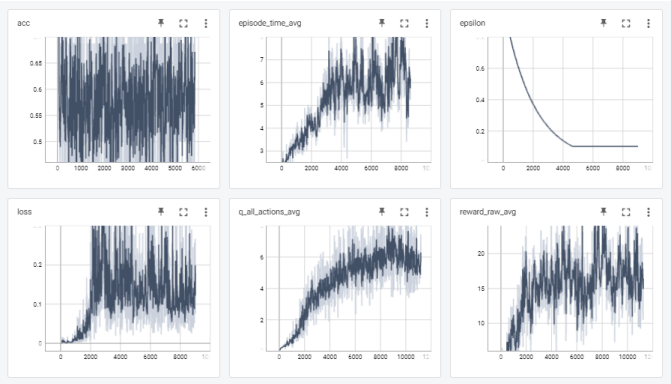

Fig. 17: Tensorboard result on Roundabout without Intersections using Advanced Segmentation

From the results of the DQN training visualization in Figure 17, it can be seen that the DQN algorithm training process has been running well. This can be seen from the value of accuracy, episode time, and rewards for driving autonomous cars which have an increasing trend as the training process progresses.

B. DQN Model Test

In this section, we will test the resulting model from the DQN training process. The test is carried out when the driving environment is under normal conditions.

episode_length	avg_speed	avg_angle_diff	avg_dist_diff	avg_reward
9.005	34.983	20.001	5.812	0.166
4.772	28.927	12.14	0.91	0.448
8.524	33.052	10.65	1.993	0.398
12.678	35.053	15.259	3.264	0.222
13.621	37.03	32.642	12.394	0.122
4.851	30.102	10.241	1.253	0.471
6.555	20.433	21.37	2.775	0.318
8.694	31.494	14.927	3.381	0.235
20.184	34.142	58.799	45.228	-0.23
10.026	28.305	12.737	2.411	0.332
5.147	29.648	17.682	1.707	0.289
8.521	32.238	13.255	2.253	0.289
7.99	32.617	13.939	2.661	0.342
5.206	30.142	13.541	1.077	0.409
6.072	23.77	36.641	3.617	0.17
5.426	31.166	12.926	0.925	0.418
5.176	27.477	12.276	1.194	0.457
8.085	33.884	11.883	1.224	0.343
10.048	27.918	15.146	3.086	0.306
7.443	33.063	12.29	2.298	0.383
29.772	38.947	65.564	58.657	-0.255
6.617	30.019	20.405	1.658	0.425
4.931	27.362	12.078	1.821	0.361
14.428	29.973	18.9	3.066	0.272
5.445	30.364	10.48	0.951	0.523
8.794	31.964	15.54	2.635	0.294
8.568	33.002	13.103	1.95	0.409
20.471	24.614	64.263	15.817	-0.076
5.276	31.54	14.182	1.594	0.349
7.645	29.516	39.559	1.573	0.349

TABLE VII: 30 result sample of DQN Model Test at Roundabout with Intersections using Grayscale Segmentation

1) DQN Model Test at Roundabout with Intersections using Grayscale Segmentation: From the results of the sampling, conclusions are obtained as in table VIII.

avg_episode_length	avg_speed	avg_angle_diff	avg_dist_diff	avg_reward
9.332	30.758	21.413	6.306	0.284

TABLE VIII: Summary of DQN Model Test at Roundabout with Intersections using Grayscale Segmentation

episode_length	avg_speed	avg_angle_diff	avg_dist_diff	avg_reward
7.431	38.229	10.205	2.0	0.34
8.094	30.83	24.743	1.187	0.392
8.418	30.337	20.039	2.106	0.257
21.587	37.044	14.712	2.588	0.28
7.019	35.495	11.015	0.849	0.465
19.252	35.029	20.848	3.259	0.267
18.607	29.382	58.62	6.881	0.071
6.529	36.715	10.4	1.301	0.348
14.337	36.066	16.105	0.99	0.421
17.025	34.249	45.883	4.252	0.15
9.511	41.363	12.647	1.503	0.32
36.864	34.917	68.418	8.128	0.069
9.973	38.902	13.901	1.56	0.304
7.34	35.594	13.333	1.547	0.45
14.864	31.482	59.456	3.588	0.112
9.987	41.68	12.595	1.952	0.297
9.767	39.018	15.001	1.341	0.373
7.011	36.749	10.62	1.488	0.321
17.4	26.631	79.254	13.336	-0.07
12.798	40.187	15.749	1.645	0.285
17.809	31.448	46.125	2.951	0.268
11.504	36.149	21.827	1.347	0.355
6.395	37.127	9.601	0.895	0.418
23.591	30.945	60.991	4.07	0.153
6.957	37.284	8.505	0.959	0.438
9.431	40.093	11.436	1.98	0.311
35.604	27.912	80.507	11.311	-0.171
9.089	42.26	10.891	1.286	0.395
7.032	36.387	11.418	1.278	0.418
8.578	34.796	15.489	1.456	0.316

TABLE IX: 30 result sample of DQN Model Test at Roundabout with Intersections using Advanced Segmentation

2) DQN Model Test at Roundabout with Intersections using Advanced Segmentation: From the results of the sampling, conclusions are obtained as in table X.

avg_episode_length	avg_speed	avg_angle_diff	avg_dist_diff	avg_reward
13.326	35.476	27.011	2.967	0.278

TABLE X: Summary of DQN Model Test at Roundabout with Intersections using Advanced Segmentation

episode_length	avg_speed	avg_alpha	avg_reward
8.374	32.202	31.098	0.113
8.796	29.909	32.023	0.093
12.195	29.678	40.528	0.068
12.439	30.384	36.557	0.12
10.246	29.72	42.027	0.063
3.138	19.86	32.557	0.085
9.982	30.777	35.319	0.129
8.185	30.64	24.322	0.112
9.719	31.814	31.563	0.081
10.908	33.314	38.641	0.097
3.299	21.882	22.22	0.122
9.27	31.347	26.635	0.12
4.27	26.317	16.402	0.152
3.765	23.032	32.14	0.169
10.776	31.581	37.949	0.091
5.588	28.784	29.461	0.155
3.219	22.536	20.392	0.186
8.508	30.489	24.669	0.122
4.457	23.68	23.546	0.146
3.824	22.575	36.989	0.105
10.599	31.869	23.992	0.132
12.438	31.08	38.071	0.088
7.873	31.305	21.448	0.15
3.562	22.864	16.225	0.136
8.758	27.877	37.786	0.076
4.641	26.732	25.345	0.106
10.472	32.074	33.322	0.101
10.937	31.784	33.864	0.095
9.244	30.508	28.196	0.12

TABLE XI: 30 result sample of DQN Model Test at Roundabout without Intersections using Advanced Segmentation

3) DQN Model Test at Roundabout without Intersections using Advanced Segmentation: From the results of the sampling, conclusions are obtained as in table XII.

avg_episode_length	avg_speed	avg_alpha	avg_reward
7.958	28.533	30.068	0.115

TABLE XII: Summary of DQN Model Test at Roundabout without Intersections using Advanced Segmentation

IV. Conclusion

A. Conclusion

From the system design and testing that has been done, some conclusions are obtained as follows. The constraints and shortcomings that the authors face are also written in the suggestions section in the hope of assisting the development of further research.

1) The autonomous car movement planning system based on the DQN algorithm which is designed to be able to accelerate and steer, is able to control the autonomous car and understand the surrounding conditions.
2) Use of image state with advanced segmentation; ie producing drivable and non-drivable image outputs produces better results than ordinary segmentation with 42.8% better average episode length performance, 15.3% better average speed, and 112.4% better average distance difference.

B. Suggestion

For further development on the topic of autonomous car movement planning research using the DQN algorithm, there are several suggestions given, including the following:

1) The sensors used for navigation from agents using the DQN algorithm can be added to more such as GPS and LIDAR, so that agents can act with more complete parameters.
2) The training process can be carried out with a longer time and using machines with stronger computing power in order to obtain a convergent model.

References

[1] M. V. Rajasekhar and A. K. Jaswal, "Autonomous vehicles: The future of automobiles," pp. 1–6, 2015.

[2] menristekbrin, "Menristek/kepala brin luncurkan icar its," *Biro Kerja Sama dan Komunikasi Publik Kemenristek/BRIN*, Aug 2020. [Online]. Available: https://www.ristekbrin.go.id/menristek-kepala-brin-luncurkan-icar-its/

[3] itsnews, "I-car, its special gift for indonesia," *ITS News*, Aug 2020. [Online]. Available: https://www.its.ac.id/news/en/2020/08/18/i-car-its-special-gift-for-indonesia/