

# **A Study on Active Suspension System with Reinforcement Learning**

**by Daoyu Shen**

Thesis submitted in fulfilment of the requirements for  
the degree of

**Master by Research**

under the supervision of Dongbin Wei, Thorsten Lammers

University of Technology Sydney  
Faculty of Engineering and Information Technology

June 2022

## Certificate of Original Authorship

### CERTIFICATE OF ORIGINAL AUTHORSHIP

I, Daoyu Shen declare that this thesis, is submitted in fulfilment of the requirements for the award of Master by Research Degree, in the School of Mechanical & Mechatronic Engineering/ Faculty of Engineering and IT at the University of Technology Sydney.

This thesis is wholly my own work unless otherwise referenced or acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

I certify that the work in this thesis has not previously been submitted for a degree nor has it been submitted as part of the requirements for a degree at any other academic institution.

This thesis does not include Indigenous Cultural and Intellectual Property (ICIP).

This research is supported by the Australian Government Research Training Program.

Signature:

Production Note:

Signature removed prior to publication.

Date: 01.06.2022

## **Acknowledgements**

I would like to take this opportunity to thank the following people for their assistance and support during my study as a master candidate.

First and the most, I would like to thank my supervisor Prof. Dongbin Wei who has provided me with a great support from the very tough time that I have been through. His valuable knowledge and excellent research attitude have guided me into the way of academia. I would like to provide my thankfulness to my previous supervisor Prof. Nong Zhang and Dr. Paul Walker. It was a brilliant time to work with them in my research period. Their attitude of research towards academic working has impressed me a lot.

I would also like to take this opportunity to thank my great colleague Shilei Zhou, Boyi Xiao, Cong Thanh Nguyen, Yeman Fan, Shun Chen. They helped me resolve a lot of problems in my research and encouraged me to keep on moving in the way I am walking on.

At last, I would like to thank all my family members, my wife Yi, my little boy Yiquan, my parent Jialiang and Jingwen, my parent in law Xiaotang and Xiaoji. they are so supportive on my study and life and are always very patient when I was in a mood. I couldn't have done this without you.

## List of Figures

Figure 2-1 Passive Suspension System.....	6
Figure 2-2 Semi Active Suspension System.....	7
Figure 2-3 Active Suspension System .....	7
Figure 2-4 Structure of RBF Neural Network .....	8
Figure 2-5 PID with RBF neural network Control .....	9
Figure 2-6 1/4 Car model with semi-active suspension.....	11
Figure 2-7 DDPG Control Structure .....	13
Figure 2-8 Reinforcement Learning Diagram.....	14
Figure 2-9 Actor-Critic Algorithm.....	15
Figure 2-10 Control Loop .....	15
Figure 3-1 Quarter Car Model .....	20
Figure 3-2 Car Body Force Analysis .....	21
Figure 3-3 Suspension Force Analysis .....	22
Figure 3-4 Mass Blocks and Force Blocks .....	26
Figure 3-5 Spring Forces and Damper Forces .....	26
Figure 3-6 Quarter Car Model .....	27
Figure 3-7 Body Acceleration with Step Input.....	27
Figure 3-8 Suspension Displacement with Step Input.....	28
Figure 3-9 Body Acceleration with Sinusoidal Input .....	28
Figure 3-10 Suspension Displacement with Sinusoidal Input .....	29
Figure 4-1 Reinforcement Learning Concept Diagram .....	31
Figure 4-2 Rewards Signal.....	34
Figure 4-3 Basic Neural Network Structure .....	35
Figure 4-4 Neural Network Node Value Transfer .....	36
Figure 4-5 Critic Network Structure .....	38
Figure 4-6 Reinforcement Learning Frame Structure.....	39
Figure 4-7 Road Excitation in Sinusoidal Signal Form .....	40
Figure 4-8 Acceleration of Car Body in Sinusoidal Input .....	41
Figure 4-9 Displacement of Suspension in Sinusoidal Input.....	41
Figure 4-10 Training Result with Sinusoidal Input .....	42
Figure 4-11 Acceleration of Car Body after Control.....	42
Figure 4-12 Displacement of Suspension after Control.....	43
Figure 5-1 Simulink Block of Road Roughness .....	49
Figure 5-2 Road Roughness Excitation with vehicle speed of 30 km/h on Grade C Road .....	49
Figure 5-3 Power Spectral Density with Different Overlap Rate .....	50
Figure 5-4 PSD of Road Roughness with Different Noise Power.....	50
Figure 5-5 Classification of Roads according ISO8648 .....	51
Figure 6-1 Learning Curve Single Layer Network.....	56
Figure 6-2 Learning Curve Comparison with Different Nodes Episode 0-200.....	57
Figure 6-3 Acceleration Comparison for Single Layer Network.....	57
Figure 6-4 RMSE of Acceleration VS Elapse Time Single Layer Network .....	58

Figure 6-5 Learning Curve Double Layer Network .....	60
Figure 6-6 Acceleration Comparison for Double Layer Network.....	61
Figure 6-7 RMSE of Acceleration VS Elapse Time Double Layer Network.....	61
Figure 6-8 Learning Curve Triple Layer Network .....	63
Figure 6-9 Acceleration Comparison for Triple Layer Network.....	64
Figure 6-10 RMSE of Acceleration VS Elapse Time Triple Layer Network.....	64
Figure 6-11 RMSE of Acceleration VS Elapse Time with All Three Network Structure.....	65
Figure 6-12 Learning Curve with Different Learning Rate .....	67
Figure 6-13 Acceleration Comparison with Different Learning Rate .....	67
Figure 6-14 RMSE of Acceleration VS Elapse Time with Different Learning Rate .....	68
Figure 6-15 Learning Curve with Different Greedy Rate.....	70
Figure 6-16 Acceleration Comparison with Different Greedy Rate.....	70
Figure 6-17 Epsilon and Exploration-Exploitation.....	71
Figure 6-18 RMSE of Acceleration VS Elapse Time with Different Greedy Rate .....	71
Figure 6-19 Learning Curve with Different Sample Rate .....	72
Figure 6-20 Acceleration Comparison with Different Sample Rate.....	73
Figure 6-21 RMSE of Acceleration VS Elapse Time with Different Sample Rate.....	73
Figure 6-22 Learning Curve with Different Action Space Size .....	75
Figure 6-23 Acceleration Comparison with Different Action Space.....	75
Figure 6-24 RMSE of Acceleration VS Elapse Time with Different Action Space Size.....	76
Figure 6-25 DDPG Network Structure .....	78
Figure 6-26 Learning Curve DQN Vs DDPG.....	80
Figure 6-27 Acceleration Comparison DQN Vs DDPG.....	81
Figure 6-28 Acceleration Comparison DQN Vs DDPG Clipped .....	81
Figure 6-29 Displacement Comparison DQN Vs DDPG .....	82
Figure 6-30 TD3 Network Structure.....	84
Figure 6-31 B Class Road Learning Curve DQN Vs DDPG Vs TD3 .....	87
Figure 6-32 B Class Road Body Acceleration Comparison Clipped.....	88
Figure 6-33 B Class Road Unsprung Acceleration Comparison Clipped.....	88
Figure 6-34 B Class Road Suspension Displacement Comparison Clipped.....	89
Figure 6-35 B Class Road Controller Force Comparison Clipped .....	90
Figure 6-36 B Class Road Tyre Dynamic Load Comparison Clipped .....	90
Figure 6-37 D Class Road Learning Curve DQN Vs DDPG Vs TD3.....	91
Figure 6-38 D Class Road Acceleration Comparison Clipped .....	92
Figure 6-39 D Class Road Unsprung Acceleration Comparison Clipped .....	92
Figure 6-40 D Class Road Displacement Comparison Clipped.....	93
Figure 6-41 D Class Road Controller Force Comparison Clipped.....	93
Figure 6-42 D Class Road Tyre Dynamic Load Comparison Clipped .....	94

## List of Tables

Table 5-1 Road Roughness Value Classified by ISO [33] .....	45
Table 5-2 Road Roughness Value with Angular spatial frequency unit [33] .....	46
Table 6-1 List of Impact Factors.....	53
Table 6-2 Neural Network Structure Category 1 .....	54
Table 6-3 Neural Network Structure Category 2.....	59
Table 6-4 Neural Network Structure Category 3 .....	62

## **Abbreviations**

<b>EV</b>	<b>Electric vehicle</b>
<b>PSS</b>	<b>Passive suspension system</b>
<b>ASS</b>	<b>Active suspension system</b>
<b>RL</b>	<b>Reinforcement learning</b>
<b>PSD</b>	<b>Power spectral density</b>
<b>NN</b>	<b>Neural Network</b>
<b>DOF</b>	<b>Degrees of Freedom</b>
<b>RBF</b>	<b>Radial basis function</b>
<b>PID Controller</b>	<b>Proportional–integral–derivative Controller</b>
<b>DQN</b>	<b>Deep Q Net</b>
<b>DDPG</b>	<b>Deep Deterministic Policy Gradient</b>
<b>FCN</b>	<b>Fully connected network</b>

## **Abstract**

The research conducted in this paper focuses on providing a superior solution for controlling the active suspension system through machine learning algorithm. An active suspension system is embedded with an extra actuator and can provide more flexibility in dealing with different road situation than a passive suspension system that only contains a fixed set of spring and damper.

Among various control methods, the machine learning control strategy has demonstrated its optimality in dealing with different classes of roads depending on its self-learning capability. In this study, an advanced method of creating pavement signal has been presented to guarantee the quality of the simulation, together with Twin Delayed Deep Deterministic Policy Gradients (TD3) in suspension control that is an application of one of the cutting-edge algorithms in Reinforcement Learning (RL). To be able to realize the research, policy gradient algorithm, Markov decision process, and neural network modelling have been adopted.

To achieve such a proposed frame structure, a vehicle suspension model has been established together with a frame of reinforcement learning algorithm and an input signal of road pavement. The performance of the proposed twin delayed reinforcement agent has been compared against Deep Deterministic Policy Gradients (DDPG) and Deep Q-Learning (DQN) algorithms under different types of pavement input. The simulation result shows its superiority, robustness, and learning efficiency over other reinforcement learning algorithms.



# Contents

Abstract.....	VII
Chapter 1. Introduction.....	1
1.1 Background .....	1
1.2 Research significance.....	2
1.3 Thesis Outline .....	3
Chapter 2. Literature review.....	5
2.1 Type of suspension system.....	6
2.2 RBF (Radial basis function network) neural network control .....	8
2.3 Active suspension control based on deep reinforcement learning .....	9
2.4 Vibration control using DDPG.....	13
2.5 Stochastic road excitation.....	16
2.6 Summary and conclusion .....	18
Chapter 3. Suspension system modeling.....	19
3.1 Theoretical modelling and parameters definition.....	19
3.2 Systematic force analysis .....	20
3.3 State space form of quarter car model.....	23
3.4 Simulation of the suspension model.....	25
Chapter 4. Reinforcement learning frame definition.....	30
4.1 Introduction of reinforcement learning .....	30
4.2 Establishment of environment, observation space and action space.....	32
4.3 Reward function definition and simulation.....	33
4.4 Policy and neural network definition .....	34
4.5 Reinforcement learning frame assembly.....	38
4.6 System test with sinusoidal signal.....	40
Chapter 5. Road excitation signal generation.....	44
5.1 Modeling .....	44
5.2 Simulation and analysis.....	48
Chapter 6. System Assembly and Simulation .....	52
6.1 DQN algorithm application.....	52
6.1.1 Impact analysis of different neural network structure .....	54
6.1.1.1 Single layer neural network structure.....	54
6.1.1.2 Double layer neural network structure .....	59

6.1.1.3	Triple layer neural network structure .....	62
6.1.2	Impact analysis of hyper-parameters of reinforcement learning .....	66
6.1.2.1	Learning Rate Analysis .....	66
6.1.2.2	Greedy Rate Analysis .....	68
6.1.2.3	Sample Rate Analysis.....	72
6.1.2.4	Action Space Size Analysis.....	74
6.2	DDPG Algorithm Application .....	76
6.2.1	Introduction.....	76
6.2.2	DDPG Algorithm .....	79
6.2.3	Simulation and comparison.....	79
6.3	TD3 Algorithm Application .....	82
6.3.1	Introduction.....	82
6.3.1.1	Twin Critic Networks .....	83
6.3.1.2	Delayed Updates.....	83
6.3.1.3	Noise Regularization .....	84
6.3.2	TD3 Algorithm.....	85
6.3.3	Simulation and Comparison.....	86
6.3.3.1	System Simulation Setup.....	86
6.3.3.2	Class B Pavement Signal Test.....	86
6.3.3.3	Class D Pavement Signal Test.....	91
Chapter 7.	Conclusions and future work .....	95
7.1	Summary .....	95
7.2	Prospective of future work .....	96
Reference	.....	98

# Chapter 1. Introduction

## 1.1 Background

Suspension system development has a long history that can trace back to the time when horse-drawn vehicles were dominating the streets. Those carriages can only work in a low speed because their suspension systems have not been designed to be functioning in a high driving speed.

In 1901 shock absorbers had been first used in vehicles by Mors in Paris. The brand-new design of suspension system with damping which is so called “Mors Machine” helped Henri Fournier defeat others competitors in Paris-to-Berlin race on 20 June 1901. In 1906 Brush Motor Company presented a suspension design in its production vehicles which utilized a coil spring. These days, this design has widely applied in almost all vehicles from different manufactures. In 1922, the idea of independent front suspension system was delivered by Lancia Lambda and soon it was spreading out in the industry and then widely accepted. In modern ages of car industry, independent suspension has been applied on four wheels in almost every vehicle and advanced control technologies have been developed and involved [1].

To the recent days, suspension systems start to pose a significant influence on design standard from the industry, which closely related to driving stability, safety and riding experience [2]. According to the latest method of category, suspension system can be put into three divisions [3], which are passive suspension, semi-active suspension and active suspension. Since passive suspension is not adjustable which means the parameters of system are fixed, it can hardly fit to different types of road conditions, while semi-active and active suspension system have changeable dynamic properties, therefore have a better performance [4] and become a new hot spot of research in both academia and industry.

According to the design purpose of the suspension system, when the car body is having some road oscillation, vehicle is not supposed to experience large disturbance, and the vibration

should be reduced in a short period of time. To enhance the overall performance of the vehicle, road holding ability is important towards an acceptable design of an automotive suspension system, while riding comforts are still provided when running over bumps and holes in the street. In this case, a real-time learning ability of the system is vital in vehicle suspension control design work when suspension parameters can be changeable according road situation. By echoing to this demand, new method has been developed to accelerate the research of suspension system which contains a damping adjustable damper or an additional facility which is called actuator which can provide an extra force to the suspension system. Machine learning, as a cutting edge controlling method has emerged in public horizon and been applied in a diversity of industrialized designing work [5] [6] [7] [8] [9] [10] [11] [12] and manufacturing process [13] [14] [15]. This study investigated the performance of an active suspension system which is controlled by a reinforcement learning algorithm in multiple road situation. The key deliverables of this study are listed below

## **1.2 Research significance**

This research focused on applying one of the most powerful and cutting-edge algorithms in Reinforcement Learning (RL), Twin Delayed Deep Deterministic Policy Gradients (TD3) in suspension control. To the best of our knowledge, this is the first time a TD3 algorithm is used in active suspension control with reinforcement learning. Among numbers of divergent controlling methods, a machine learning control, especially, a reinforcement learning control has demonstrated its unique performance which illustrated a big potential. Reinforcement learning uses dynamic environmental data to output the best sequence of actions from action space that produces the optimal result. To achieve that target, a component which is called as agent is used in reinforcement learning to explore the environment, communicate with the environment, and learn from the environment without knowing what the environment is. This

approach is perfectly suitable in controlling the force of an actuator when interacting with a dynamic process of active suspension system controlled by a stochastic road signal.

Considering the current reinforcement learning algorithms, DQN can only handle a discrete action space while the force from the actuator is actually continuous which limits the control performance. DDPG is able to cope with a continuous action space, however is sometimes trapped in a second optimal solution due to the complexity of the system with nonlinear and stochastic attributes. A TD3 strategy tackles this problem by reducing the overestimation bias from DDPG to get a better performance. Therefore, based on the result of literature review that have been shown above, in this paper, a suspension control strategy with TD3 deep reinforcement learning algorithm is proposed to fill the blank of this area.

### **1.3 Thesis Outline**

The thesis consists of 7 chapters, shown as follows:

Chapter 1: The background information, research significance and the overview of this paper are introduced.

Chapter 2: A literature review of the modern suspension system, traditional method of suspension control, reinforcement learning application and road pavement signal generation are covered in this chapter.

Chapter 3: The modelling of suspension system is introduced in this chapter. This includes the establishment of the dynamic model, parameters definition and simulation analysis of the suspension model.

Chapter 4: In this chapter, a reinforcement learning frame is created. The chapter introduces the background knowledge of machine learning and reinforcement learning, it also covers key components definition in RL, which consists of environment, observation and action space, reward signal and training policy application. A simulation test forms the last part of this chapter.

Chapter 5: This chapter consists of how a reliable stochastic pavement signal can be generated. The chapter focuses on the current method of creating a road signal and provides an optimized method in generating an accurate road signal according to the industry standard.

Chapter 6: This chapter provides the system level validation and simulation. A detailed analysis and study are conducted into evaluating the impact of hyper-parameters of reinforcement learning towards the learning progress and control performance in the system. Three different algorithms are applied into the control system, they are DQN, DDPG and TD3. The performance of all algorithms is illustrated and presented. According to the original plan, hardware validation was schemed, and the result was expected to be illustrated in this chapter. However due to the COVID restriction applied in the time frame of this study, the lab was not accessible, hence, the hardware validation was finally canceled.

Chapter 7: Conclusions of this work are given and further recommendations are summarized.

## Chapter 2. Literature review

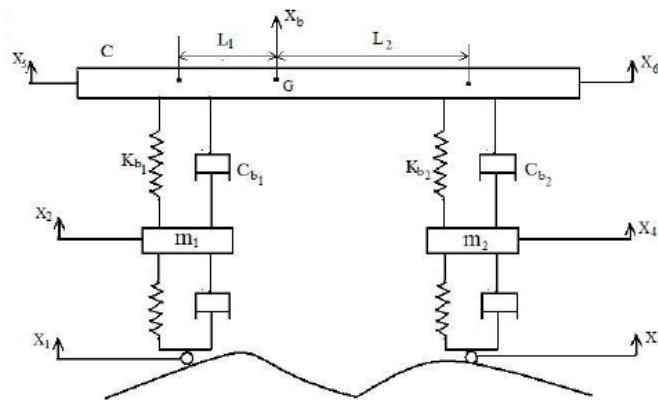
This chapter intends to introduce about background information and classification of suspension system. Also, the paper detail some classic and modern methods in current practices of suspension system control. Turkey investigated the design of the active suspension model based on the linear-quadratic-Gaussian (LQG) methodology[16]. Bhanu has defined the category of suspension systems according to their function[3]. Turkey and Akcay are trying to investigate the performance of a half car suspension model through integrating a method of a mixed  $H_2$ - $H_\infty$  algorithm[17]. Qiang Zhao and Jiaying Yin have worked out an RBF neural network method and applied it into the active suspension system control[18]. Yagiz and Hacioglu created a non-linear full vehicle model of seven degree of freedom, by using backstepping method they investigated the system both in time and frequency domains[19]. Seong-Jae Kim, Hyun-Soo Kim and Dong-Joong Kang have put some effort into the investigation of DDPG algorithm and its application in suspension control with the road simulation in the form of sinusoidal and step signal[20]. Witters and Swevers studied the possibility of identifying a variable changeable damper by applying a multilayer perception neural network and then defined the damper dynamics through their work[21]. Liu Ming and Li Yibin also implement a study in suspension system control with DDPG algorithm[22], they improve the road simulation method through a practice of using a random white noise signal. Fateh and Alavi developed a new method in controlling an active suspension system through the input loops from two internal active forces and a displacement[23]. Through these researches, a clear view of current technology of suspension system development can be presented and a possible way of a future direction of research is demonstrated.

## 2.1 Type of suspension system

According to Bhanu's paper[3], suspension plays an important role in car industry. It is working for reducing and absorbing the vertical vibration, abating and restraining the shock and vibration. It also poses a significant influence on the driving comfort, safety requirement and road handling of performance. Recently, the research and development of active suspension has become a hotspot for research.

The three main types of suspension system are defined as passive suspension system, semi-active suspension system and active suspension system. Each type of suspension system has its pros and cons. However, semi-active suspension system is most used across the industry.

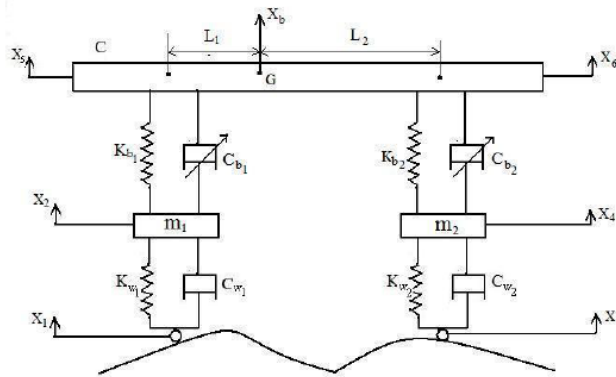
A passive suspension system is demonstrated in Figure 2-1. This system contains energy storing units which are springs, and energy dissipating parts which are dampers. Since springs and dampers are not able to create a controllable active force which can add energy to the suspension system, this type of suspension is called passive suspension system.



**Figure 2-1 Passive Suspension System**

According to Figure 2-2, a structure of semi active suspension system is presented. To realize a balance between functional advantage and cost of development, a semi active suspension system has emerged.

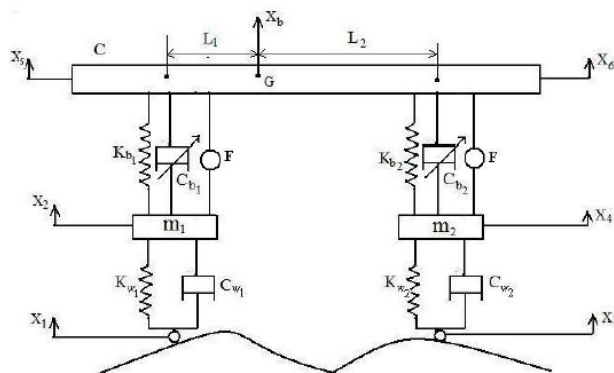




**Figure 2-2 Semi Active Suspension System**

In this design, on one hand, a passive suspension spring is kept, on the other hand, the damping coefficient can be modified in according to the road conditions. Based on the design of the damper, orifice area can be changed to impact the fluid flow so that the damping force can be adjusted accordingly.

Regarding to an active suspension system which is showed in Figure 2-3, a force actuator is placed in parallel to a passive suspension system. Parameters such as accelerations of sprung mass and unsprung mass, can be monitored by the sensors installed. The measurement results are delivered as analog signals which will be sent to a controller, then the controller shall provide appropriate feedbacks to the force actuator.

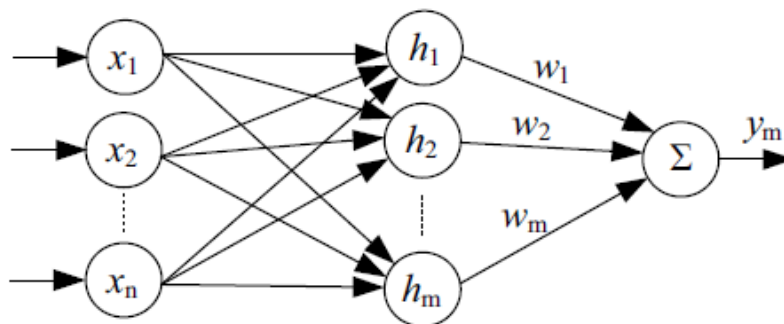


**Figure 2-3 Active Suspension System**

Because active suspension system requires additional energy and structure to power, so it is apparently, a tradeoff by considering the simplicity and complexity, high low cost of the whole system.

## 2.2 RBF (Radial basis function network) neural network control

Before machine learning was widely researched, neural network was a research focus due to its special attribute of working as a general function estimator. Although not a direct application of machine learning control, it has contributed to the development of machine learning. Therefore, it is worthwhile to look into the applications related to suspension control. Qiang Zhao and Jiaxing Yin[18], has developed a method to control the suspension system by applying RBF neural network control. The RBF neural network principle is first brought up by J. Moody and C. Darken, which is a neural network with the function of domestic estimation[24]. The benefit of RBF neural network is obvious, it provides a quick response regarding to the converge of the results and the accuracy of the output is also very high, therefore, its application has crossed a wide range of, such as pattern recognition, function fitting and real-time control. A diagram of the structure of the RBF neural network is shown in Figure 30, the parameters training is based on an algorithm of gradient descent.



*Figure 2-4 Structure of RBF Neural Network*

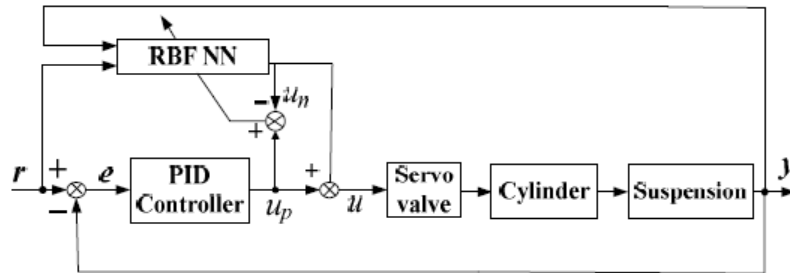
The input vector of RBF Neural network is denoted as  $X = [x_1, x_2, x_3, \dots, x_n]$ , the radial basis vector is denoted as  $H = [h_1, h_2, h_3, \dots, h_m]$ , where  $h_j$  is the Gaussian basis function, which can be described as

$$h_j = \exp\left(-\frac{X - C_j}{2b_j^2}\right), j = 1, 2, \dots, m \quad (1)$$

where  $C_j$  is the central vector,  $B_j$  is the basis width parameter. The weight vector is the network that is denoted as  $W = [w_1, w_2, w_3, \dots, w_n]$ , therefore the output is

$$y_m(k) = W^T H = w_1 h_1 + w_2 h_2 + \dots + w_m h_m \quad (2)$$

Then author uses an integrated control method of PID and RBF neural network to rule the servo system. As shown in Figure 2-5, RBF neural network has been fed with signal from a PID controller as the signal that functions as a supervision. After couple of cycles of learning, RBF neural network can follow the PID controller with high precision.



*Figure 2-5 PID with RBF neural network Control*

### 2.3 Active suspension control based on deep reinforcement learning

Liu Ming and Li Yibin[22], claim in their paper that active or semi-active suspension system is better than passive suspension regarding to the performance of vehicle body vibration and ride comfort. They also claimed that a strong real-time learning ability from the system is a key

towards an optimal control result in matching the road excitation and suspension parameters. According to their opinion, the traditional neural network controller is mostly working on offline learning scenario by dealing a big amount of static samples. Therefore, they proposed deep reinforcement learning strategy as an alternative to tackle the problem. An improved DDPG (Deep Deterministic Policy Gradient) algorithm has been applied into the system. To eliminate the disadvantage on learning efficiency from DDPG algorithm, empirical samples are involved. According to the simulation, it is easy to conclude that a DDPG algorithm integrated suspension system has a better performance on different class of roads comparing with the performance of a passive suspension system.

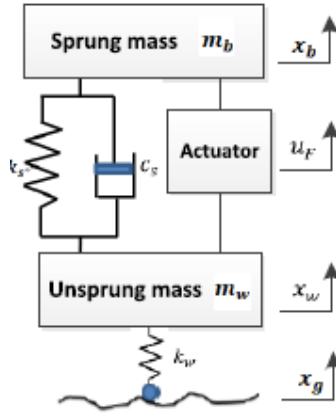
Deep Reinforcement Learning has become the main focus in regarding to the family of machine learning. It is widely admitted that deep learning is good at perception and reinforcement learning has a capability of decision making. And a DRL is actually a combination of advantages from deep learning and reinforcement learning. the criteria of DRL allows it to be adapted easily into different circumstance in control area. DRL has a unique advantage that it can be learning while accepting samples. This function optimizes the parameters from the samples with which it has been fed up and then decide the actions that will be taken in the next step. So, it can be fit to complicated situation that the system never come across before.

During the learning period, the performance declines. To resolve this issue, passive suspension system is activated in the learning gap of the DDPG control. Some positive training experiences have been shared to the DDPG algorithm so that the training efficiency can be enhanced.

In this paper, author uses a 2-DOF quarter car dynamic model to apply into the study. Below Figure 2-6 gives the idea of the model.

Where,  $m_b$  is mass of the vehicle body,  $k_s$  is spring stiffness,  $C_s$  is damping coefficient,  $m_w$  is unsprung mass,  $k_w$  is stiffness of tire,  $u_F$  is active control force generated by linear motor

actuator,  $x_g$  is road excitation,  $x_b$  and  $x_w$  are absolute displacements of body and tire respectively.  $x_b - x_w$  is suspension deflection.  $\ddot{x}_b$  is body acceleration.



**Figure 2-6 1/4 Car model with semi-active suspension**

By using Newton's second law, and the state space theory, the model can be described in below form:

$$\begin{cases} \dot{X} = AX + BU \\ Y = CX + DU \end{cases} \quad (3)$$

Where,

$$X = [x_w \quad x_b \quad \dot{x}_w \quad \dot{x}_b]^T$$

$$U = [x_g \quad u_F]^T$$

$$Y = [\ddot{x}_b \quad x_b - x_w]^T$$

$$A = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -\frac{k_s + k_w}{m_w} & \frac{k_s}{m_w} & -\frac{c_s}{m_w} & \frac{c_s}{m_w} \\ \frac{k_s}{m_b} & -\frac{k_s}{m_b} & \frac{c_s}{m_b} & -\frac{c_s}{m_w} \end{bmatrix}$$

$$B = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ -\frac{k_w}{m_w} & -\frac{1}{m_w} \\ 0 & \frac{1}{m_b} \end{bmatrix}$$

$$C = \begin{bmatrix} k_s & -k_s & c_s & -c_s \\ -1 & 1 & 0 & 0 \end{bmatrix}$$

$$D = \begin{bmatrix} 0 & \frac{1}{m_b} \\ 0 & 0 \end{bmatrix}$$

In their paper, they also established a road roughness model. They used PSD which is power spectral density to describe the random roughness of road surface. If looking into industrial standard, power spectrum of road roughness is given by an equation as follows:

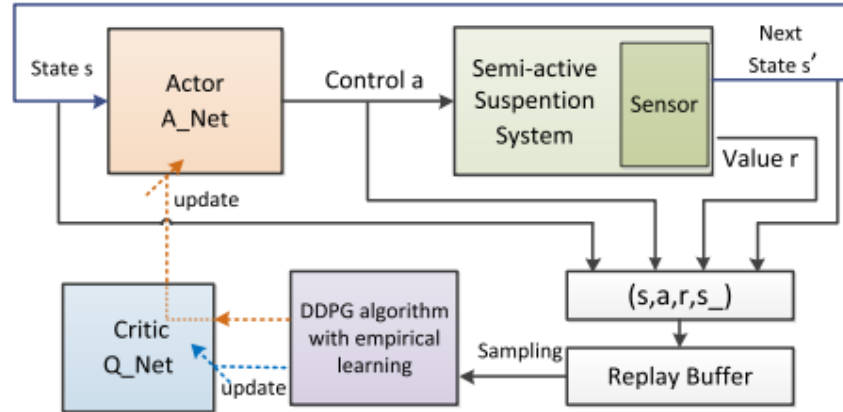
$$G_q(n) = G_q(n_0) \left(\frac{n}{n_0}\right)^{-\omega} \quad (4)$$

Then the road roughness model expression is:

$$\dot{g}(t) = -2\pi n_{00} \nu g(t) + 2\pi n_0 \sqrt{G_q(n_0)} \nu \omega(t) \quad (5)$$

This model is well in line with the standard PSD from industry. The structure of suspension control system based on DDPG is shown in Figure 2-7

There are two networks in the whole system: actor network and critical network. Actor network provides an action to the suspension system based on the information received. The system then applies DDPG algorithm on critic network to judge to the actions from the actor network and then, adjust the parameters of the actor network. By taking observation of previous and current state into consideration, the network is updated, the judging network is called a critical network.



**Figure 2-7 DDPG Control Structure**

According to this structure, the reward function has been decided as below equation

$$r = -(k_1 y_1^2 + k_2 y_2^2 + k_3 y_3^2) \quad (6)$$

In order to improve the performance of the model, a passive suspension have been used as learning samples. By doing this, a good reference for learning has been provided therefore improves the learning speed.

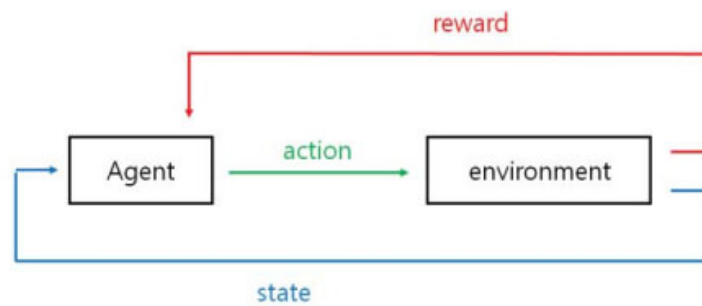
After simulation, the author claims that a DDPG controlled active suspension system can be well applied into all kinds of road conditions. Also, the author mentions that the performance of the method that is mentioned above is considerably outstanding comparing to other strategies.

## 2.4 Vibration control using DDPG

Seong-Jae Kim, Hyun-Soo Kim and Dong-Joong Kang[20], also developed a method of using DDPG strategy to control vehicle suspension system. In their thesis, they propose DDPG algorithm because the other learning method can only output discrete actions, whereas DDPG algorithm is about to deliver continuous actions.

Reinforcement learning is a machine learning method. It is a method in which the environment and agents interact with each other and learn the reward value for the current state, future behavior, and the reward of behavior. There are some basic components of reinforcement learning: action, state, reward, policy, value, and Q-value. ‘Action’ means all the behaviors

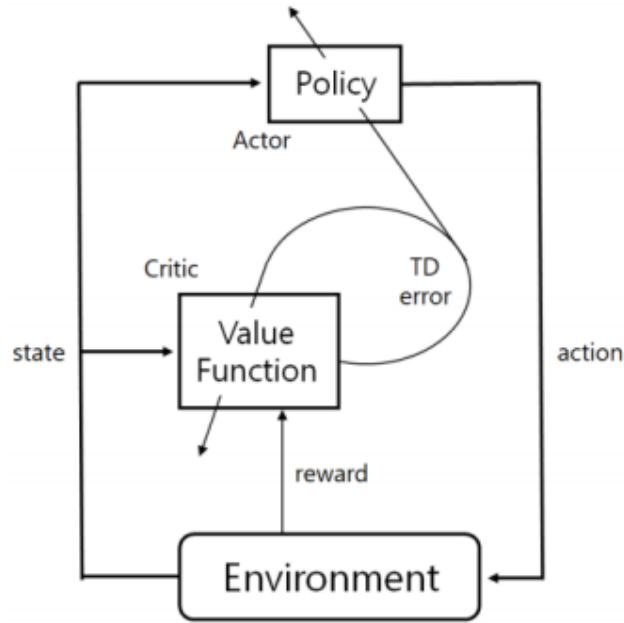
that agents can take. ‘State’ indicates the current state agent received from the environment. ‘Reward’ denotes the value of the last action received from the environment. ‘Policy’ is the action that the agent should take based on the current state. ‘Value’ is the expected reward in the current state. ‘Q-value’ is the expected reward after taking the current action. In the case of active suspension system, the agent is active suspension and the environment is the quarter car model. The state is the displacement and velocity of the car body and wheel. Action is a force from the active system. Reward model is relative to car body’s displacement and velocity. The result from using DDPG algorithm show that the reward of system converges to 0, which confirms that it is working well. Below picture shows a diagram of a reinforcement learning.



**Figure 2-8 Reinforcement Learning Diagram**

The Actor-Critic learning algorithm is a reinforcement learning algorithm. The concept of the Actor-Critic learning algorithm is that it should learn both the policy network and the value network. The policy network is called the actor and the value network is called the critic. The actor receives the current state of the environment and produces an action. The critic receives the state and reward and produces a TD error, which is the difference between the target value and the current value. With the TD error, the actor network updates in the direction suggested by the critic network. Figure 2-9 shows the diagram of Actor-Critic algorithm.



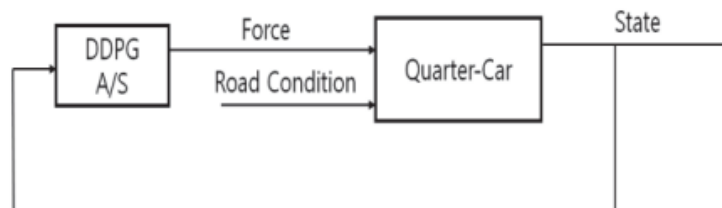


**Figure 2-9 Actor-Critic Algorithm**

In their paper they established a quarter car model to start their analysis, which can be described with a state space form.

To apply the DDPG algorithm, quarter car active suspension system must be connected with DDPG algorithm. In the system, the agent is active suspension and the environment is the quarter car model. The state is the displacement and velocity of the car body and the wheel. The action is the force from the active suspension. The reward model is related to the car body's displacement and velocity. Below is the reward model and diagram of whole control loop.

$$x_1^2 + 0.1\dot{x}_1^2 + 0.001f_s \quad (7)$$



**Figure 2-10 Control Loop**

The author simulated the whole system with the cosine and step road input and claimed that using DDPG algorithm for an active suspension system does contribute to reduction of the vehicle body. The conclusion is based on the application of DDPG algorithm towards active suspension system with two types of road conditions. However apparently, author of the paper applies road excitation only with cosine signal and step signal, which is an ideal situation of the road situation. Therefore, the output of simulation only suggests a reference result which is less meaningful in the real world. A following up work can be commenced with adding road excitation according to the real industry standard.

## 2.5 Stochastic road excitation

Regarding recent research, vehicle riding comfort and handling stability is importantly impacted by road surface roughness [25] [26] [27] [28] [29]. So far, one commonly implemented method to illustrate the stationary road pavement energy is called power spectral density which is also known as PSD[30]. When a car goes on a road with a constant velocity, the road roughness can be considered as a stochastic process in space domain which obey to Gaussian probability distribution[31]. In this case, PSD is an ideal method to describe the distribution of roughness energy in space domain[32]. According to ISO 8608[33], a PSD of road displacement  $G_d$  can be presented in the form of spatial frequency:

$$G_d(n) = G_d(n_0) \left( \frac{n}{n_0} \right)^{-w} \quad (8)$$

Or in the form of angular spatial frequency:

$$G_d(\Omega) = G_d(\Omega_0) \left( \frac{\Omega}{\Omega_0} \right)^{-w} \quad (9)$$

However, it is important to realize that those PSD equations do not take vehicle velocity into account. In the vehicle vibration system, it is essential to consider the speed of the car because it impacts greatly to the dynamic of the whole system.

Considering the assumption that the vehicle drives in a constant velocity, some researches have been carried out by studying the road simulation by using stationary random vibration method. According to Goenaga[34], a methodology of sinusoidal approximation has been applied into the simulation considering that the vehicle travels at a constant speed. A hypothesis is presented that longitudinal road roughness is able to be structured through a group of sinusoidal waves that contain a variety of wavelengths, amplitudes and phases, which can be summarized into the equation below.

$$Z_R(s) = \sum_{i=1}^N A_i \sin(\Omega_i s - \phi_i) \quad (10)$$

where  $A_i$  is defined as

$$A_i = \sqrt{\Phi(\Omega_i) \frac{\Delta\Omega}{\pi}} \quad (11)$$

As argued by Yunqing[35], the vehicle in most cases is travelling in a variable speed considering the vehicle motion of start, acceleration and break. A step forward research has been brought up by him to investigate the road profile as a non-stationary random process. the road roughness can still be obtained through a white noise signal which is following a certain power spectral density with a modification on the equation which has been showed below.

$$\dot{z}_r(t) + 2\pi u(t)n_0 z_r(t) = \sqrt{G_q(\Omega_0)u(t)}w(t) \quad (12)$$

where,  $z_r(t)$  is the road roughness,  $w(t)$  is a white noise signal whose power spectral density is 1. Similar to stationary random process, a simulation can be established in Simulink to check

the performance of the model. Given an initial car speed of 0 m/s, and then being increased to 2 m/s.

However, none of these research have a quantitative analysis on how good is the simulation result in comply with the ISO 8608 standard. It is very important to guarantee the quality of the signal because in a lot of cases, it will be used as an input in driving a vehicle suspension model.

## **2.6 Summary and conclusion**

The target of this project is to study the possibility of applying machine learning into the control of an active suspension system in different pavement situation. According to the literature review that has been conducted in this chapter, attempts have been initiated in integrating reinforcement learning control with suspension system, however, the current works are not perfect, some areas can still be improved. For instance, in simulation, the road pavement signal is designed to be a cosine or step signal which is quite ideal. The algorithms that are applied can sometimes lead to an unstable situation due to an internal error. Method that is used to generate stochastic road signal hasn't taken accuracy into consideration.

In this research, a TD3 reinforcement algorithm is defined as the controlling unit of the whole system, together with an optimized pavement generator that are used to address these research gaps that have been mentioned above. Nonlinearities of the system and hyper parameters of machine learning are analyzed and the relative impacts are also disclosed.

## **Chapter 3. Suspension system modeling**

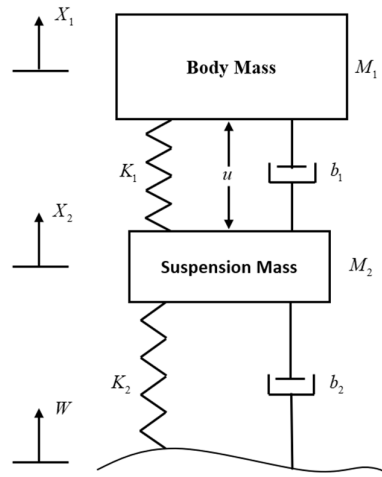
Vehicle longitudinal and vertical analysis form two major part of the current research areas. Vehicle state [36] [37] analysis and calibration [38] [39] [40] [41] [42] [43] [44] are the main direction in working with a longitudinal research, whilst suspension design contributes a lot to the car vertical study. When considering the suspension modeling, road holding ability is important towards an acceptable design of the system, while riding comforts are still provided when running over unstable street. When the car body is having oscillation, suspension should provide the force to cancel the vibration in a short period of time. The traditional passive suspension system is able to store energy through the spring and dissipate it from the damper, however the parameters of which are normally not changeable, therefore, they cannot fit well into different pavement situation. At the same time, active suspension system is gradually forming a trend in the automotive industry and is spreading down from luxury vehicles to the normal ones. In this study a combination of a 2-DOF quarter car model and an actuator is established which is utilized as an active suspension system [45] [46] [47].

A quarter car model with a force actuator is used to work as an active suspension system. Considering the dynamic performance, when a car moves through uneven roads, the spring from suspension is responsible for absorb and store the energy, which is then dissipated in the form of heat by the damper. The purpose of an actuator is to reduce the acceleration of car body and displacement of suspension by getting involve into energy transforming process via applying a force to the suspension with extra energy consumption. Therefore, it is necessary to carry out system modeling through Newton's second law and dynamic system theory.

### **3.1 Theoretical modelling and parameters definition**

Vehicle suspension system design is a typical control problem that can be very challenging because it is often connected with another vehicle research area. Before designing any control

system, it is critical to establish a mathematical model of the suspension system. A single degree multiple spring-damper system can be used to describe a quarter of a car model. There exist some methods of creating such a model, however the damping of the tire is often ignored. In this paper, to guarantee the accuracy of the simulation, a damping coefficient is added to the suspension model, a diagram of this system is shown below Figure 3-1.



**Figure 3-1 Quarter Car Model**

The system parameters are as follows.

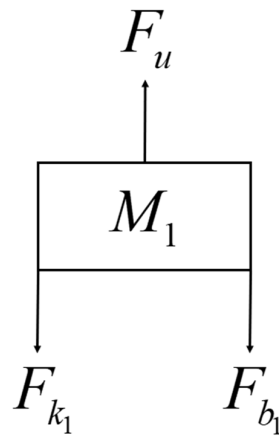
$(M_1)$	body mass	365 kg
$(M_2)$	suspension mass	43 kg
$(K_1)$	spring constant of suspension system	24,000 N/m
$(K_2)$	spring constant of wheel and tire	3500,000 N/m
$(b_1)$	damping constant of suspension system	2,126 N.s/m
$(b_2)$	damping constant of wheel and tire	1,000 N.s/m
$(u)$	control force from the actuator	

### 3.2 Systematic force analysis

According to Figure 3-1, assuming  $X_1 > X_2$ ,  $X_2 > W$ .

To be able to model the system, it is necessary to sum the forces that act on both masses (body and suspension) and integrate the accelerations of each mass twice so that velocities and positions can be granted.

First, only considering the forces that have been applied to  $M_1$ , there are three forces that acts on  $M_1$ , one is from spring, the others are from damper and actuator which is the input  $u$ , the analysis result is showed in Figure 3-2



**Figure 3-2 Car Body Force Analysis**

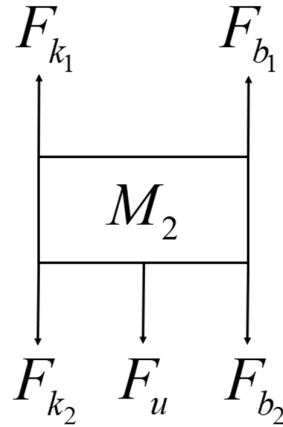
From above diagram, by applying Newton's second law, the differential equation of longitudinal motion of  $M_1$  can be obtained as below

$$\begin{cases} M_1 a_1 = F_u - F_{k_1} - F_{b_1} \\ F_{k_1} = K_1(X_1 - X_2) \\ F_{b_1} = b_1(V_1 - V_2) = b_1(\dot{X}_1 - \dot{X}_2) \end{cases} \quad (13)$$

Based on the above three equations, it can be derived that

$$M_1 \ddot{X}_1 = F_u - K_1(X_1 - X_2) - b_1(\dot{X}_1 - \dot{X}_2) \quad (14)$$

Second, only considering the forces that have been applied to  $M_2$ , based on the assumption,  $M_2$  has five forces act on it. Two come from springs, two come from dampers, and the last is from the actuator, the analysis result is showed in FIGURE 3-3.



**Figure 3-3 Suspension Force Analysis**

Through the same strategy, the motion of  $M_2$  can be summarized by a group of equations showed below

$$\begin{cases} M_2 a_2 = F_{k_1} + F_{b_1} - F_{k_2} - F_{b_2} - F_u \\ F_{k_1} = K_1(X_1 - X_2) \\ F_{b_1} = b_1(V_1 - V_2) = b_1(\dot{X}_1 - \dot{X}_2) \\ F_{k_2} = K_2(X_2 - W) \\ F_{b_2} = b_2(V_2 - V_W) = b_2(\dot{X}_2 - \dot{W}) \end{cases} \quad (15)$$

Based on the above three equations, it can be derived that

$$M_2 \ddot{X}_2 = K_1(X_1 - X_2) + b_1(\dot{X}_1 - \dot{X}_2) - K_2(X_2 - W) - b_2(\dot{X}_2 - \dot{W}) - F_u \quad (16)$$

Differential equations (14) and (16) have summarized the dynamic relationship of the quarter car suspension model.



### 3.3 State space form of quarter car model

A dynamic system can be represented by a group of differential equations because of its property. In a lot of cases, a transfer function is a good way to describe a single input and single output system. However, when they system requires a multiple inputs and outputs, a transfer function is not suitable in this situation, and a state space method can be chosen to solve the problem. High order differential equations can be repacked into a set of first order differential equations by using state space method, which makes the system being analyzed easily.

For a continuous linear time-invariant system, the state space form can be presented as below

$$\begin{aligned}\dot{X} &= AX + BU \\ Y &= CX + DU\end{aligned}\quad (17)$$

To be able to derive the state space model of our quarter car model, equation (14) and equation (16) has been rewritten as

$$\begin{cases} \ddot{X}_1 = -\frac{K_1}{M_1}(X_1 - X_2) - \frac{b_1}{M_1}(\dot{X}_1 - \dot{X}_2) + \frac{F_u}{M_1} \\ \ddot{X}_2 = \frac{K_1}{M_2}(X_1 - X_2) + \frac{b_1}{M_2}(\dot{X}_1 - \dot{X}_2) - \frac{K_2}{M_2}(X_2 - W) - \frac{b_2}{M_2}(\dot{X}_2 - \dot{W}) - \frac{F_u}{M_2} \end{cases}\quad (18)$$

Define state matrix

$$X = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \end{bmatrix} = \begin{bmatrix} X_1 \\ \dot{X}_1 \\ X_2 \\ \dot{X}_2 \end{bmatrix}\quad (19)$$

Then equation (19) and equation (18) can be rewritten into a state space form

$$\begin{cases} \dot{a}_1 = a_2 \\ \dot{a}_2 = -\frac{K_1}{M_1}(a_1 - a_2) - \frac{b_1}{M_1}(a_2 - a_4) + \frac{F_u}{M_1} \\ \dot{a}_3 = a_4 \\ \dot{a}_4 = \frac{K_1}{M_2}(a_1 - a_3) + \frac{b_1}{M_2}(a_2 - a_4) - \frac{K_2}{M_2}(a_3 - W) - \frac{b_2}{M_2}(a_4 - \dot{W}) - \frac{F_u}{M_2} \end{cases}\quad (20)$$

Define input matrix

$$\begin{bmatrix} w_1 \\ w_2 \end{bmatrix} = \begin{bmatrix} W \\ \dot{W} \end{bmatrix} \quad (21)$$

$$U = \begin{bmatrix} F_u \\ w_1 \\ w_2 \end{bmatrix} \quad (22)$$

Combined with equation (21), equation (20) can be rewritten into below form

$$\begin{cases} \dot{a}_1 = a_2 \\ \dot{a}_2 = -\frac{K_1}{M_1}a_1 - \frac{b_1}{M_1}a_2 + \frac{K_1}{M_1}a_3 + \frac{b_1}{M_1}a_4 + \frac{1}{M_1}F_u \\ \dot{a}_3 = a_4 \\ \dot{a}_4 = \frac{K_1}{M_2}a_1 + \frac{b_1}{M_2}a_2 - \frac{K_1+K_2}{M_2}a_3 - \frac{b_1+b_2}{M_2}a_4 + \frac{K_2}{M_2}w_1 + \frac{b_2}{M_2}w_2 - \frac{1}{M_2}F_u \end{cases} \quad (23)$$

According to the previous study, the most important fact that we are concerning is the acceleration of the vehicle body and the displacement of the suspension system because they are directly related to the riding comfort and road handling. Based on this, an output matrix can be decided

$$\alpha = a_1 - a_3 \quad (24)$$

$$Y = \begin{bmatrix} \dot{a}_2 \\ \alpha \end{bmatrix} \quad (25)$$

Based on the above investigation, the quarter car suspension model can be summarized into a state space model form

$$\begin{aligned} \dot{X} &= AX + BU \\ Y &= CX + DU \end{aligned}$$

where,

$$X = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \end{bmatrix}, U = \begin{bmatrix} F_u \\ w_1 \\ w_2 \end{bmatrix}, Y = \begin{bmatrix} \dot{a}_2 \\ \alpha \end{bmatrix},$$

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -\frac{K_1}{M_1} & -\frac{b_1}{M_1} & \frac{K_1}{M_1} & \frac{b_1}{M_1} \\ 0 & 0 & 0 & 1 \\ \frac{K_1}{M_2} & \frac{b_1}{M_2} & -\frac{K_1+K_2}{M_2} & -\frac{b_1+b_2}{M_2} \end{bmatrix},$$

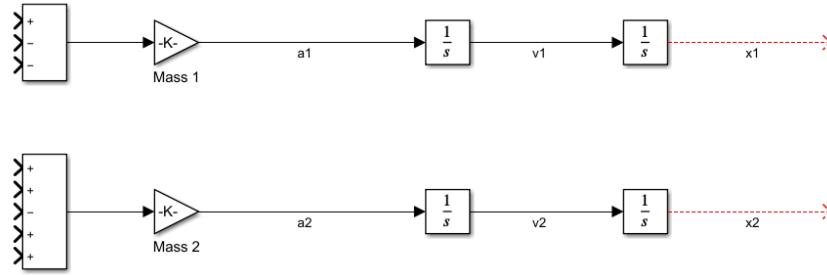
$$B = \begin{bmatrix} 0 & 0 & 0 \\ \frac{1}{M_1} & 0 & 0 \\ 0 & 0 & 0 \\ -\frac{1}{M_2} & \frac{K_2}{M_2} & \frac{b_2}{M_2} \end{bmatrix},$$

$$C = \begin{bmatrix} -\frac{K_1}{M_1} & -\frac{b_1}{M_1} & \frac{K_1}{M_1} & \frac{b_1}{M_1} \\ 1 & 0 & -1 & 0 \end{bmatrix},$$

$$D = \begin{bmatrix} \frac{1}{M_1} & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

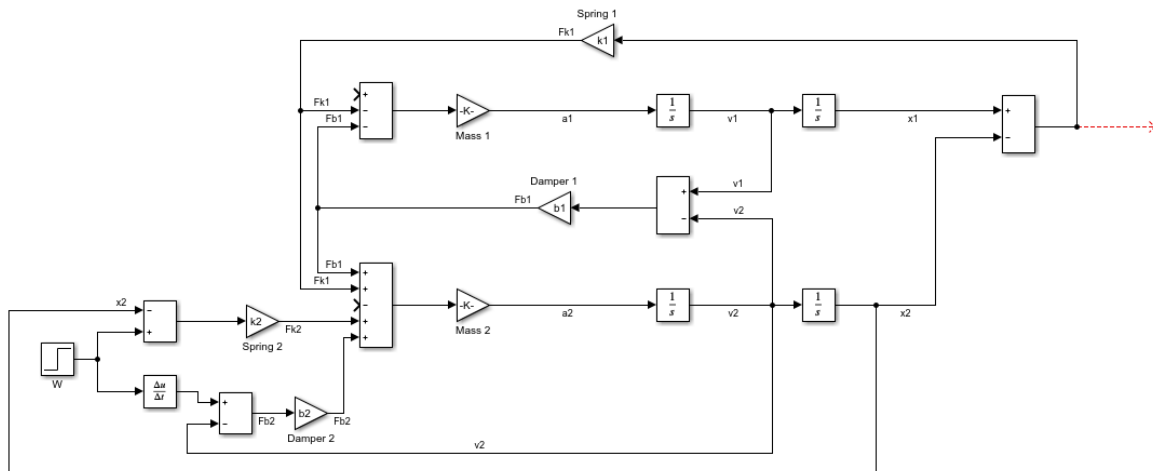
### 3.4 Simulation of the suspension model

Based on the above work, it is necessary to realize the simulation in Simulink, so that it can be further used to connect with machine learning algorithm and the road excitation signal. The blocks that represent the summary of the forces have been added and linked with gain blocks that work as the masses of the system. Then they have been connected with integral blocks to work out the velocity and displacement. Below picture demonstrates these relationships.



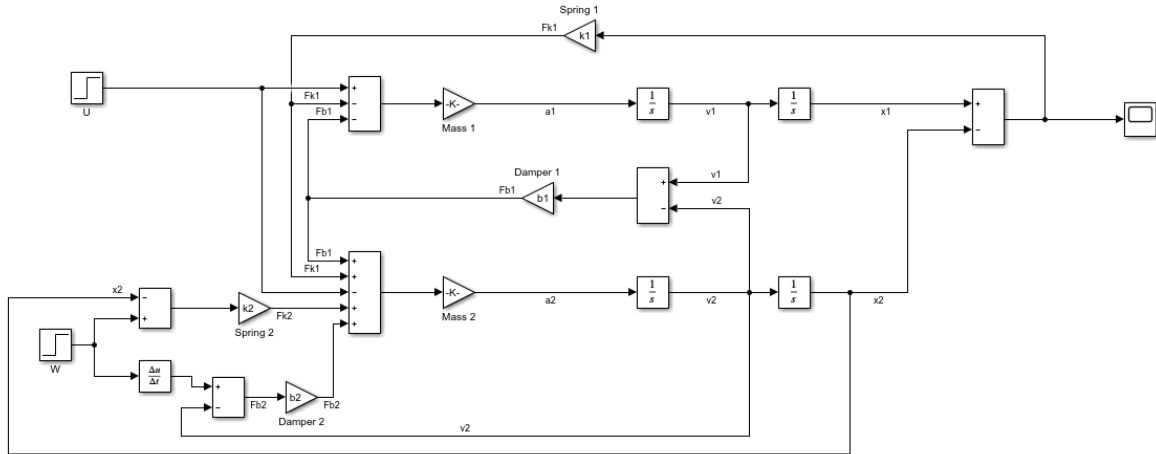
**Figure 3-4 Mass Blocks and Force Blocks**

After that, spring forces and damper forces are looped back to the force block which drives the system.



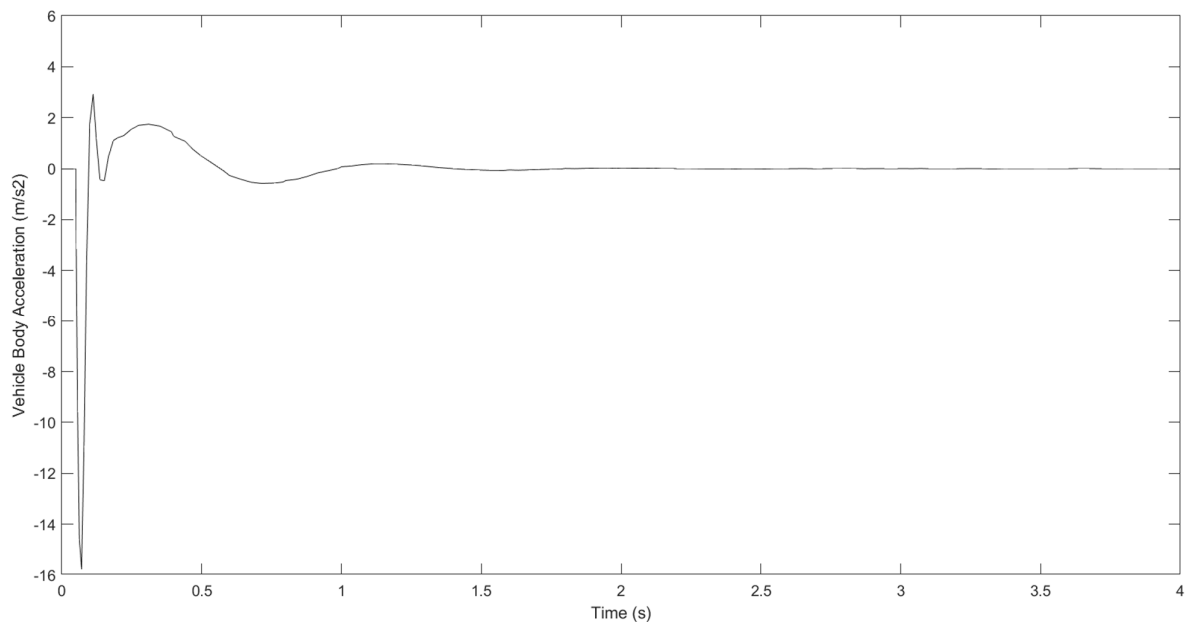
**Figure 3-5 Spring Forces and Damper Forces**

The last job is to add actuator force  $U$  to the system, then a complete quarter car model is showed in below diagram. With the knowledge I learned from the Matlab Advanced tutorial, I was able to create the whole system in Simulink which allows me to continue my study.

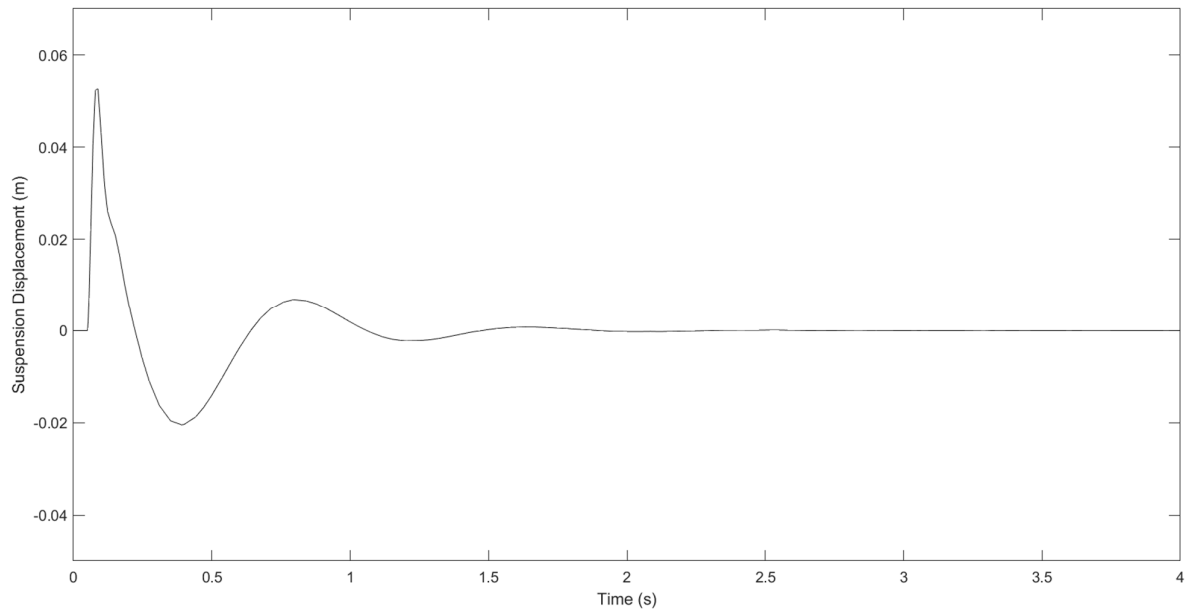


**Figure 3-6 Quarter Car Model**

After input the parameters that has been defined previously, the model is ready for simulation. Set the force  $U$  to zero, because we assume that no actuator force is created in this stage. Set another assumption that the car is running over a pothole with 0.05m in depth. There for in the step input block  $W$ , set the final value to -0.05. Then run the simulation in Simulink. When the simulation is done, the following chart can be presented.

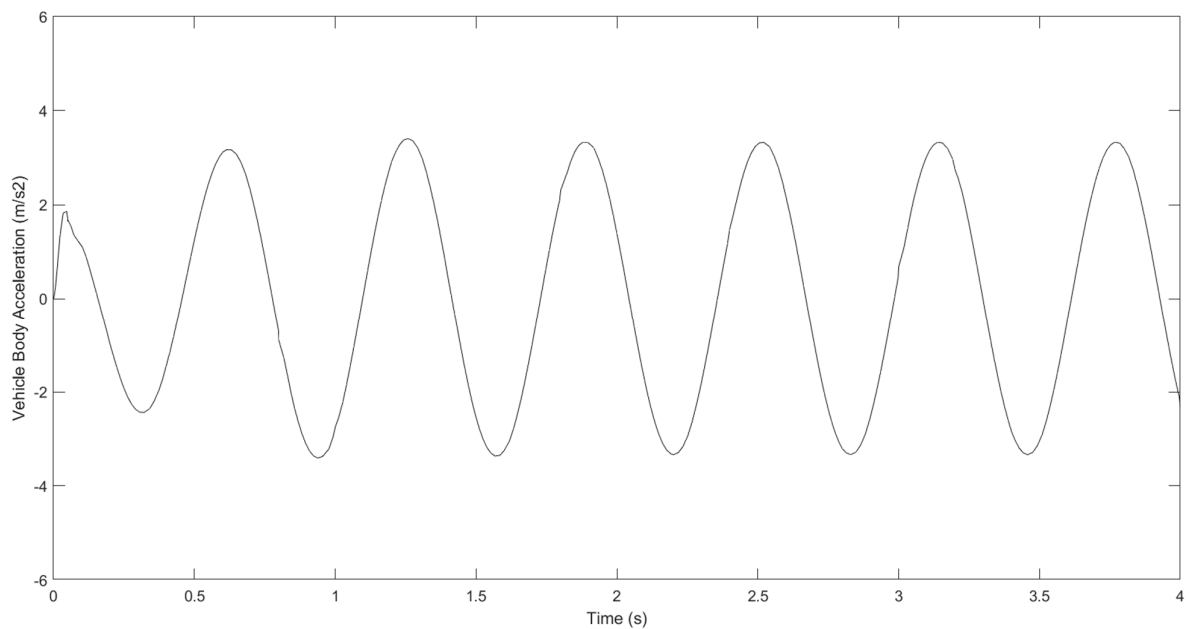


**Figure 3-7 Body Acceleration with Step Input**

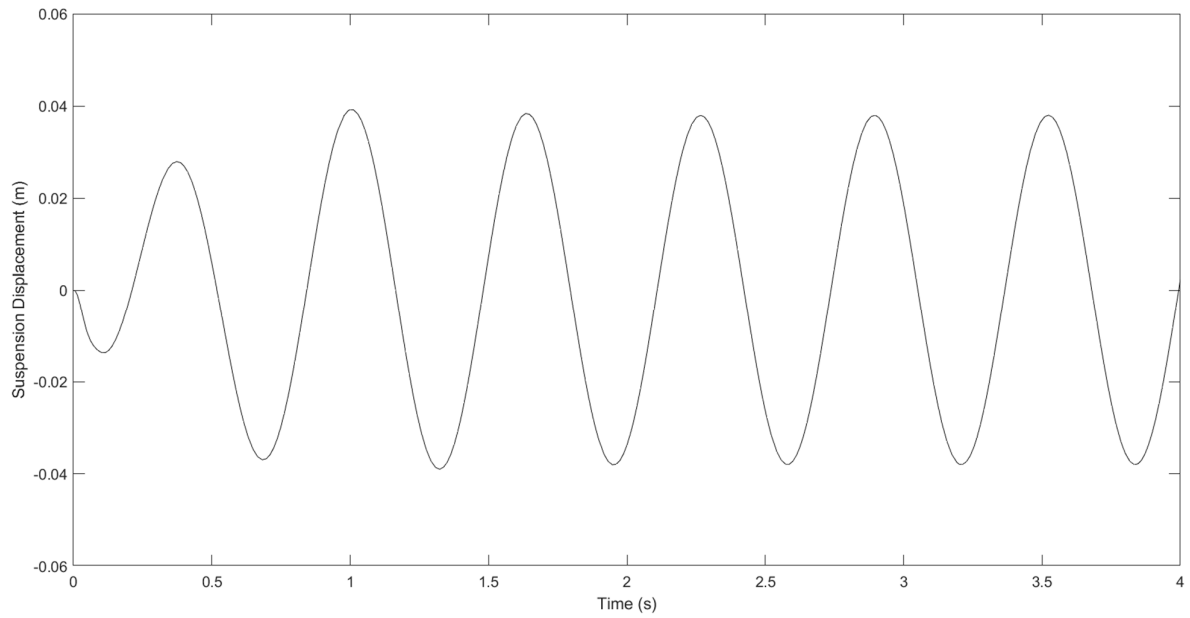


**Figure 3-8 Suspension Displacement with Step Input**

Change the input from step signal to a sinusoidal input to simulate a bumpy road condition. Set the signal frequency to 10 rad/sec and amplitude to 0.025 m. The result is displayed in the below diagram.



**Figure 3-9 Body Acceleration with Sinusoidal Input**



***Figure 3-10 Suspension Displacement with Sinusoidal Input***

The quarter car suspension system has been successfully simulated, and two interfaces have been prepared for a reinforcement learning controller and a road excitation signal to plug, so that a next stage study can be spread out.

## Chapter 4. Reinforcement learning frame definition

### 4.1 Introduction of reinforcement learning

In reinforcement learning, an agent is interacting with an unknown environment by choosing an action in a sequence of time steps based on observation and trying to achieve a maximum of reward accumulation. This process can be modeled as a Markov decision process (MDP) which consists of a state space  $S$ , an action space  $A$ , an initial state distribution with density, a stationary transition dynamics distribution and a reward function  $r$  [48]. The agent then implements its policy to work with the Markov decision process to get a set of states, actions and rewards so that direction of gradient can be revealed.

There are basically three types of learning algorithm that form the category of machine learning. The first one is called unsupervised learning which is used to determine the pattern or hidden structure of a data set that has not been classified or labelled. The next is known as supervised learning which aims to train a computer to label a given input. The last one is reinforcement learning which has been considered as a completely different approach. Supervised and unsupervised learning both use static data sets, whilst reinforcement learning uses dynamic environmental data and has a lot of current applications in reality [49] [50] [51] [52] [53]. The purpose of doing so is not to classify or label the data, but to determine the best sequence of actions that produces the optimal result. To achieve that target, reinforcement learning uses an agent to explore the environment, interact with the environment, and learn from the environment. This approach is perfectly suitable in controlling the force of an actuator when interacting with an active suspension system.

There is a function in the agent to receive state observations (inputs) and map them to action sets (Output). In reinforcement learning, this function is called a strategy. Strategy is a logic function that is composed of editable and adjustable parameters which are decided by the actions taken, the observations of environment state and the reward value obtained. In a

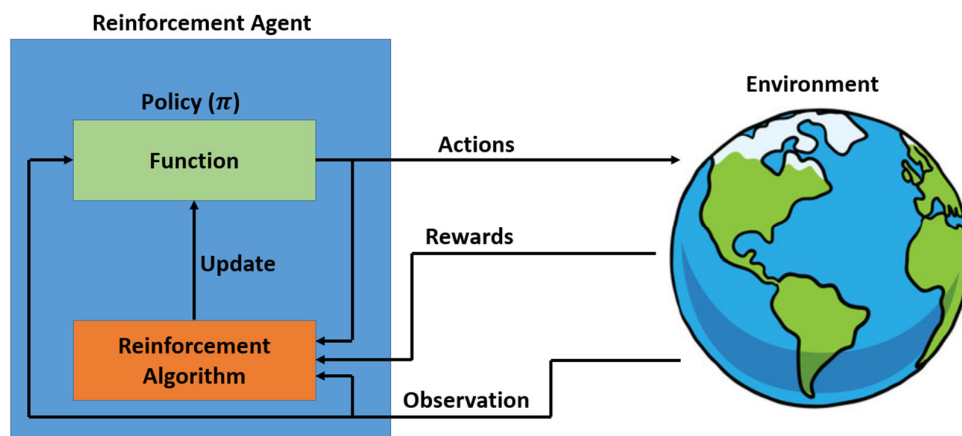


summary, it works based on a given set of observation and then determines the action to be taken.

The ultimate purpose of the agent is to work out the best environmental interaction strategies through learning process. So that the agent can always take the optimal action in any state which guarantees that a maximum long-term reward can be obtained.

The learning process is generally an effort which systematically adjusts these parameters to converge to the optimized strategy. In this way, it is possible to just focus on setting up an appropriate structure of the strategy without having to manually adjust the entire function to get the best parameters.

The basic reinforcement learning concept is showed below Figure 4-1:



**Figure 4-1 Reinforcement Learning Concept Diagram**

Policy gradient algorithm is widely applied in dealing with reinforcement learning projects.

The fundamental concept is to construct a policy  $\pi_{\theta} = (a|s)$  and makes it equal to a probability distribution  $P[a|s; \theta]$  which is able to select an action  $a$  randomly in the state  $s$  based on a parameter vector  $\theta$ . A policy gradient algorithm will change the policy parameter in regarding to a situation with higher reward accumulation through policy sampling.

## 4.2 Establishment of environment, observation space and action space

The target of the agent is to utilize the reinforcement learning algorithm to obtain the optimized policy in interacting with the environment. No matter how an environment changes, the agent can guarantee a group of corresponding actions will be chosen according to that change, to get the maximized reward in the long run. Policy is actually a function or a group of functions with adjustable parameters. Parameters of functions can be modified by reinforcement learning algorithm based on a current action, observation and rewards. The process of parameters optimization is called learning.

Reinforcement learning workflows:

1. Establish environment
2. Define action
3. Define observation
4. Design reward function
5. Select a policy
6. Select an algorithm to train the agent
7. Apply the policy

Environment in this study is the suspension system of a quarter car model, it can not only receive the actions that are created by the agent, but also output the observations and rewards back to the agent. The environment has been established in sub task 1.

The best thing about reinforcement learning is that the agent is not necessary to carry any knowledge on the environment, however, still be able to work out how to be interactive with it. The action of the model is defined as a force which is created by the actuator of the active suspension system. The action space is a discrete space and has been finalized as a force interval from -4000N to +4000N which has an increment of 500N. The observation contains

three signal input, they are acceleration of the car body, displacement of the suspension system and the force from suspension spring.

### 4.3 Reward function definition and simulation

Reward is a function of “state” and “action”, which output a scalar representing “goodness”.

**Reward = function (State, Action)**

The reward function is close to a cost function used in LQR to penalize poor system performance. Of course, the difference is that the cost function tries to minimize the cost, while the reward function is to maximize the reward. But they are actually resolving a same problem, because reward can be seen as the opposite of cost. The instantaneous reward can be defined as below equation:

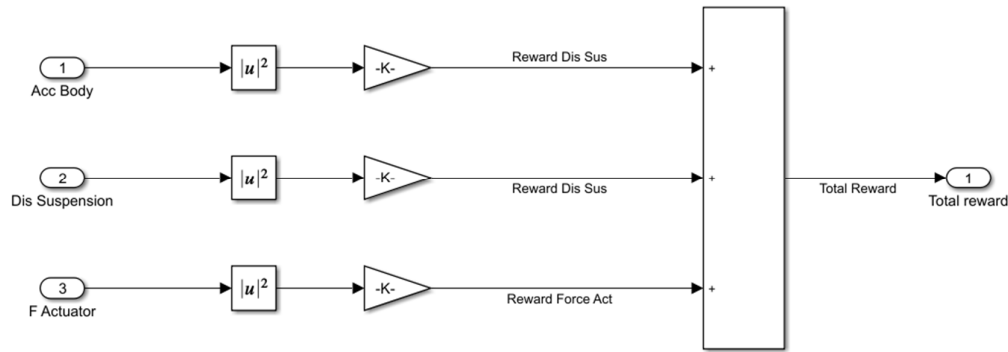
$$r_i = s_i^T Q_i s_i + a_i^T R_i a_i + 2s_i^T N_i a_i \quad (26)$$

where Q, R, N are the weight matrix, s is the observation vector and a is the action vector.

By combining the idea from [54] and [55], the reward signal is defined as below

$$r(t) = P_1 Acc(t)^2 + P_2 Dis(t)^2 + P_3 F(t)^2 \quad (27)$$

where,  $P_1, P_2, P_3$  are the weight scalars,  $Acc(t)$  is the vehicle body acceleration in t,  $Dis(t)$  is the suspension displacement in t,  $F(t)$  is the actuator force in t. Based on this concept, a reward signal block can be created in the Simulink as below Figure 4-2.



*Figure 4-2 Rewards Signal*

#### 4.4 Policy and neural network definition

The policy of the project has been defined by using Q-table which maps states and actions to values. The Q-table strategy will check the value of each possible action based on the current state, then choose the action with the highest value. Using Q-table to train the agent includes determining correct value from the table corresponding to each pair of state and action. After the table is completely filled with the correct values, the selection with maximized long-term rewards is quite straightforward.

Regarding to the agent training algorithm, in this study, a critic network which contains a certain structure of neural network has been chosen to perform the task. There are currently a diversity of ways in utilizing neural network application into the project design [56] [57] [58], and also there are a lot of different types of network architectures which are available in dealing with various tasks [59] [60] [61].

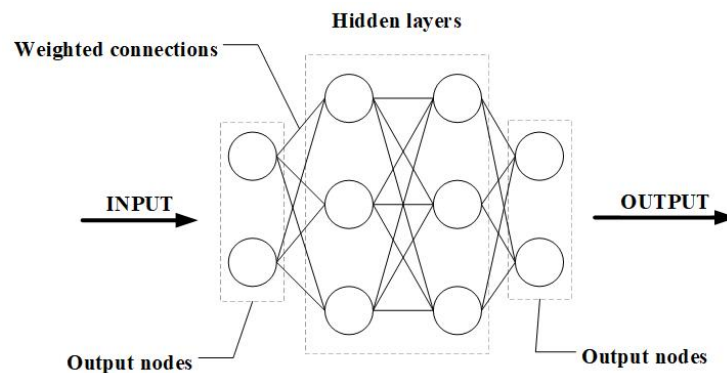
Neural network works as a general function approximator that imitates any input and output relationship and is expected to be one of the key factors in this study. Intuitively, a more complex input output relation requires a more complex neural network structure [49]. However, the situation can be different considering the application in various environment. Therefore, it is necessary to define the architecture of the network prior to implement different algorithms [50] [52].

Although the target imitating function itself can be very complicated, the general nature of neural networks can ensure that some kind of neural network can achieve the goal.

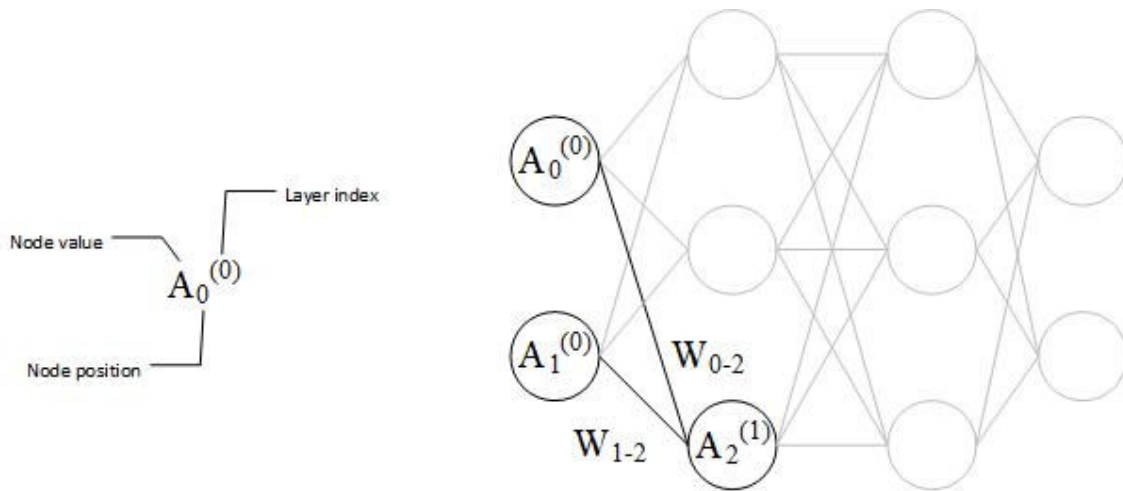
Therefore, comparing of finding a nonlinear structured function that is suitable for a typical scenario, it is better to use a neural network with the same combination of nodes and connections so that it can be applied into many different environments. The only difference is parameter configuration, learning process will conduct a systematically parameters optimization to find the best matching between input and output. A simple neural network has a basic structure illustrated in Figure 4-3.

The nodes on left side are input nodes, a node corresponds to an input of the function, and the nodes on right side are output nodes. The layers in the middle are called the hidden layers. This network has two inputs, two outputs, and 2 hidden layers, which contains three nodes in each layer. For a fully connected network, there is a weighted connection that connects each node from different layers.

To describe this network in a math perspective is that the any value from a specific node is equal to the sum of all input nodes that have been fed into that node plus a bias, and each of the input node needs to be multiplied with its respective weight coefficient. This relationship has been illustrated in Figure 4-4 and the math equation can be described in equation (28).



**Figure 4-3 Basic Neural Network Structure**



**Figure 4-4 Neural Network Node Value Transfer**

$$A_2^{(1)} = W_{0-2} \cdot A_0^{(0)} + W_{1-2} \cdot A_1^{(0)} + B_2^{(1)} \quad (28)$$

This calculation can be performed on any node in any layer and what can be granted is a group of linear equations in the form of compact matrix which is showed in equation (29), and the purpose of which is to transfer the value from one layer of nodes to the next layer of nodes.

$$\begin{cases} A^{(1)} = W^{(1)} \cdot A^{(0)} + B^{(1)} \\ A^{(2)} = W^{(2)} \cdot A^{(1)} + B^{(2)} \\ A^{(3)} = W^{(3)} \cdot A^{(2)} + B^{(3)} \\ \dots \end{cases} \quad (29)$$

Now the question is how can a series of cascaded linear equations act as a general function approximator? To be more specific, how do they work as non-linear functions? There is still one important piece missing in the chain, the activation function. An activation function can be applied to change a node value before it is delivered to the next layer. The mathematical expression of this concept is showed in equation (30).

$$A^{(1)} = f_{activation}(W^{(1)} \cdot A^{(0)} + B^{(1)}) \quad (30)$$

There are several different activation functions. The common feature of those functions is nonlinearity, which is essential for constructing a network that can approximate any function. The reason is that many nonlinear functions can be simulated as a combination of a group of weighted activation functions. Below are the two commonly used activation functions.

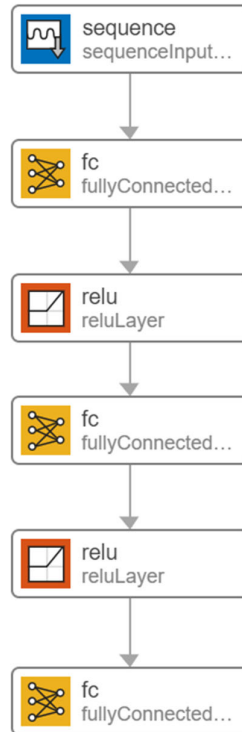
$$f(x) = \sigma(x) = \frac{1}{1 + e^{-x}} \quad (31)$$

$$\begin{cases} f(x) = 0, x \leq 0 \\ f(x) = x, x > 0 \end{cases} \quad (32)$$

Equation (31) is the called sigmoid activation function. It generates a smooth curve, which makes any input between negative infinity and positive infinity be compressed into a value between zero and one. Equation (32) is called ReLU function. It can make any negative node value become zero, while keep positive value remains no change.

Through above investigation, the neural network now can be explained in a mathematical way that allows us to utilize it into the project design.

The critic network that has been applied in this study is basically a fully connected neural network that contains one input layer, one output layer and two hidden layers. The structure is showed in below picture.



*Figure 4-5 Critic Network Structure*

## 4.5 Reinforcement learning frame assembly

Based on above design, the reinforcement learning frame has been established in Simulink which has been showed in Figure 4-6.

The action space is connected with the suspension block to provide a live controlling force. The observation space is fed with the signals from suspension model which contains the information of acceleration, displacement and controlling force. By adding the reward signal, the whole reinforcement model is ready to work.

Some critical parameters have been defined in following context:

1. Observation definition:

Observation 1 is the acceleration of the car body, observation 2 is the displacement of the suspension system, and observation 3 is the force of the actuator.

2. Action space definition:

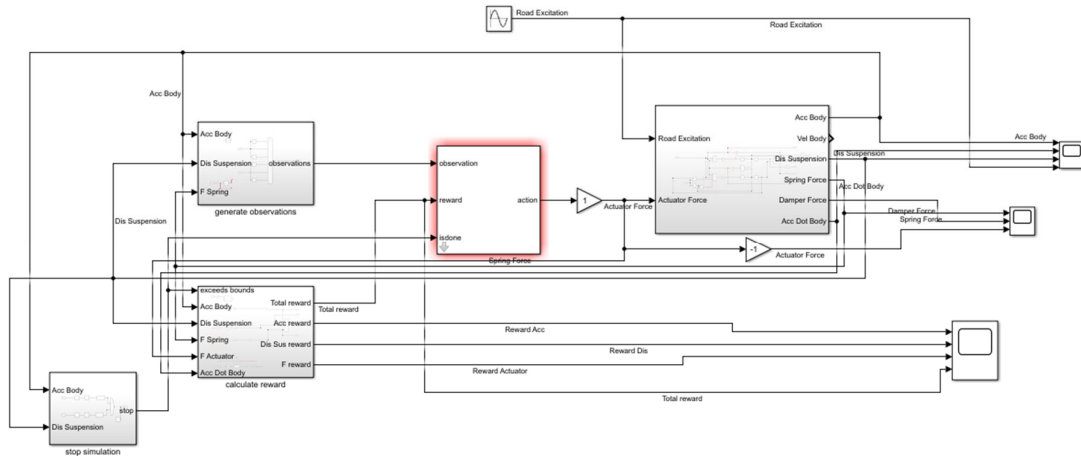


The action space is defined as a space of force which is created by the actuator. The range of the force is from -4000 N to +4000 N, with an increase of 500 N.

Therefore, the whole space consists of 15 elements of force.

### 3. Sample time and learning period definition

Sample time is defined as  $T_s = 0.01s$  and learning period is defined as  $T_f = 10s$



**Figure 4-6 Reinforcement Learning Frame Structure**

### 4. Critic network (neural network) definition

Critic Network	Remarks
1 Sequence Input Layer	3 observation input
2 Fully Connected Layer	24 nodes
3 ReLu Layer	
4 Fully Connected Layer	48 nodes
5 ReLu Layer	
6 Fully Connected Layer	1 action output

### 5. Other parameters

Learning rate is 0.001

Gradient threshold is 1

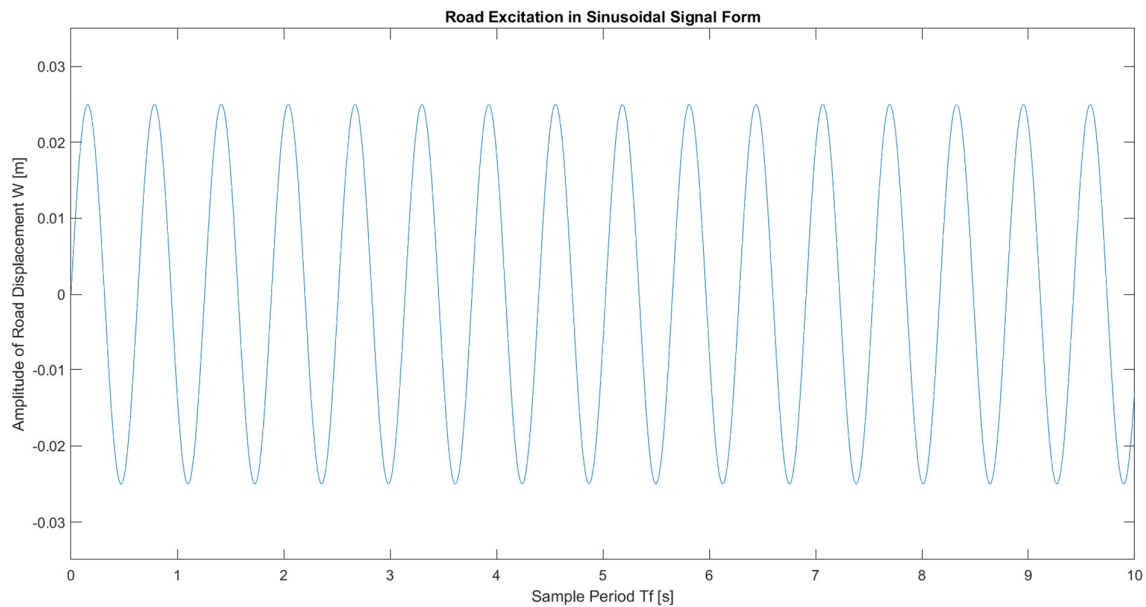
Discount factor is 0.9

Greedy exploration epsilon is 0.99

Maximum episode is 2500

#### 4.6 System test with sinusoidal signal

In the first-round test, a sinusoidal signal has been implemented in the test to work as a road roughness input. The sinusoidal signal has been defined of having an amplitude of 0.025m and a frequency of 10 rad/sec, the diagram of the signal is showed in below Figure 4-7.

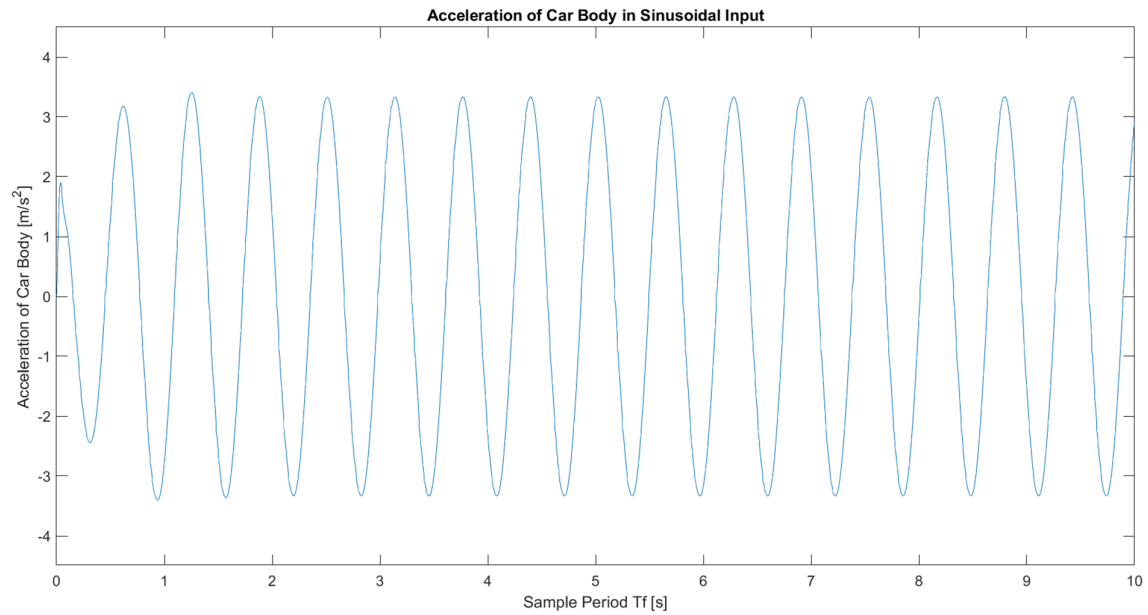


**Figure 4-7 Road Excitation in Sinusoidal Signal Form**

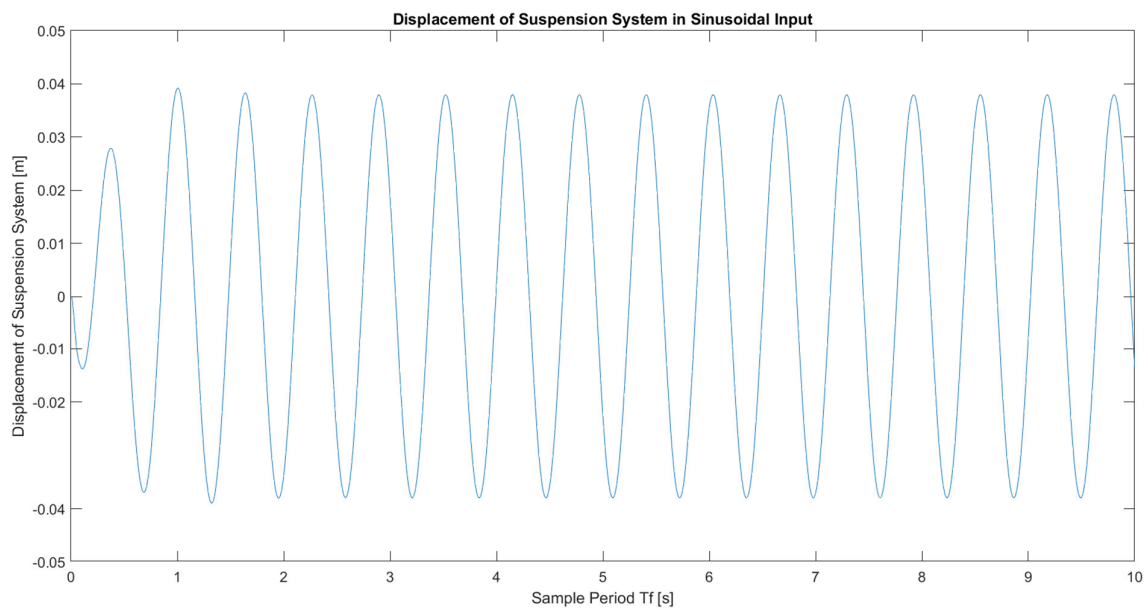
After connecting the sinusoidal signal to the suspension system, the dynamic of the system can be expressed in a way of focusing on two key elements, which are the acceleration of the vehicle body and the displacement of the suspension system. The result has been illustrated in below Figure 4-8 and Figure 4-9.

Next step is to plug in the reinforcement learning algorithm to the whole system and then start training the agent according to the parameters that have been defined previously. The total training period contains 2500 episode, and each episode contains 1000 steps. Training result has been illustrated in Figure 4-10, from the learning result, it can be concluded that the reward

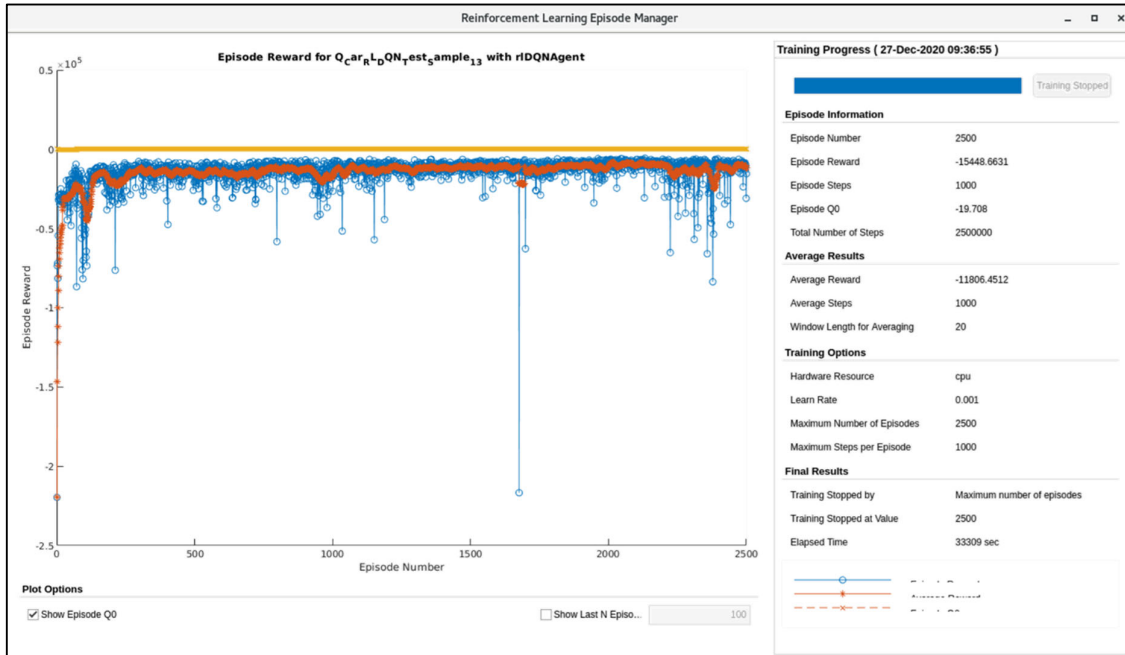
starts to become stable after 500 episodes, and there is no significant improvement from episode 500 to the end of learning.



**Figure 4-8 Acceleration of Car Body in Sinusoidal Input**

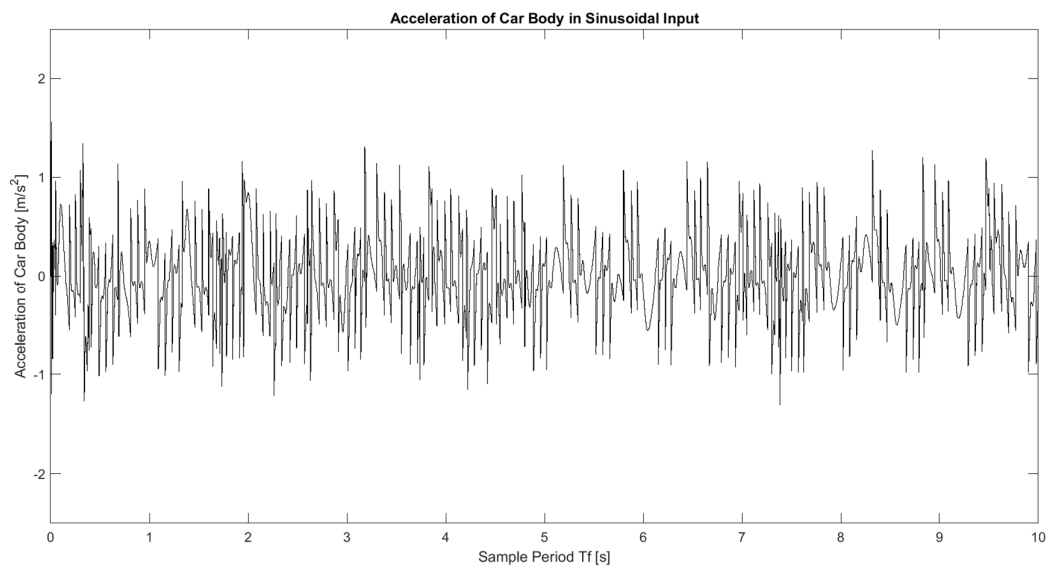


**Figure 4-9 Displacement of Suspension in Sinusoidal Input**



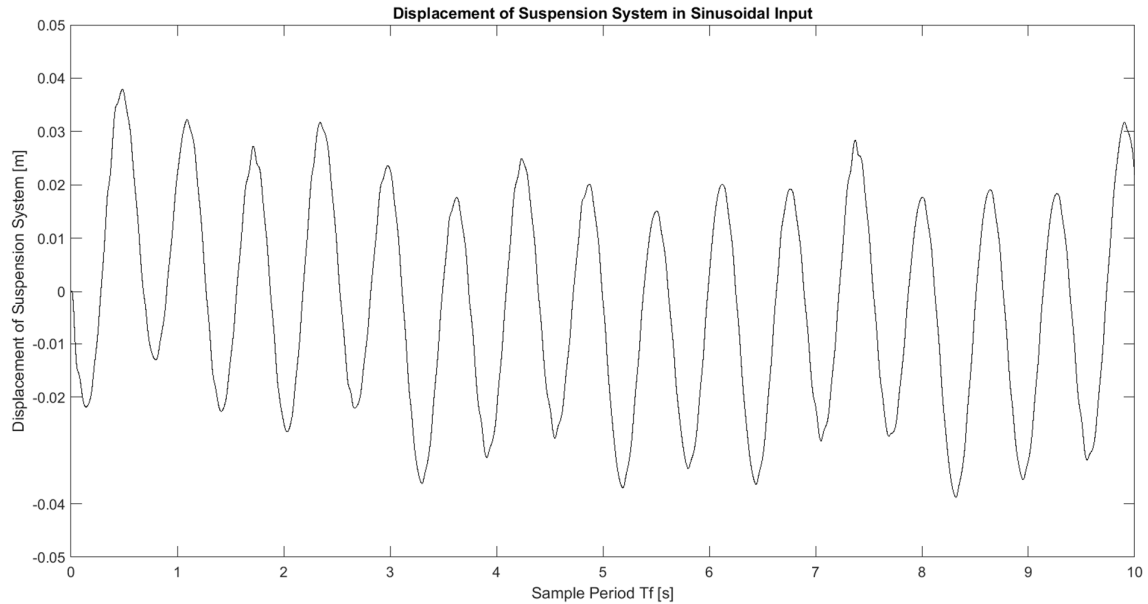
**Figure 4-10 Training Result with Sinusoidal Input**

After the actuating force was controlled via the reinforcement learning agent, the result shows a positive optimization. The acceleration has been improved significantly. The peak-to-peak amplitude has been decreased 57.8% from  $6.804 \text{ m/s}^2$  to  $2.871 \text{ m/s}^2$ . And the Root Mean Square of the signal has been decreased 81.4% from 2.181 to 0.404. The result is showed in below Figure 4-11.



**Figure 4-11 Acceleration of Car Body after Control**

The displacement of suspension also has a small improvement. The peak to peak amplitude has been decreased 1.8% from 0.07814 m to 0.07675 m. And the Root Mean Square of the signal has been decreased 27.2% from 0.02346 to 0.01844. The result is showed in below Figure 4-12.



***Figure 4-12 Displacement of Suspension after Control***

The current test is only based on a sinusoidal input signal to work as road excitation, a further investigation will be carried out once a better road roughness simulation is ready.

## Chapter 5. Road excitation signal generation

Road pavement model plays a critical part in vehicle design industry, especially in suspension design. Therefore, it is vital to guarantee the quality of the signal. The accuracy of the pavement simulation can have a significant influence in suspension analysis or in other general vehicle level investigations [62] [63]. Currently, one tool that is widely used to simulate stationary road roughness is called power spectral density (PSD) [34]. When a vehicle is running on the road with a constant velocity, the road pavement is able to be considered as a space domain stochastic process. It is normally represented as a Gaussian probability distribution [31]. Therefore, PSD is an advanced solution to deliver the distribution of roughness energy especially within space domain [32] [33] [64]. In a vehicle vibration system simulation, it is also essential to consider the speed of the vehicle because it impacts greatly to the dynamic of the whole system.

### 5.1 Modeling

PSD of road disturbance input can be presented by [33, 65]

$$G_q(n) = G_q(n_0) \left( \frac{n}{n_0} \right)^{-w} \quad (33)$$

where  $n$  is spatial frequency in  $m^{-1}$ .  $n$  is the reciprocal of  $\lambda$  which is wavelength,  $n_0$  is reference of spatial frequency in  $m^{-1}$ . Normally,  $n_0 = 0.1 m^{-1}$ .  $G_q(n_0)$  is the power spectral density of road roughness when  $n = n_0$ . It is also known as road roughness coefficient in  $m^3$ . It represents different grades of road.  $w$  is frequency index, it indicates the frequency structure of the road roughness, in most cases,  $w = 2$ .

**Table 5-1 Road Roughness Value Classified by ISO [33]**

degree of roughness $G_q(n_0)$ in $10^{-6} m^3$ where $n_0 = 0.1/m$			
Road class	Lower limit	Geometric mean	Upper limit
A (very good)	-	16	32
B (good)	32	64	128
C (average)	128	256	512
D (poor)	512	1024	2048
E (very poor)	2048	4096	8192

Define angular spatial frequency  $\Omega$  :

$$\Omega = 2\pi n \quad (34)$$

Define reference angular spatial frequency  $\Omega_0$  :

$$\Omega_0 = 2\pi n_0 \quad (35)$$

So equation (33) can be rewritten as:

$$G_q(\Omega) = G_q(\Omega_0) \left( \frac{\Omega}{\Omega_0} \right)^{-w} \quad (36)$$

where  $\Omega$  is spatial angular frequency in  $\text{rad}/m^{-1}$ .  $\Omega_0$  is reference of spatial angular frequency, normally  $\Omega_0 = 1 \text{ rad}/m^{-1}$ .  $G_q(\Omega_0)$  is the power spectral density of road roughness when  $\Omega = \Omega_0$ .  $w$  is frequency index, it indicates the frequency structure of the road roughness, in most cases,  $w = 2$ .

**Table 5-2 Road Roughness Value with Angular spatial frequency unit [33]**

degree of roughness $G_q(n_0)$ in $10^{-6} m^3$ where $\Omega_0 = 1rad / m$			
Road class	Lower limit	Geometric mean	Upper limit
A (very good)	-	1	2
B (good)	2	4	8
C (average)	8	16	32
D (poor)	32	64	128
E (very poor)	128	256	512

Both equation (33) and equation (36) can be used to express the power spectral density of road roughness. When a vehicle is driving at a constant speed  $u$  and passing by a road with a spatial frequency  $n$ , the time frequency  $f$  can be presented with following equation

$$f = u \times n \quad (37)$$

Meanwhile the relationship between PSD in time frequency and spatial frequency can be explained as

$$G_q(f) = \frac{1}{u} G_q(n) \quad (38)$$

Then another general form of power spectral density of road roughness can be derived from a combination of equation (37), equation (38) and equation (33).

$$G_q(f) = G_q(n_0) \frac{n_0^2 u}{f^2} \quad (39)$$

Considering the relationship between time angular frequency  $\omega$  and time frequency  $f$

$$\omega = 2\pi f \quad (40)$$

Combining equation (40) and equation (34), below equation can be deduced.



$$\frac{\omega}{\Omega} = \frac{2\pi f}{2\pi n} = \frac{f}{n} = u \quad (41)$$

Therefore

$$\omega = u \times \Omega \quad (42)$$

Then the relationship between PSD in time angular frequency and spatial angular frequency can be explained as

$$G_q(\omega) = \frac{1}{u} G_q(\Omega) \quad (43)$$

From the same way,  $G_q(\omega)$  can be explained as

$$G_q(\omega) = G_q(\Omega_0) \frac{\Omega_0^2 u}{\omega^2} \quad (44)$$

Consider equation (44), when  $\omega \rightarrow 0$ ,  $G_q(\omega) \rightarrow \infty$ , therefore, it is necessary to involve  $\omega_0$  which is a lower cut-off time angular frequency [66].  $\omega_0 = 2\pi f_0 = 2\pi u n_0$ . Based on this, equation (44) can be improved to

$$G_q(\omega) = G_q(\Omega_0) \frac{\Omega_0^2 u}{\omega^2 + \omega_0^2} \quad (45)$$

Equation (45) can be noticed as a response from a linear first order system with an excitation from white noise. According to the stochastic vibration theory, the following formula is able to be granted.

$$G_q(\omega) = |H(\omega)|^2 S_w(\omega) \quad (46)$$

where  $H(\omega)$  is the transfer function,  $S_w(\omega)$  is power spectral density of white noise  $W(t)$  and normally, it is equal to 1.

From equation (45) and equation (46),  $H(\omega)$  can be granted as below:

$$H(\omega) = \frac{\Omega_0 \sqrt{G_q(\Omega_0) u}}{\omega_0 + j\omega} \quad (47)$$

According to Laplace Transform, above equation can be rewritten as:

$$H(s) = \frac{\Omega_0 \sqrt{G_q(\Omega_0)} u}{\omega_0 + s} \quad (48)$$

According to this, it is easy to deduct that  $H(s)(\omega_0 + s) = \Omega_0 \sqrt{G_q(\Omega_0)} u$ , furthermore,  $sH(s) + \omega_0 H(s) = \Omega_0 \sqrt{G_q(\Omega_0)} u$  can be worked out, therefore, considering the Laplace Transform,

$$\dot{X}_r(t) + \omega_0 X_r(t) = \Omega_0 \sqrt{G_q(\Omega_0)} u w(t) \quad (49)$$

$\because \omega_0 = 2\pi u n_0, \Omega_0 = 2\pi n_0$ ,  $\therefore$  equation (49) can be rewritten to

$$\dot{X}_r(t) + 2\pi u n_0 X_r(t) = 2\pi n_0 \sqrt{G_q(\Omega_0)} u w(t) \quad (50)$$

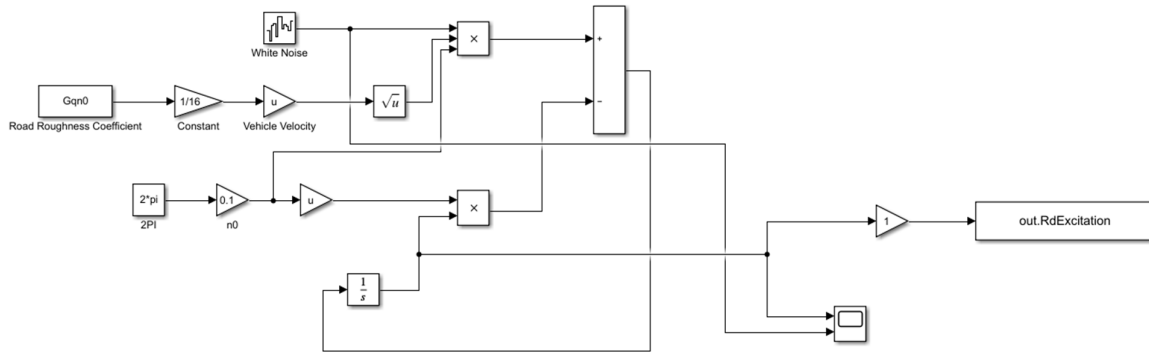
where  $X_r(t)$  is road excitation,  $u$  is vehicle velocity,  $n_0$  is reference of spatial frequency in  $m^{-1}$ . Normally,  $n_0 = 0.1 m^{-1}$ ,  $G_q(\Omega_0)$  is the power spectral density of road roughness when  $\Omega = \Omega_0$ ,  $w(t)$  is Gauss white noise with 0 mean value.

Therefore, by using derivative equation (50), the relationship can be established from white noise  $w(t)$  to road excitation  $X_r(t)$ .

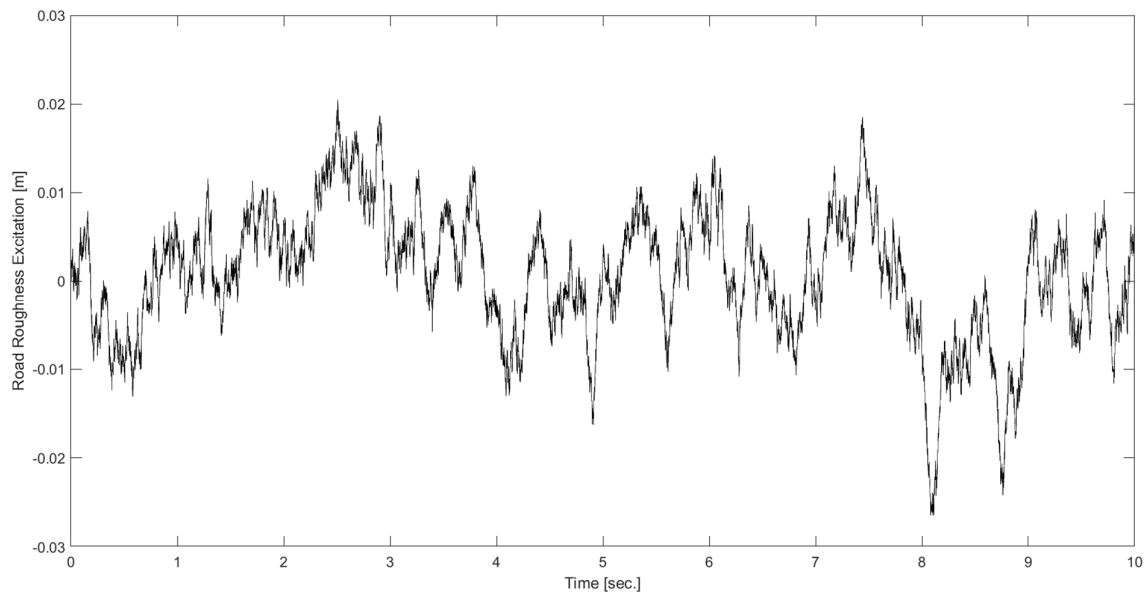
## 5.2 Simulation and analysis

The model is built in Matlab/Simulink, above relationship can be carried out with simulation showed in below Figure 5-1.

Assuming a vehicle drives in 30 km/h on a grade C road, input these parameters into the Simulink block, after that define the white noise power to 10 and sample time to 0.001 second, then a road roughness excitation can be granted and is showed in Figure 5-2.



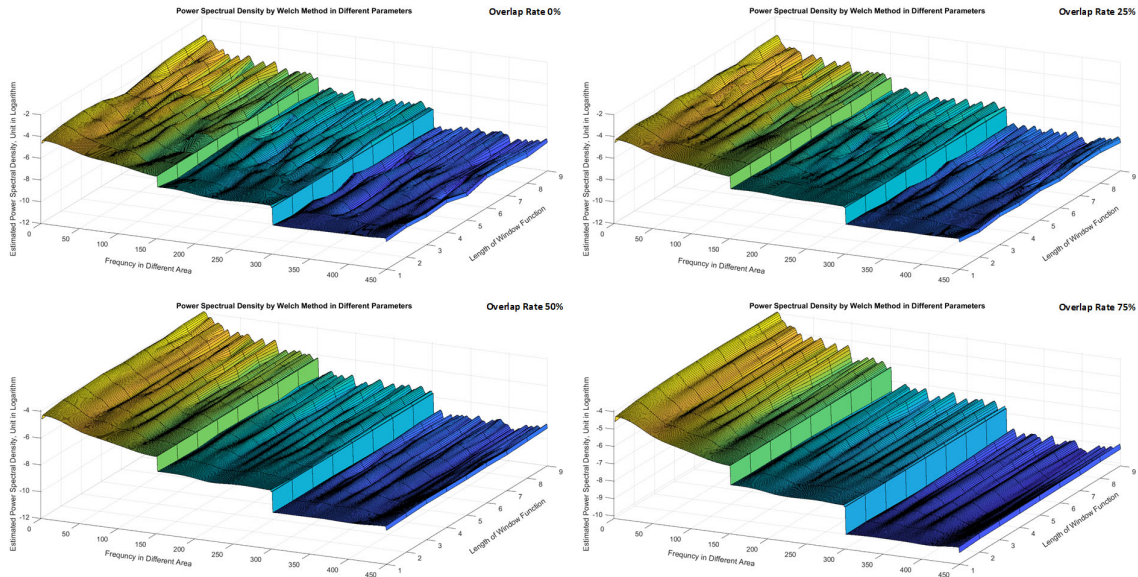
**Figure 5-1 Simulink Block of Road Roughness**



**Figure 5-2 Road Roughness Excitation with vehicle speed of 30 km/h on Grade C Road**

One of the commonly implemented method to illustrate the stationary road vibration is power spectral density which is also known as PSD. When a car goes on a road with a constant velocity, the road pavement can be considered as a stochastic process in space domain which obey to Gaussian probability distribution. In this case, PSD is an ideal method to describe the distribution of roughness energy in space domain.

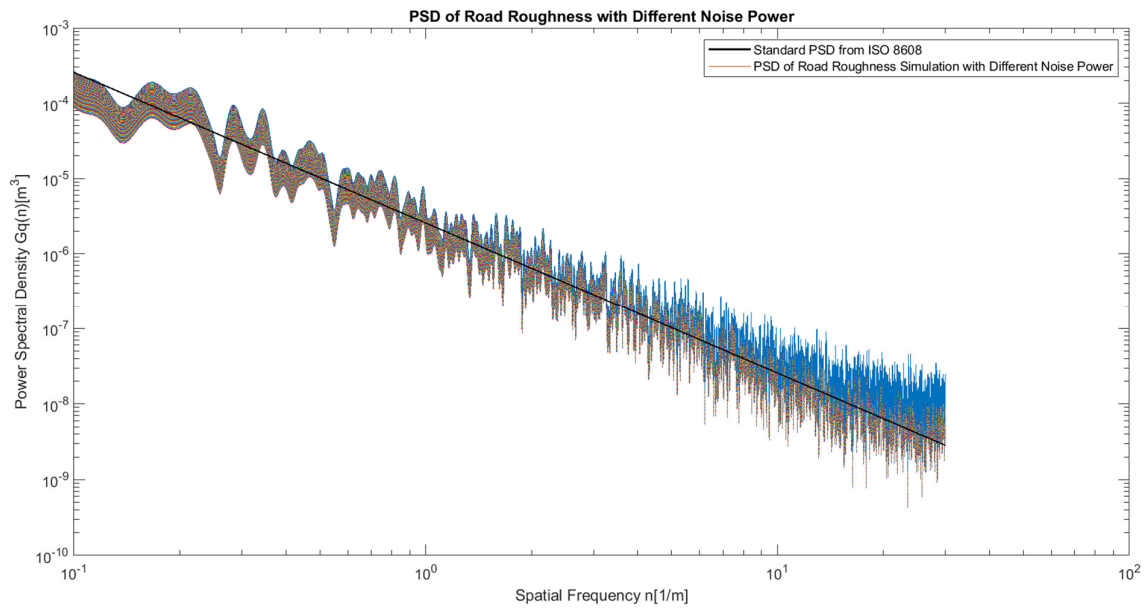
In this study, a parameters-optimized Welch's method is applied in which a relationship between the parameters and the accuracy of the result has been uncovered in Figure 5-3.



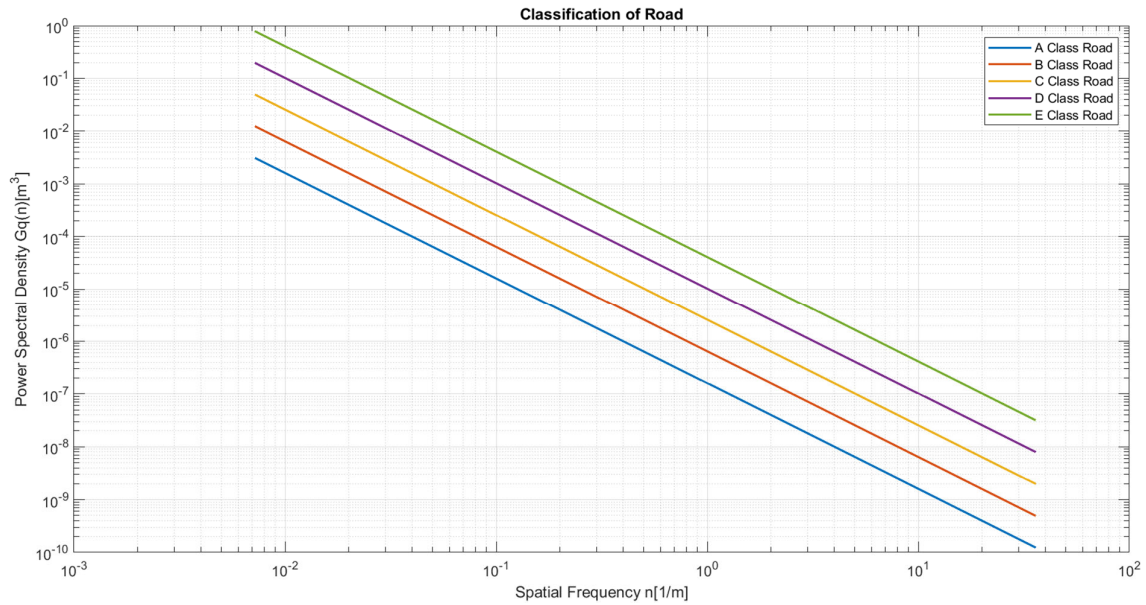
**Figure 5-3 Power Spectral Density with Different Overlap Rate**

Based on these diagrams, the parameters of Welch’s Method can be decided.

Also, a detailed comparison toward standard has been conducted which contributes to the reliability of the simulation. The result has been shown in below diagram.



**Figure 5-4 PSD of Road Roughness with Different Noise Power**



**Figure 5-5 Classification of Roads according ISO8648**

By analyzing and studying the relationship between the spatial frequency and time frequency power spectral density of the road surface, with the help of filtering white noise generation method, combined with Matlab software, a practical road surface generation simulation model is established. By comparing the simulation result with ISO 8608, it has been discovered that some parameters pose a significant influence on the performance of the simulation. Based on this, a study on parameters optimization has been conducted and a group of values have been finalized which enables the simulation to better match to the standard. Therefore, the signal that is generated through above method is reliable and accurate enough to perform as a road excitation input for vehicle control research [67].

## Chapter 6. System Assembly and Simulation

### 6.1 DQN algorithm application

As demonstrated, the top assembly of system which contains a suspension model, a reinforcement learning frame and a signal source of road excitation has been successfully established. A deep Q learning network algorithm is selected to perform the task. Currently, together with the development of a diversity of core algorithms [68] [69] [70] [48] [71], there are lot of different application with deep learning[49] [50] [51] [52] [53] and some other applications are closely related with vehicle dynamic control [72] [73] [74].

Major work in current stage is to test the performance of this system and to find out the key elements that have a significant impact towards the reinforcement learning progress and the final result. To initiate the study, the very first step is to define the pavement, an input signal of Class C road in the vehicle speed of 60 km/h has been chosen to represent a scenario where a car drives at a moderate speed on an intermediate roughness road. It makes sense to start the simulation by using a normal case and late on expanding to other circumstances.

Next, the focus should be landing on defining the parameters of training in reinforcement learning algorithm. Because there are not a lot of current literatures that have revealed the relationship between RL parameters and the performance of suspension system control, it is necessary to make assumptions based on the application of reinforcement learning in other areas.

To be able to respond the requirement, a group of impact factors have been selected to be tested in the system under such an anticipation that parameters like neural network structure, learning rate, greedy rate, sample rate, and size of action space shall pose a significant impact on either the efficiency of learning progress or the performance of the final result. Those factors have been decided to form an **Impact Factors Group 1** and **Impact Factors Group 2** that have been illustrated in below Table 6-1.

Each impact factor will be defined in different values and then an individual learning process will be conducted correspondingly. Later on, by comparing the performance under different cases, the influence from impact factor will emerge. After the best practices of all parameters in **Group 1** and **Group 2** can be defined, there is a possibility to involve more impact factors that are assumed to be less contributive for the test. Maybe some of them can eventually be confirmed to be important towards the performance of the whole system.

*Table 6-1 List of Impact Factors*

<b>Group 1</b>	
Impact Factors	Number of Sample Tested
Neural Network Structure with Single Layer	5
Neural Network Structure with Double Layer	4
Neural Network Structure with Triple Layer	4
<b>Group 2</b>	
Impact Factors	Number of Sample Tested
Learning Rate	6
Greedy Rate	5
Sample Rate	5
Action Space Size	5

### 6.1.1 Impact analysis of different neural network structure

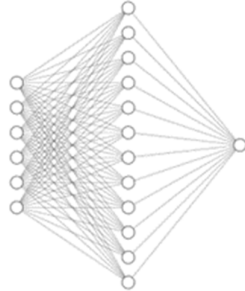
Neural network works as a general function approximator that imitates any input and output relationship and is expected to be one of the most key factors in this study. Intuitively, a more complex input output relation requires a more complex neural network structure, therefore, it is important to find out how networks with different nodes and structures can pose a different influence to the learning process and final result.

In this study, the structure of neural network has been divided into three big categories. They are, one hidden layer network, two hidden layer network and three hidden layer network. In each category, different samples have different structures together with different number of neurons. Detailed information about each category has been summarized from Table 6-2 to Table 6-4, the type of network is decided to be fully connected only at this stage. More different and complicated network type can be considered in application towards further studies.

#### 6.1.1.1 Single layer neural network structure

The study will start with an investigation in a single layer structure network as illustrated in Table 6-2. There are totally five samples with different quantity of nodes in the layer to be tested.

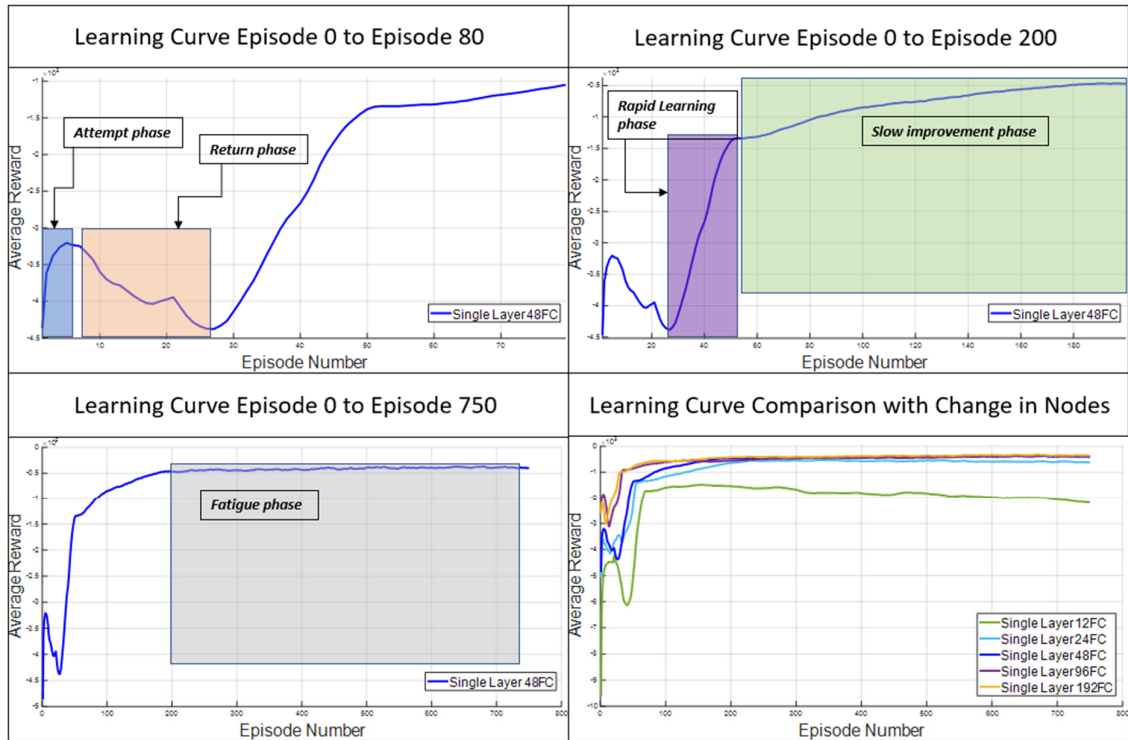
*Table 6-2 Neural Network Structure Category 1*

Category 1: One-layer structure		
Samples	Hidden Layer Nodes	NN-Architecture Schematics
1	12	
2	24	
3	48	
4	96	
5	192	



It can be seen from Figure 6-1 that a typical learning curve contains five different phases. They have been defined as **Attempt phase**, **Return phase**, **Rapid learning phase**, **Slow improvement phase**, and **Fatigue phase**. An explanation towards each phase has been discussed below.

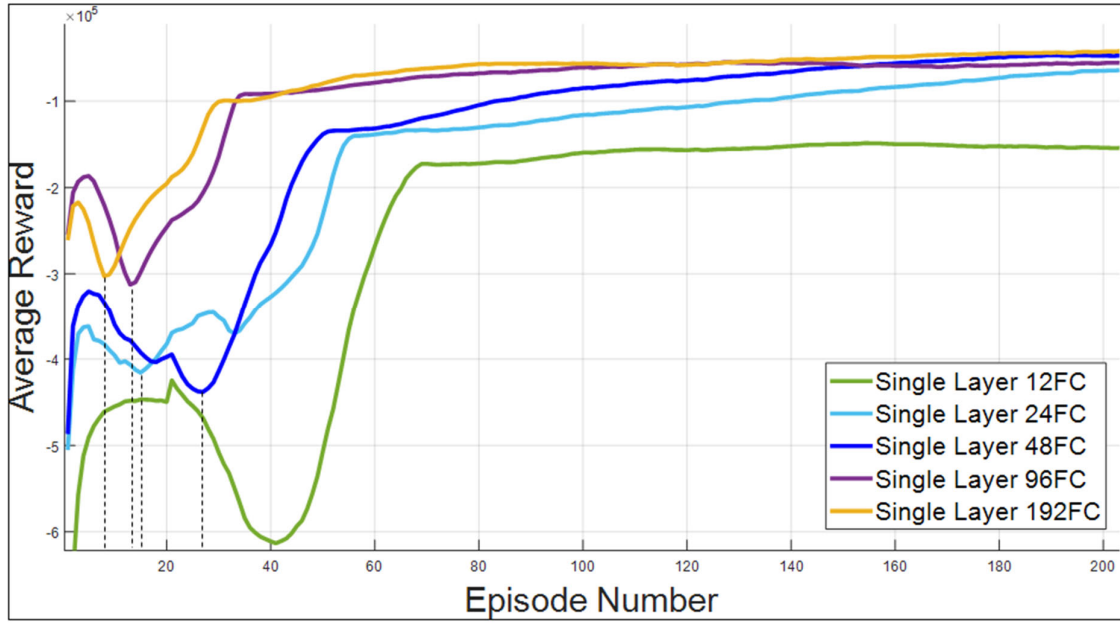
- **Attempt phase** (Approx. from Episode\_0 to Episode\_5): the agent begins to interact with environment and try to establish a method in collecting reward. In this phase, an increasement of reward can be observed however, the system performance is lower than no control.
- **Return phase** (Approx. from Episode\_5 to Episode\_15): the agent is trying to explore the whole action space and filling the Q table. In this phase, it seems to be more exploration than exploitation, therefore a return is demonstrated regarding to the reward collection.
- **Rapid learning phase** (Approx. from Episode\_15 to Episode\_50): the agent has explored sufficiently in action space and found the correct direction in gradient decent. Therefore, in this phase, a significant improvement in reward can be seen.
- **Slow improvement phase** (Approx. from Episode\_50 to Episode\_200): the agent has explored most of the action space and Q table is under fine tuning. The system is still improving in performance however, it takes much longer in learning.
- **Fatigue phase** (Approx. from Episode\_200 to Episode\_750): there is no further improvement can be observed, the system has touched the limit of performance. This phase can be reduced to improve the efficiency of learning process.



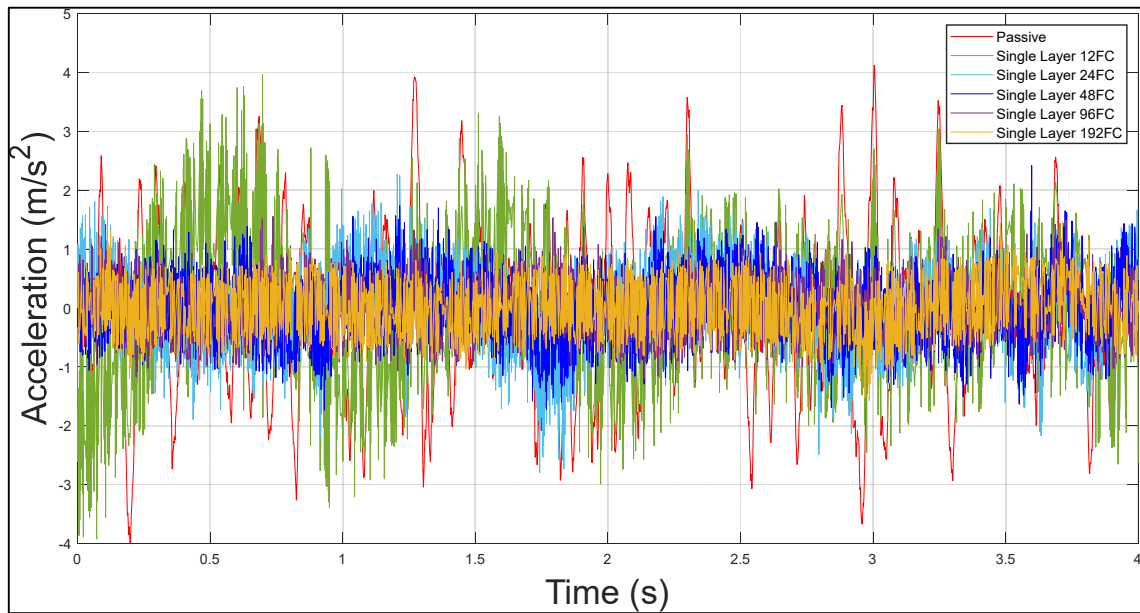
**Figure 6-1 Learning Curve Single Layer Network**

When comparing the learning performance of single layer neural network structure with different nodes from Figure 6-1 and Figure 6-2, there are some conclusions that can be summarized.

1. The more nodes network contains the earlier the system can start rapid learning phase and the shorter the Slow improvement phase lasts.
2. The Rapid learning phase takes around 35 Episodes, there is no big improvement by using more nodes in shortening this period.
3. The network with only 12 nodes has a much lower performance than other structures.
4. The overall improvement from 96 nodes to 192 nodes is not significant, which indicates that the increasement of reward in system has reached a certain level of limit.



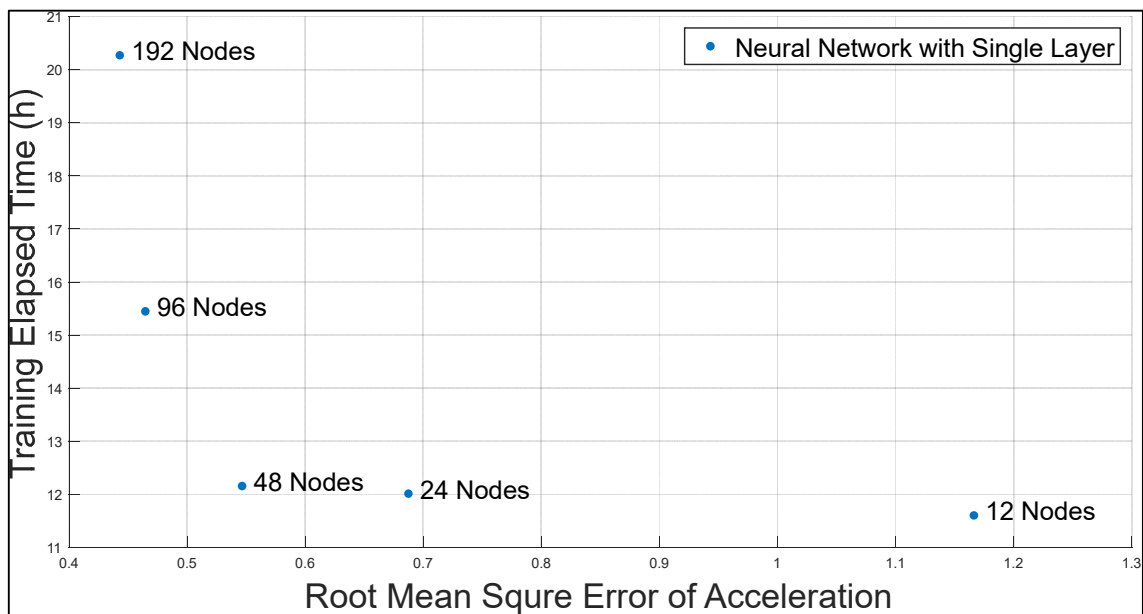
**Figure 6-2 Learning Curve Comparison with Different Nodes Episode 0-200**



**Figure 6-3 Acceleration Comparison for Single Layer Network**

The major job of the task is to reduce the acceleration of the vehicle body to achieve the target of riding comfort. From Figure 6-3, **Error! Reference source not found.** it can be observed that the acceleration has been reduced from maximum  $4m/s^2$  in the condition of no control to  $2m/s^2$  with 24 and 48 nodes fully connected network, and is further improved to around  $1m/s^2$  with 96 and 192 nodes network. The network structure that contains only 12 nodes is not able to demonstrate any improvement.

The diagram in Figure 6-4 provides a picture of how different networks can deliver a diverse performance, however, it does demonstrate the result in such a way that acceleration signal created by each network can be quantified and have a more accurate comparison. Therefore, in this paper, it is decided to use the method of RMSE which is root mean square error to analyze each signal and then illustrate the result of different network together with training elapsed time.



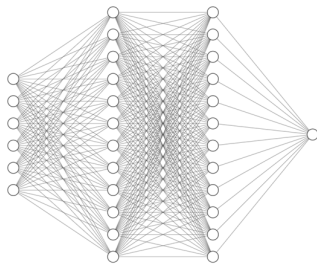
**Figure 6-4 RMSE of Acceleration VS Elapse Time Single Layer Network**

As shown in Figure 6-4, it can be synthesized that considering the network with just one hidden layer, the more nodes a network contains, the better the performance is in RMSE of acceleration, however the longer time it takes to finish the training.

### 6.1.1.2 Double layer neural network structure

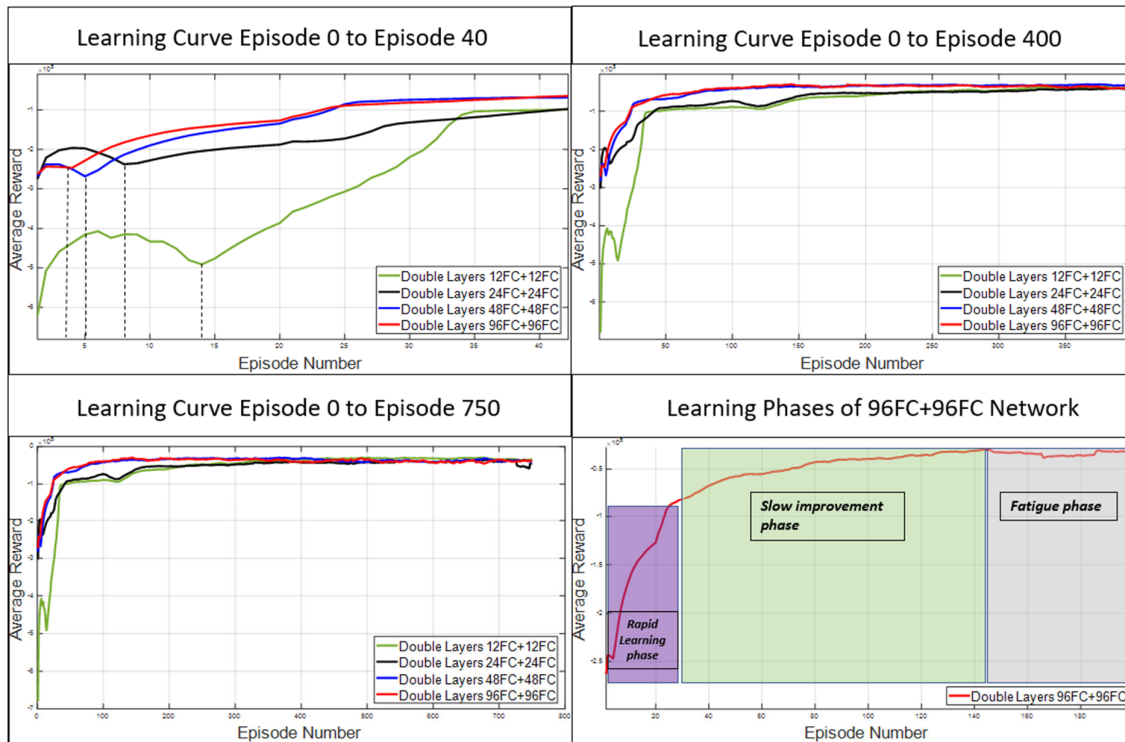
Next, it is necessary to change the network structure from single layer to double layer and apply the same learning process, after that makes an analysis towards the result. The samples that contain different nodes in each layer have been summarized in Table 6-3.

*Table 6-3 Neural Network Structure Category 2*

<b>Category 2: Two-layer structure</b>			NN-Architecture Schematics
Samples	Hidden Layer Nodes		
	1 <sup>st</sup> Layer	2 <sup>nd</sup> Layer	
1	12	12	
2	24	24	
3	48	48	
4	96	96	

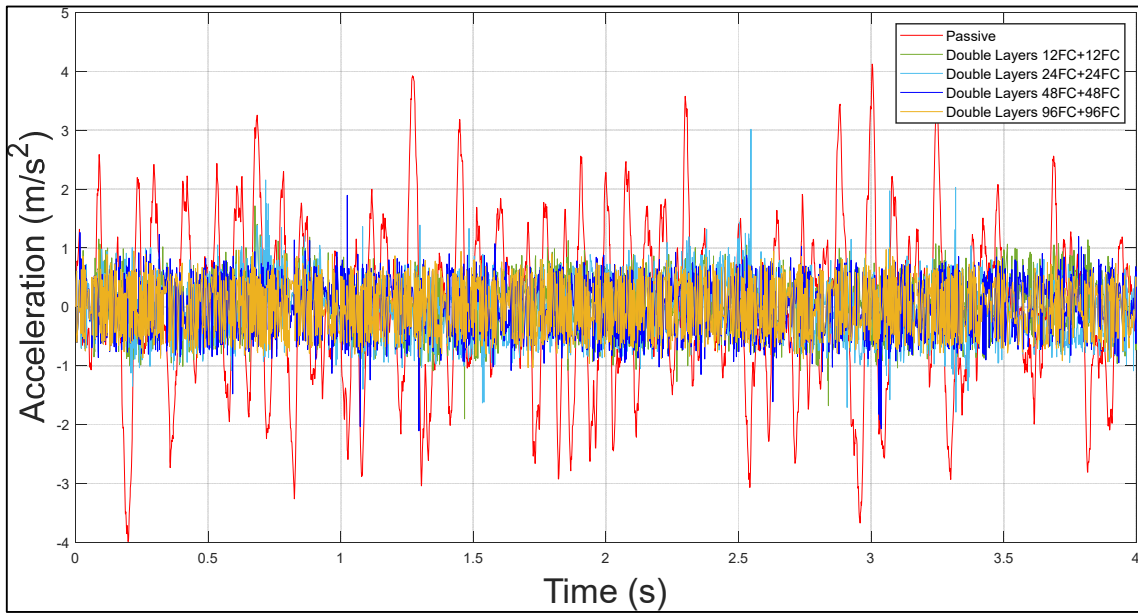
When comparing the learning performance of double layer neural network structure with different nodes from Figure 6-5, there are also some regulations that can be synthesized.

1. Just like single layer network, the more nodes network contains the earlier the system can start rapid learning phase and the shorter the Slow improvement phase lasts.
2. The overall improvement from 48+48 nodes network to 96+96 nodes network is not significant, which indicates that the increasement of reward in system has reached a certain level of limit.
3. Unlike the other networks, learning phase of a 96+96 network is unique. It has not a significant return phase and the rapid learning phase starts very early. Thus, the learning phase can almost be divided into just three major phases.
4. The fatigue phase of network with 48+48 nodes and 96+96 nodes start around Episode\_150 which indicates a possibility of a potential reduction in current 750 episodes in total.

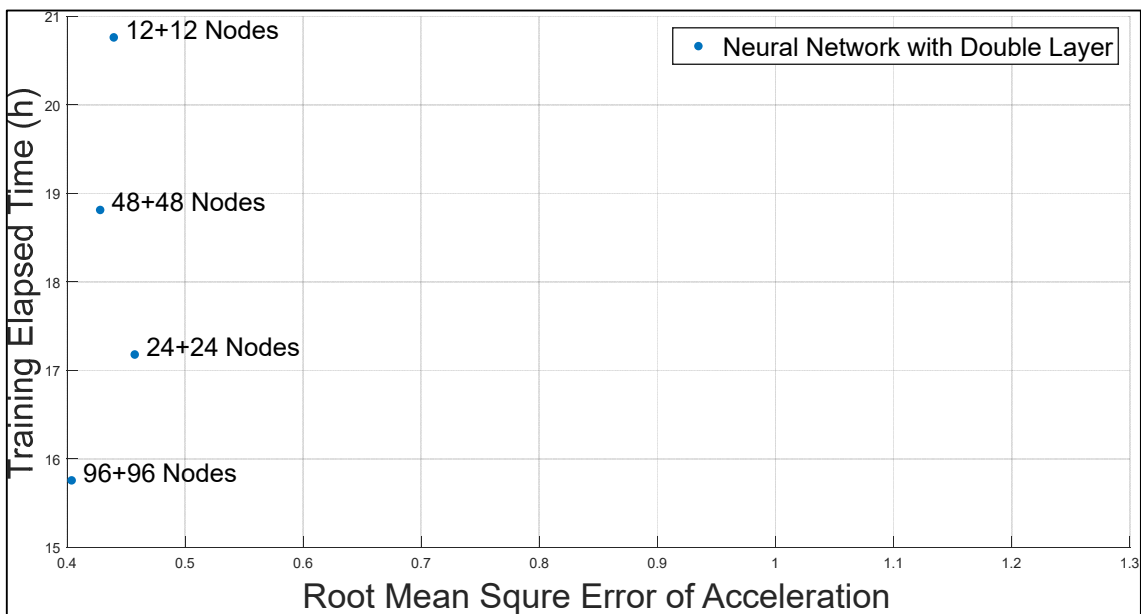


**Figure 6-5 Learning Curve Double Layer Network**

From Figure 6-6 it can be observed that the acceleration has been reduced significantly compared to the condition of no control. However only the network with 96+96 nodes illustrated the result that acceleration has been controlled basically under  $1m/s^2$ . The acceleration values from other networks still sometimes go up to  $2m/s^2$  or even  $3m/s^2$ . Just like single layer network, the result from double layer network also suggests a propensity that a network structure with more nodes can generally provide a more stable and reliable performance.



**Figure 6-6 Acceleration Comparison for Double Layer Network**



**Figure 6-7 RMSE of Acceleration VS Elapse Time Double Layer Network**

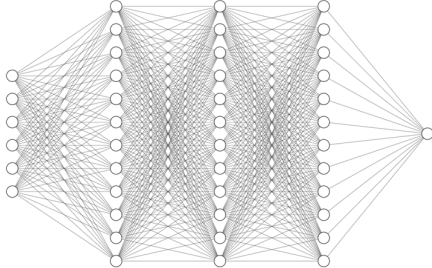
Figure 6-7 has presented some very different facts compare to Figure 6-4. First, the difference in RMSE of acceleration across networks with different nodes is not as big as that from single layer network. However, the network with maximum nodes is still taking a lead in performance. Second, the training elapsed time shows a completely opposite tendency comparing with single network. The more nodes the network contains, the less time the system need to finish training.

This result is inspiring, because it means that the layer number of the neural network structure actually poses an impact on the training efficiency. Even with the same node number, a double layer network is more superior than a single layer network considering the time that network training requests. An assumption can be raised that a multi-layer network takes less time to find out a solution considering the complexity of the current environment. Combining with the conclusions from the learning curve study, it seems obvious that two layers network definitely has more potential in dealing with the task.

### 6.1.1.3 Triple layer neural network structure

Additional layer has been added to the network to increase the complexity of the universal function approximator. Based on the improvement performance from one-layer network to two-layers network, it is interesting to see the result from a three layers network. The detailed structure and nodes arrangement are listed in Table 6-4.

*Table 6-4 Neural Network Structure Category 3*

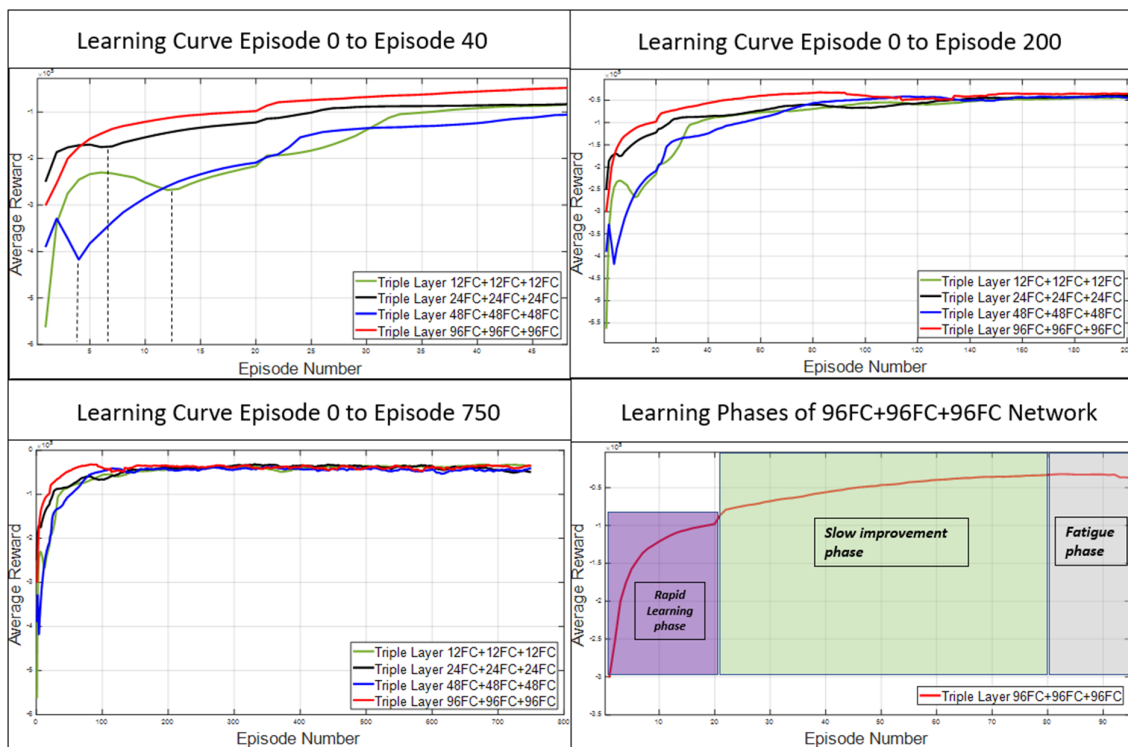
<b>Category 3: Three-layer structure</b>				
Samples	Hidden Layer Nodes			NN-Architecture Schematics
	1 <sup>st</sup> Layer	2 <sup>nd</sup> Layer	3 <sup>rd</sup> Layer	
1	12	12	12	
2	24	24	24	
3	48	48	48	
4	96	96	96	

From Figure 6-8, similar phenomenon can be observed compared to those from double layer networks. A summary of conclusion is listed below.

1. Just like single layer and double layer network, the more nodes network contains the earlier the system can start rapid learning phase and the shorter the Slow improvement phase lasts.

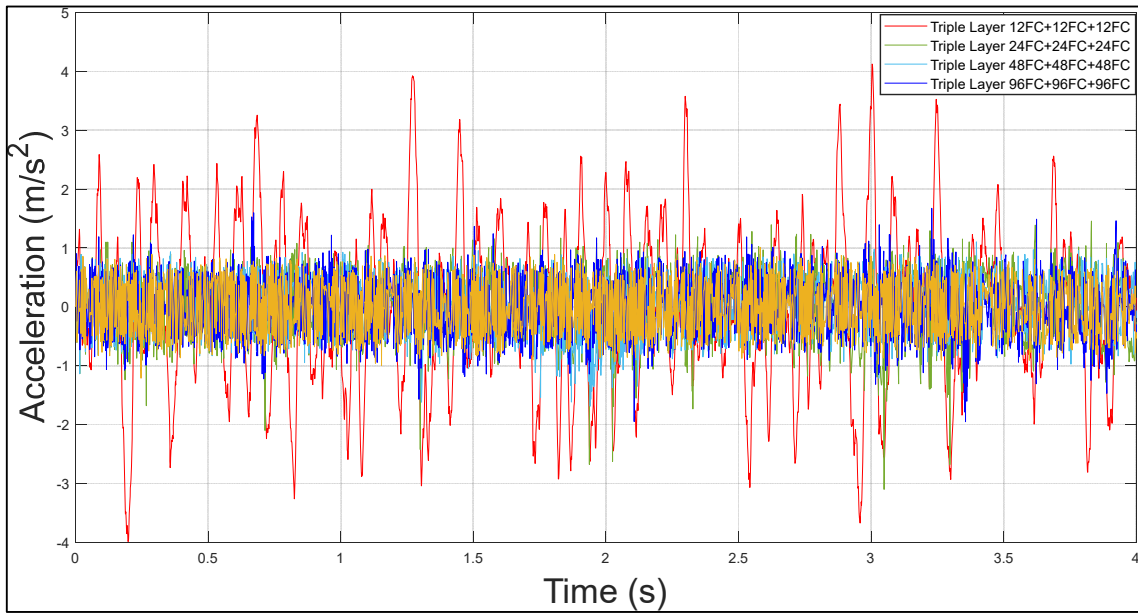


- The overall difference of all four learning curves is not significant, which indicates that the three-layer network system is not very sensitive towards the number of neurons regarding to the learning process.
- Just like a 96+96 network, the network that contains 96+96+96 nodes has not a return phase at all. The curve of rapid learning phase is very smooth and rapid learning starts almost from Episode\_1, which confirms a very efficient learning progress.
- The fatigue phase of network with 96+96+96 nodes start even earlier comparing with the best one from a double layer network. The fatigue actually begins around Episode\_80.

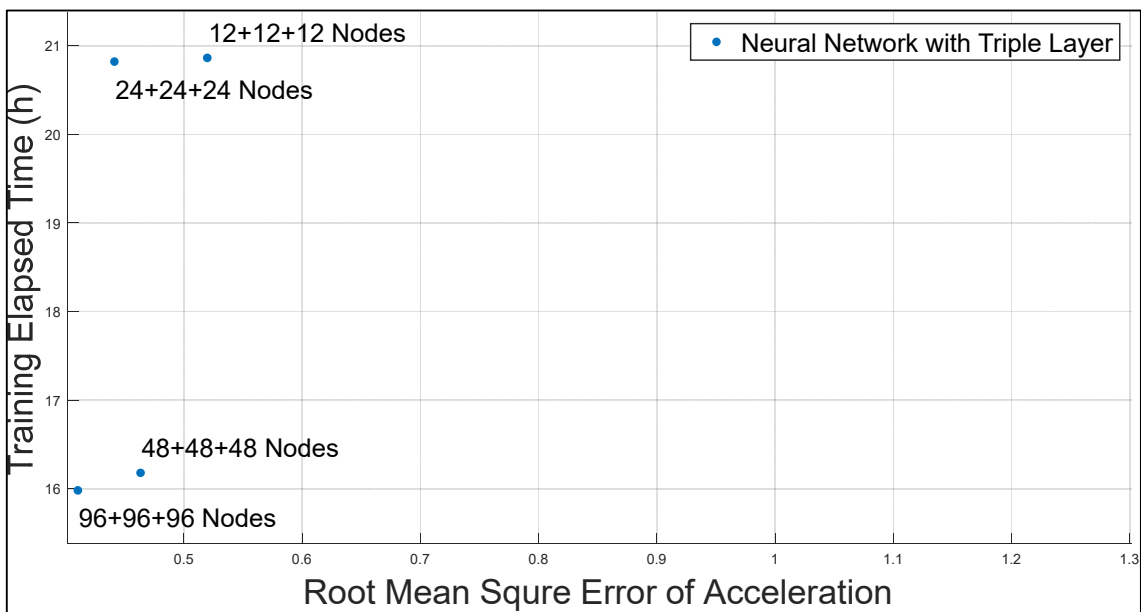


**Figure 6-8 Learning Curve Triple Layer Network**

As summarized from above statement, the learning curve of a triple network with 96+96+96 nodes is very smooth that illustrates a certain level of advantage in learning efficiency.



**Figure 6-9 Acceleration Comparison for Triple Layer Network**

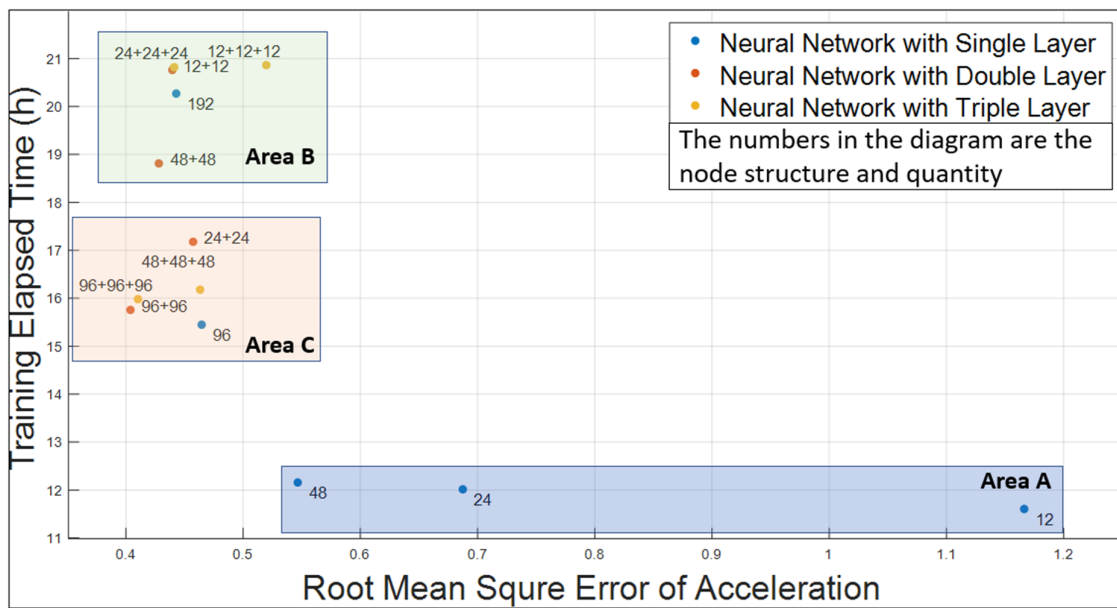


**Figure 6-10 RMSE of Acceleration VS Elapse Time Triple Layer Network**

Similar to the result of double layer network, it can be observed from Figure 6-9 that the accelerations of triple layer agent have been reduced noticeably compared with passive control. However only the network with 96+96+96 nodes illustrated the result that acceleration has been reduced to around  $1m/s^2$ . The acceleration values from other networks still sometimes go up to  $2m/s^2$  or even  $3m/s^2$  which is similar to the cases from those double layer networks.

This seems to tell us that keeping adding an extra layer of nodes is not able to provide a continuous improvement on the acceleration, the major advantage of a multi-layer network is that it actually created a better learning curve to enhance the learning efficiency.

As can be summarized from Figure 6-10, the difference in RMSE of acceleration across triple layer networks with different nodes is not significant which is similar to the result from a double layer system. Training elapsed time also demonstrates a similar tendency that proves more nodes the network contains, the less time the system needs to finish training.



**Figure 6-11 RMSE of Acceleration VS Elapse Time with All Three Network Structure**

Trying to summarize and compare the performances across all three different types of networks, Figure 6-11 has actually provided a good vision of result. Single layer networks with nodes quantity under 50 in Area A took less time in training the agent, however the improvement of acceleration was far from satisfactory. Agents that are located in Area B can provide a relatively better performance compared with those from Area A, but took a long time in training the algorithm. The networks in Area C have reached a good balance in considering both acceleration and training elapsed time, therefore, have a higher priority to be implemented in

the further research. Also, it can be observed that the overall performance of a triple layer system has not gone beyond that from a double layer system.

### **6.1.2 Impact analysis of hyper-parameters of reinforcement learning**

The following study will be focused on Learning Rate, Greedy Rate, Sample Rate and Action Space Size. These are also important hyper-parameters that pose a significant impact on the converge of the learning, process stability and final performance.

#### **6.1.2.1 Learning Rate Analysis**

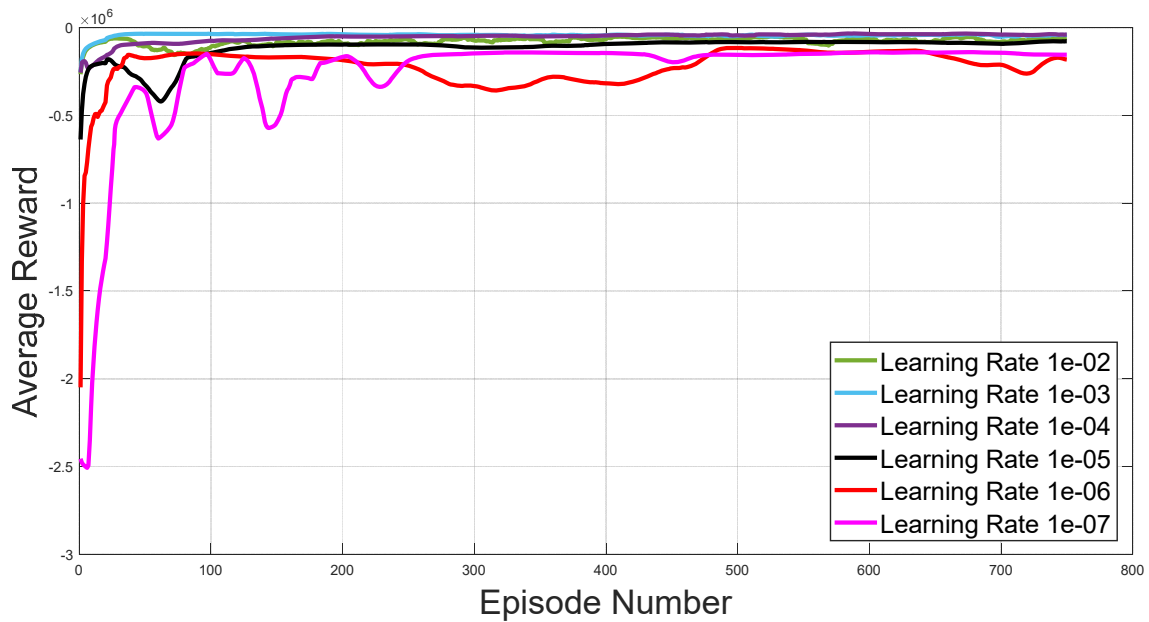
Stochastic gradient descent algorithm is an algorithm of optimization that uses batch examples from training database to estimate the error gradient for the current observation from the environment. It utilizes back propagation of errors to improve and update the weights of the model, which is used to train deep neural networks.

In the training process, the quantity of the weights that has been updated in the training process is defined as “Learning Rate”. In reinforcement learning, the learning rate is a configurable hyperparameter which is normally a scaler in the range of 0 to 1.

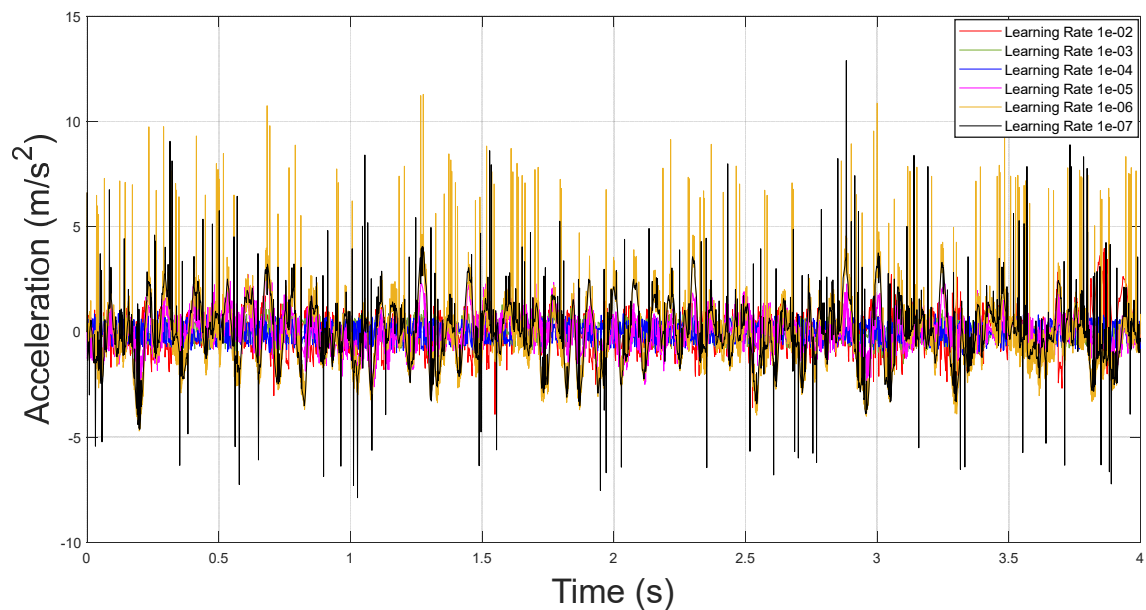
Intuitively, smaller learning rate will end up with more training episodes given a smaller change is applied to the weight of the network in each step, whereas bigger learning rate pose a bigger change in the weight and requires less episode.

It is very important and at the same time very difficult to define a proper learning rate. A rate that is too large may cause the neural network to converge very quickly however, it may land on a second optimal position. However, a very small learning rate may lead the learning get stuck in some place and make the whole process go divergent.

From Figure 6-12 it can be observed that all learning curves are able to be convergent, however the small the learning rate is the more oscillating the curve contains. When the learning is finished, the average reward the collected by large learning rate agent is much less than the small learning rate agent which suggests they might be getting stuck in somewhere in control.

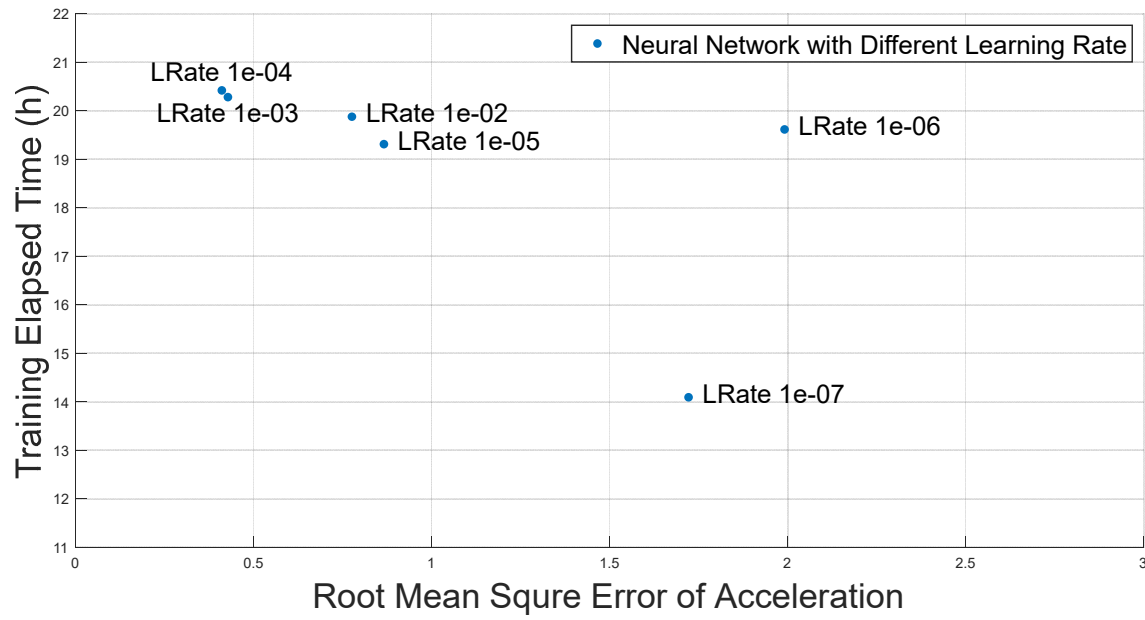


**Figure 6-12 Learning Curve with Different Learning Rate**



**Figure 6-13 Acceleration Comparison with Different Learning Rate**

By looking into Figure 6-13, it actually tells us that learning rate 1e-06 and 1e-07 can hardly provide any improvement in acceleration while learning rate 1e-02 and 1e-05 are demonstrating an observable improvement. Learning rate 1e-03 and 1e-04 seems to provide a very satisfactory result, a more detailed and quantitative comparison is showed in the following graph.



**Figure 6-14 RMSE of Acceleration VS Elapse Time with Different Learning Rate**

It is very clear that learning rate 1e-06 and 1e-07 are far from an accepting result by looking into Figure 6-14 even if learning rate 1e-07 shows a surprisingly low training time. Learning rate 1-02 and 1e-05 are both able to provide an improvement in RMSE of acceleration however, 1e-03 and 1e-04 are the best agent among all test samples.

Through the study of learning rate in the simulation, it is revealed the importance of this hyper parameter, it is directly related to the final performance of the control system, therefore it is worthwhile to have a calibration in the tuning process especially when the performance is going divergent.

### 6.1.2.2 Greedy Rate Analysis

To better understand the greedy rate in reinforcement learning, it is necessary to first know about the concept of exploration and exploitation. In RL, the agent is trying to interact with the environment however is not knowing anything about the environment. A model free algorithm is actually relying on trial and error. During the trials, the size of action space decides the number of actions that an agent can apply. Sometimes, the actions are not actually taken when interacting with the environment, however the algorithms can still update and estimate the

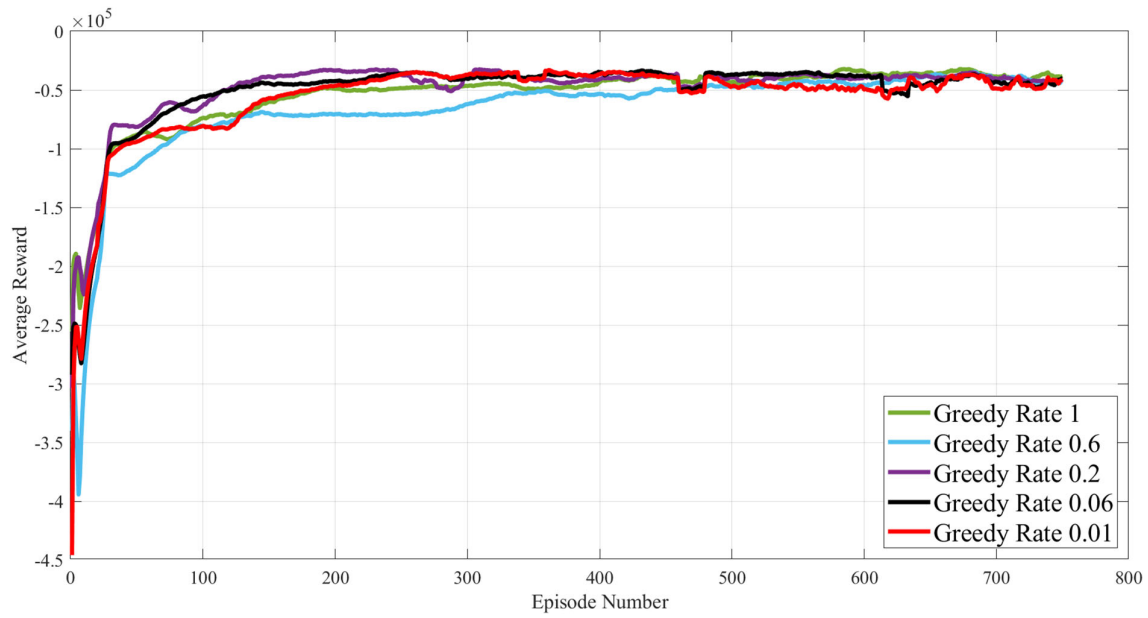
values from those actions. Hence, the agent is able to explore and benefit from actions that actually had never happened in the learning phase.

In the beginning of the training, the agent initially has very limited information from the environment, in that case, an agent can choose to explore the environment by selecting a stochastic action which brings back an unpredictable outcome so that it gets to know more about the environment. Alternatively, it can choose to exploit and pick up an action with the highest value based on its previous experience in interacting with an environment in the purpose of getting an optimal reward.

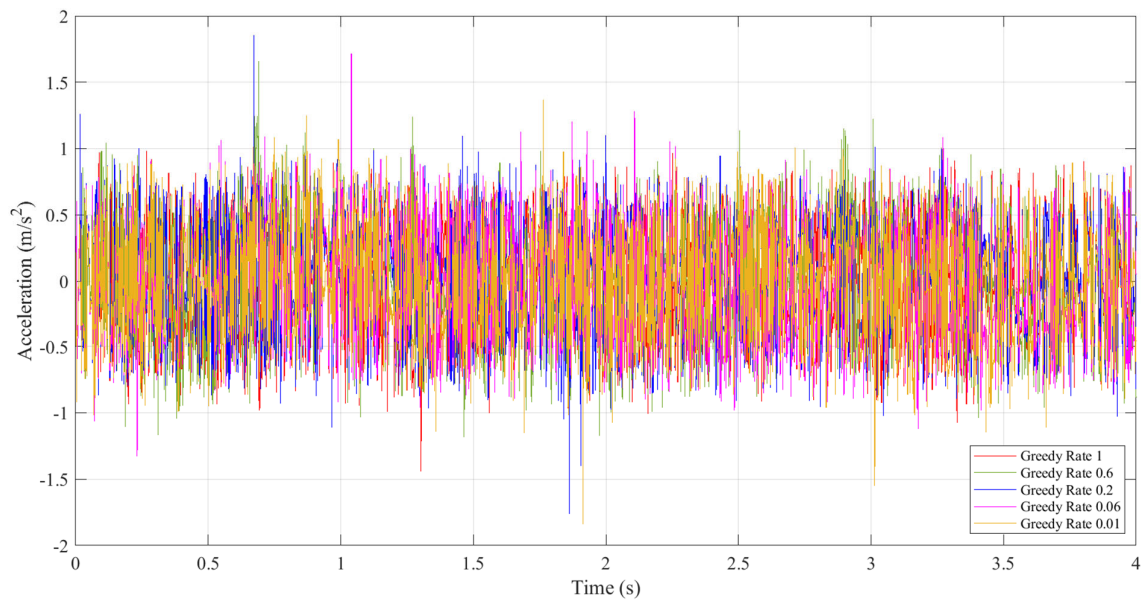
The movement of exploring the result via a random action versus exploiting the best solution in action space regarding to the current knowledge is called the exploration and exploitation trade off. On one hand, an agent can potentially collect a higher reward through exploration application therefore benefit the final performance in the long run. On the other hand, if an agent starts exploiting, it immediately gets more reward, however, might get trapped in a sub-optimal behavior.

When the training has started, the agent has little knowledge of the outcomes of the possible actions, in this stage, sufficient initial exploration is necessary. As time goes by and more experiences have been collected in the replay buffer, exploitation shall take the lead and start improving the process. However, it is also very dangerous if the algorithm is only doing exploiting, a greedy agent can get easily stuck in a sub-optimal state if the environment is changed with the time pass. In this consideration, a balance between exploration and exploitation is vital and should be carefully considered in the reinforcement learning.

In Q Learning, Epsilon or  $\epsilon$  is used to represent the greedy rate, and by applying that rate, the agent is able to utilize both exploitations to improve performance from its prior knowledge and exploration to find potential high reward areas.



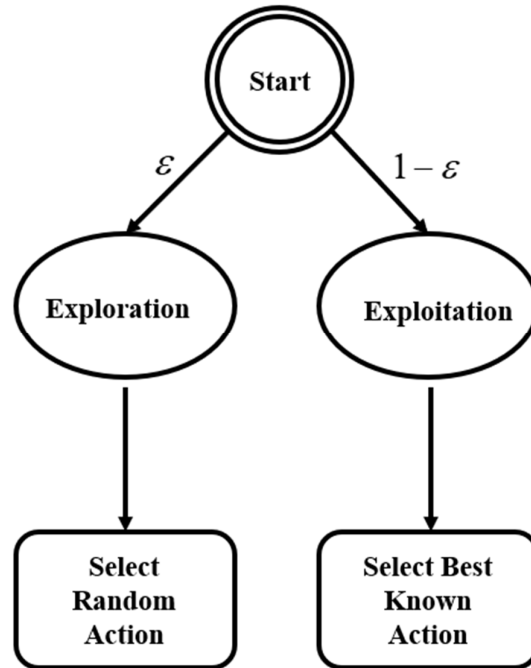
**Figure 6-15 Learning Curve with Different Greedy Rate**



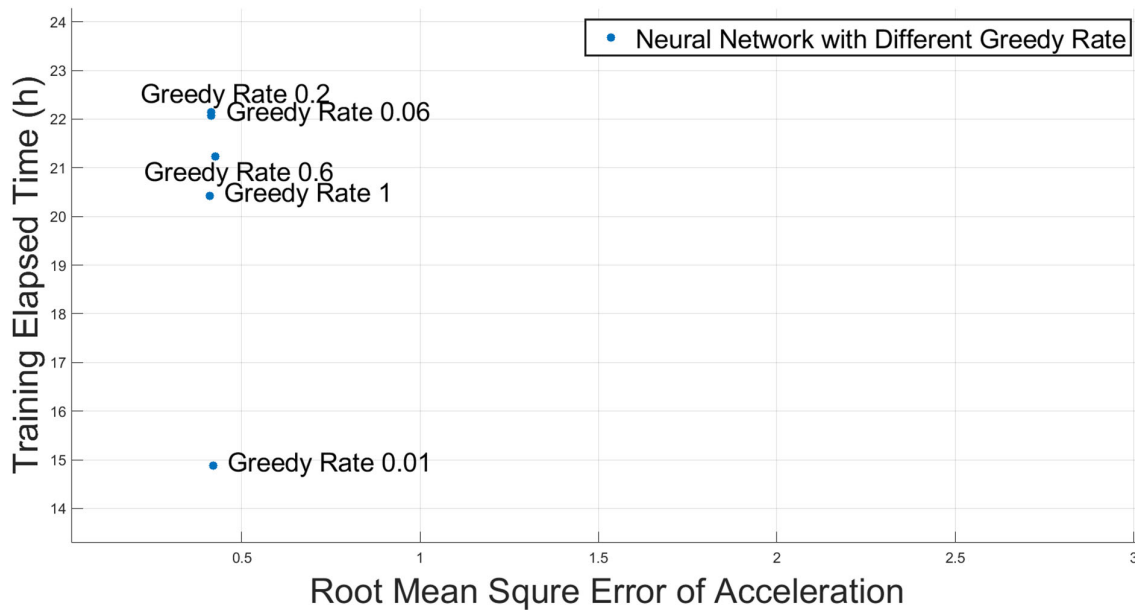
**Figure 6-16 Acceleration Comparison with Different Greedy Rate**

In this study, Epsilon is defined as a probability threshold to either select an action stochastically or select the action with a maximize state-action value. The bigger the Epsilon is, the more likely the agent is conducting a random movement. Below graph illustrates the relationship between Epsilon and exploitation-exploration trade off.





*Figure 6-17 Epsilon and Exploration-Exploitation*



*Figure 6-18 RMSE of Acceleration VS Elapse Time with Different Greedy Rate*

In actual application, there is another parameter which is called  $\epsilon$ \_Decay that is also get involved. This decay is to change the greedy rate in the learning process to imply a gradually reduced exploration and a gradually increased exploitation. This is very in line with an actual

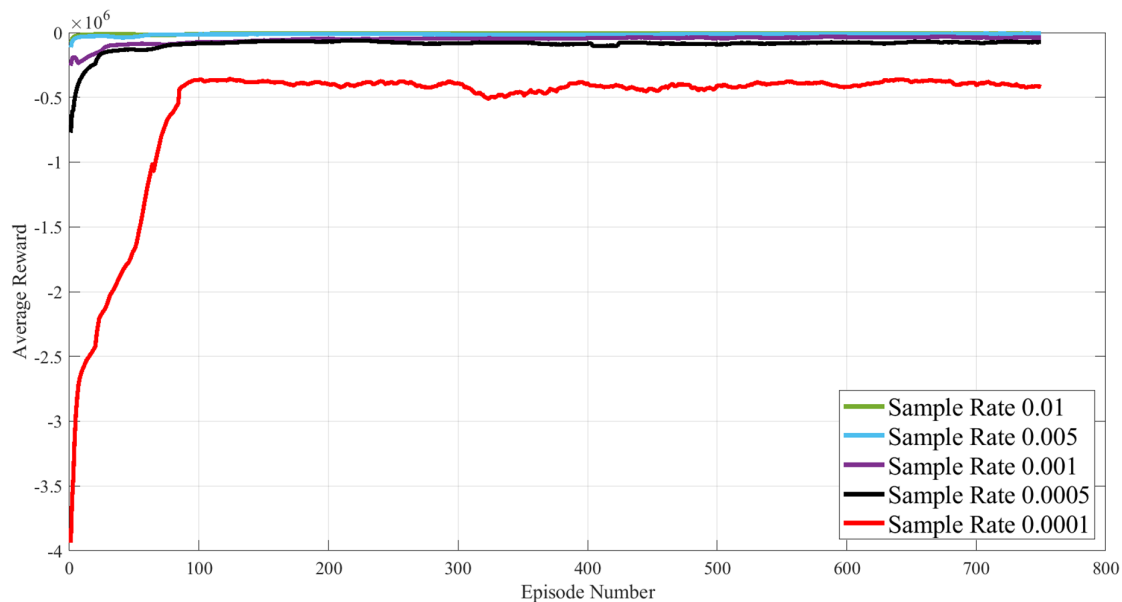
human learning procedure, that is as the time goes by, more experiences are accumulated and therefore, exploitation can bring back more benefit comparing with a random approach. In the algorithm, in the end of each training time step, if Epsilon is greater than a preset threshold  $Epsilon\_Min$ , then it is updated using the following formula.

$$Epsilon = Epsilon \times (1 - Epsilon\_Decay) \quad (51)$$

It can be summarized from Figure 6-15, Figure 6-16, and Figure 6-18, in the current setup, Epsilon does not impose a significant impact into the learning process and control process due to the existence of the  $Epsilon\_Decay$ .

### 6.1.2.3 Sample Rate Analysis

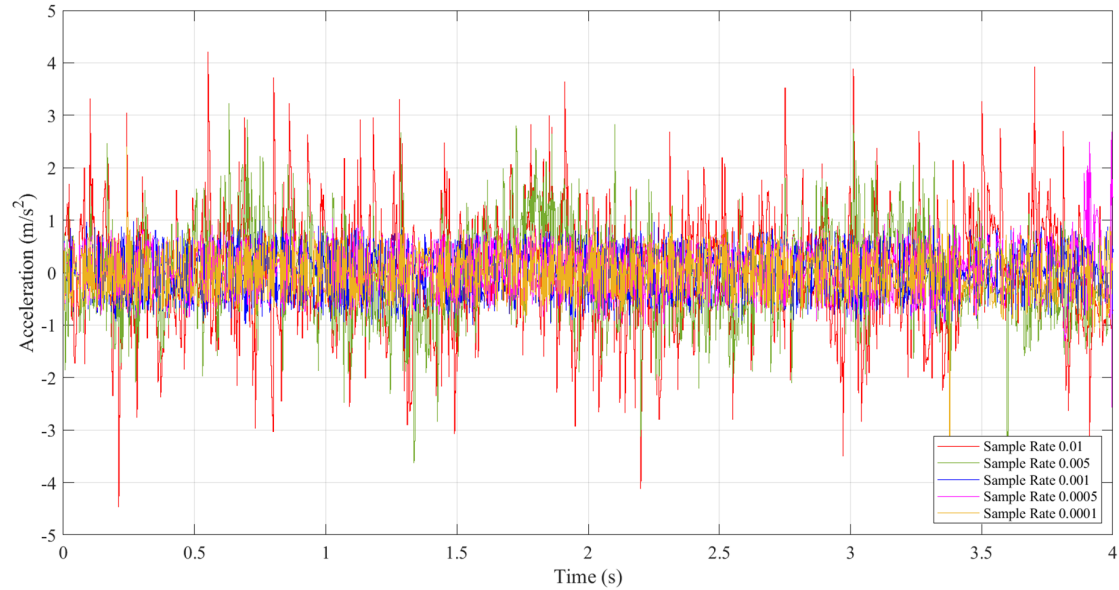
Sample rate in this study is defined as the number of forces that has been conducted from the suspension actuator in one second. Intuitively, the more forces that can be generated in a time unit, the better the performance is. However, the training time is also need to be considered.



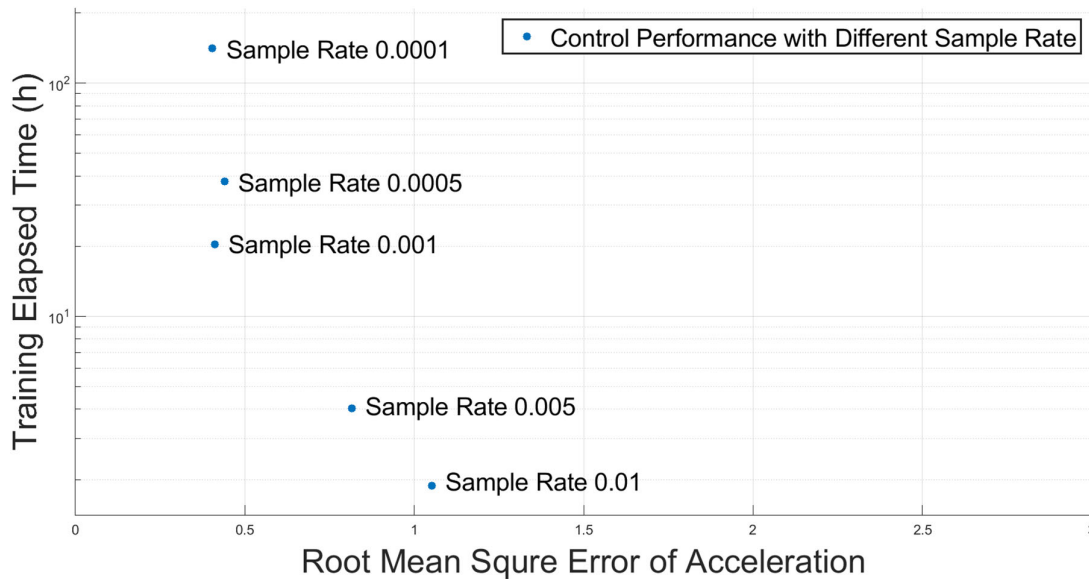
**Figure 6-19 Learning Curve with Different Sample Rate**

From Figure 6-19, it is demonstrated that the smaller the sample rate is the less the rewards can be collected by the agent. This is because that when the sample rate reduces, the number of actions is actually increasing, more actuator actions are conducted in a unit time, therefore,

creating more opportunities to receive rewards (in this design, they are normally negative scalars). So, when the sample rate is changing, by only looking into the learning curve may not be able to give us the correct information about how good the agent is performing.



**Figure 6-20 Acceleration Comparison with Different Sample Rate**



**Figure 6-21 RMSE of Acceleration VS Elapse Time with Different Sample Rate**

By looking into the Figure 6-20 and Figure 6-21, it can be summarized that the lower the sample rate is the less training time the agent takes. When the sample rate is reduced to 0.0001s, the training time is dramatically increased to more than 100 hours which is not practical. Plus,

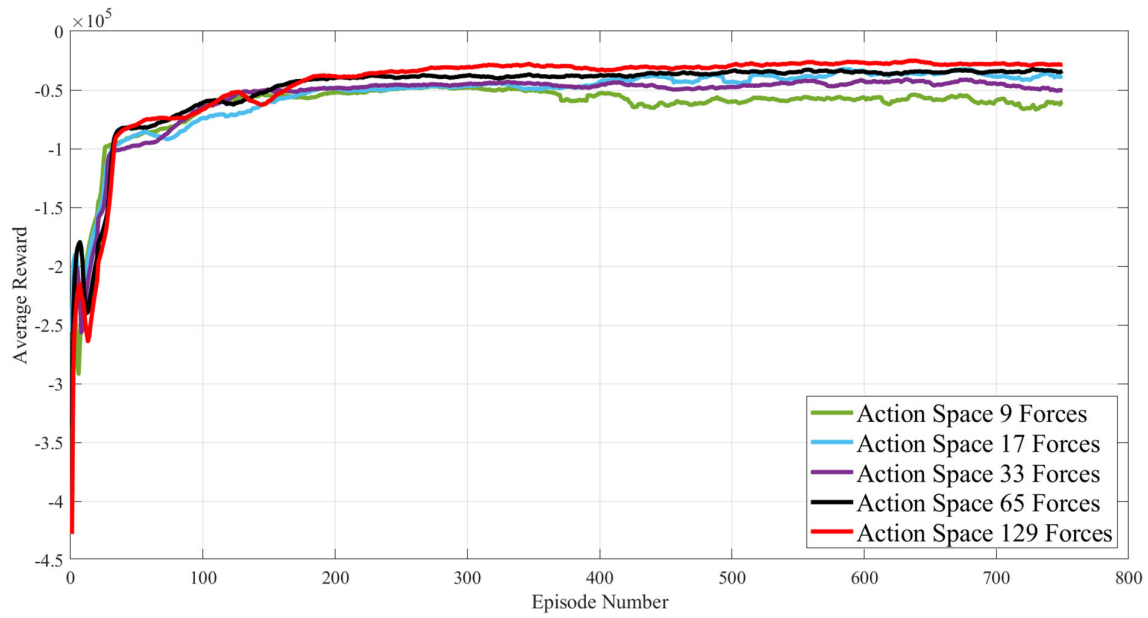
0.0001 s sample rate means that the actuator will need to produce 10,000 forces per second, it is very difficult to find such an actuator in the real world, therefore the study is only for theory analysis and boundary check. The good news is that the performance of the acceleration has not been optimized with the decrease of sample rate, which illustrated that the it is not a linear change. When the sample rate reaches 0.001 s, the performance of acceleration is also reaching a peak. The actuator with a 0.001 s responding time does exist in the world therefore provides an opportunity to transfer the theory study to an actual application in physical world.

#### **6.1.2.4 Action Space Size Analysis**

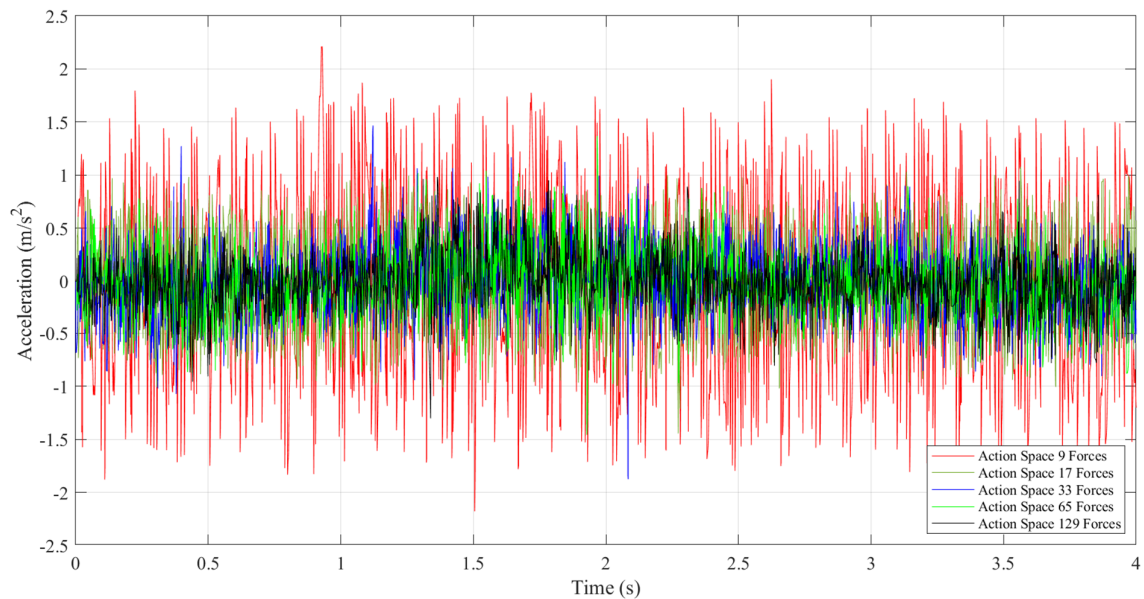
Due to the fact that DQN is not able to work with a continuous action space, therefore a discrete action space is needed to be compatible with the algorithm. Action space size in this research is defined as the number of forces that can be controlled by the actuator. Intuitively, the more different forces the actuator can control, the better the control result is. However, it is still worthwhile to check how the change of the action space size can impact the learning process and final acceleration performance.

It can be easily summarized from Figure 6-22 that when the number of the action force increases, the average reward is that is collected by the agent also improves. The trajectory of each learning curve is quite similar to each other.

Through Figure 6-23, it can be seen that the acceleration has already been improved from approximately 2 m/s to 1.5 m/s with an action space that contains only 7 forces. When the forces are increased into 17 forces, the acceleration is improved to 1 m/s. And a continuous improvement can be observed with the expand of the action space.



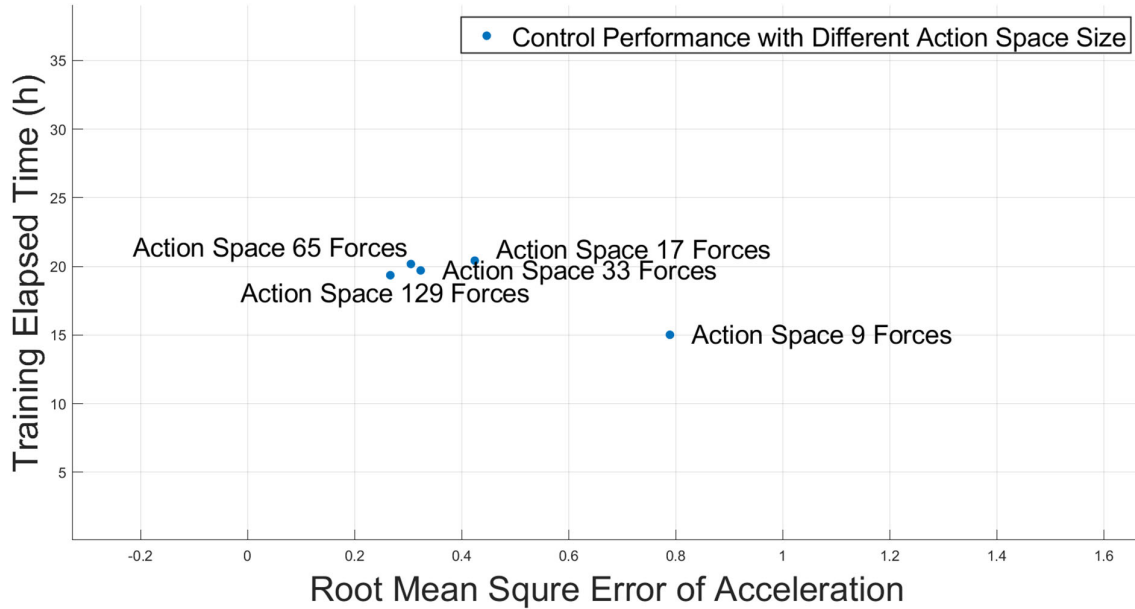
**Figure 6-22 Learning Curve with Different Action Space Size**



**Figure 6-23 Acceleration Comparison with Different Action Space**

From Figure 6-24, a big improvement in RMSE of acceleration can be seen when the action space is increased from 9 forces to 17 forces. And when the space is expanded to 33 forces, there is also a noticeable change in RMSE. However, then the action space is continuously increased to 65 and 129 forces, the improvement is able to be observed however is not that significant. The interesting thing is that size of the action space seems not to be impacting the

training time. The elapsed time from each agent doesn't provide a significant difference which reveals a potential improvement direction by using an algorithm which contains a continuous action space.



*Figure 6-24 RMSE of Acceleration VS Elapse Time with Different Action Space Size*

## 6.2 DDPG Algorithm Application

### 6.2.1 Introduction

Deep Deterministic Policy Gradient (DDPG) is an algorithm that learns a Q-function and a policy at the same time. The policy is learned from Q function, and Q function is learned from Bellman equation and off policy data.

DDPG is basically an upgrade from Deep Q Net (DQN), they both have the same motivation in Q learning. If the optimal action-value function  $Q^*(s, a)$  is worked out, then the optimal action  $a^*(s)$  can be solved in any given state.

$$a^*(s) = \arg \max_a Q^*(s, a) \quad (52)$$

The biggest improvement of DDPG comparing with DQN is that it can cope with a continuous action space [69] which is interacting with the environment. This is related to how the algorithm

is computing the max over actions. The max works out perfectly when the action space is consisted of a finite number of discrete actions because each Q value from an action can be calculated and compared individually. However, when the action space become continuous, it is impossible to work out the max Q in the same way. A simple approach is to discretize the action space. But this may lead to a lot of problems, one of those is called the curse of dimensionality which means that the number of actions expand exponentially with the number of degrees of freedom increasing. Such huge action space is not only hard to explore, but also blocks the way of solving optimization problem. Therefore, using a normal optimization algorithm would make it impossible from calculating  $\max_a Q^*(s, a)$  considering that fact that it needs to be run in every step when agent is taking an action to interact with the environment.

Regarding to the fact that action space is continuous, so the function  $Q^*(s, a)$  is considered to be differentiable which provides a chance to utilize a gradient-based learning rule for a policy  $\mu(s)$ . In this case, instead of applying a huge workload to work out  $\max_a Q(s, a)$ , it is more practical to create an estimation on it which can be described with a new equation  $\max_a Q(s, a) \approx Q(s, \mu(s|\theta^\mu))$ . This approach is based on a DPG algorithm [48].

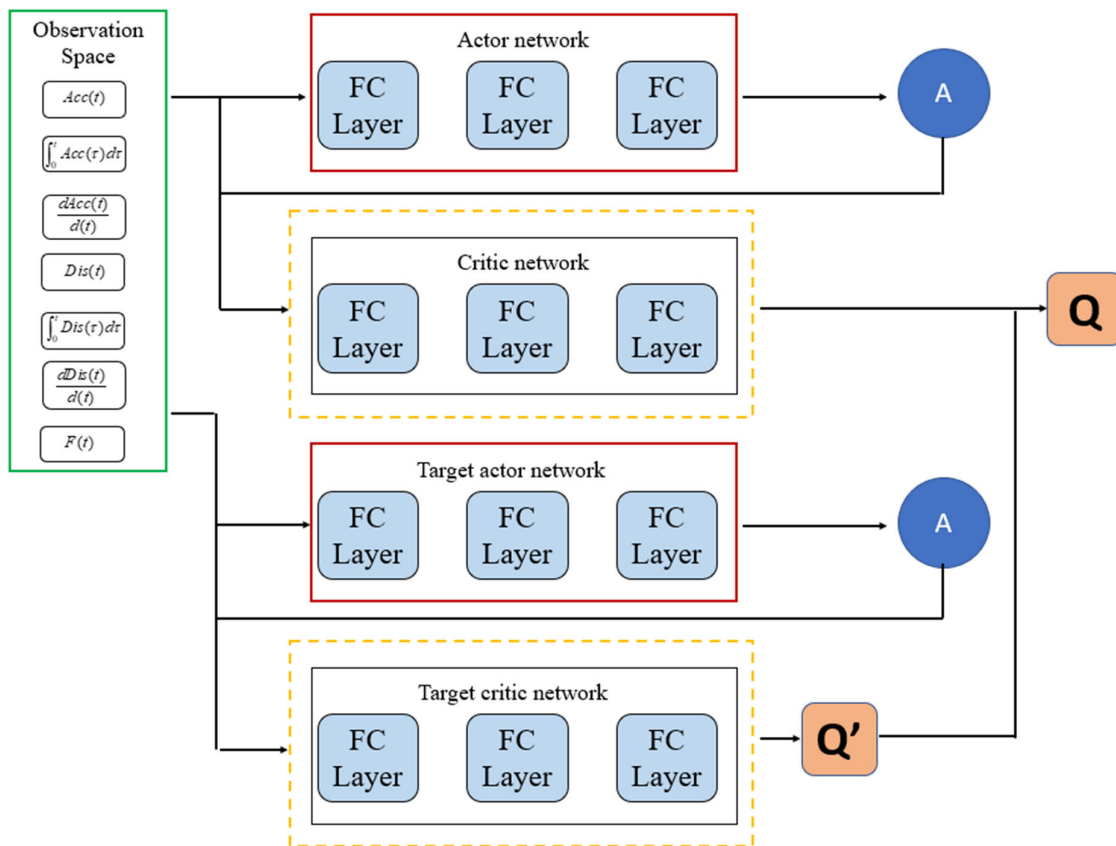
The DPG algorithm deterministically maps states to a specific action through an actor function  $\mu(s|\theta^\mu)$  which specifies the current policy. A chain rule is applied to update the actor from the start distribution J to receive expected return.

$$\begin{aligned} \nabla_{\theta^\mu} J &\approx E_{S_t \sim \rho^\beta} [\nabla_{\theta^\mu} Q(s, a|\theta^Q) \Big|_{s=S_t, a=\mu(S_t|\theta^\mu)}] \\ &= E_{S_t \sim \rho^\beta} [\nabla_a Q(s, a|\theta^Q) \Big|_{s=S_t, a=\mu(S_t)} \nabla_{\theta^\mu} \mu(s|\theta^\mu) \Big|_{s=S_t}] \end{aligned} \quad (53)$$

In [48], it has been proved that this is a policy gradient. When applying a general function approximator such as a neural network in Q learning might pose a negative impact on its convergence due to non-linear characteristics. But it is necessary to do so considering a large

state space situation. A mini batch concept is developed in DDPG is in the same level of DPG. The major contribution here is creating a modified version of DPG together with the approach in DQN which integrates a neural network function approximator to learn in a big state and action space online.

DDPG manages four neural networks at the same time, the actor, critic target-actor, and target-critic. A target network enables the agent to learn from those experiences in a steady policy, which is merely a lagged version of a current agent network. Network structure of a DDPG Algorithm is illustrated in Figure 6-25.



**Figure 6-25 DDPG Network Structure**



## 6.2.2 DDPG Algorithm

---

### DDPG algorithm

---

Randomly initialize critic network  $Q(s, a | \theta^Q)$  and actor  $\mu(s | \theta^\mu)$  and  $\theta^\mu$ .

Initialize target network  $Q'$  and  $\mu'$  with weight  $\theta^{Q'} \leftarrow \theta^Q, \theta^{\mu'} \leftarrow \theta^\mu$

Initialize replay buffer  $R$

**for** episode = 1, M **do**

Initialize a random process  $\mathcal{N}$  for action exploration

Receive initial observation state  $s_1$

**for** t = 1, T **do**

Select action  $a_t = \mu(s_t | \theta^\mu) + \mathcal{N}_t$  according to the current policy and exploration noise

Execute action  $a_t$  and observe reward  $r_t$  and observe new state  $S_{t+1}$

Store transition  $(s_t, a_t, r_t, s_{t+1})$  in  $R$

Sample a random minibatch of  $N$  transitions  $(s_t, a_t, r_t, s_{t+1})$  from  $R$

Set  $y_i = r_i + \gamma Q'(s_{i+1}, \mu'(s_{i+1} | \theta^{\mu'})) | \theta^{Q'}$

Update critic by minimizing the loss:  $L = \frac{1}{N} \sum_i (y_i - Q(s_i, a_i | \theta^Q))^2$

Update the actor policy using the sampled policy gradient:

$$\nabla_{\theta^\mu} J \approx \frac{1}{N} \sum_i \nabla_a Q(s, a | \theta^Q) \Big|_{s=s_i, a=\mu(s_i)} \nabla_{\theta^\mu} \mu(s | \theta^\mu) \Big|_{s_i}$$

Update the target networks:

$$\theta^{Q'} \leftarrow \tau \theta^Q + (1 - \tau) \theta^{Q'}$$

$$\theta^{\mu'} \leftarrow \tau \theta^\mu + (1 - \tau) \theta^{\mu'}$$

**end for**

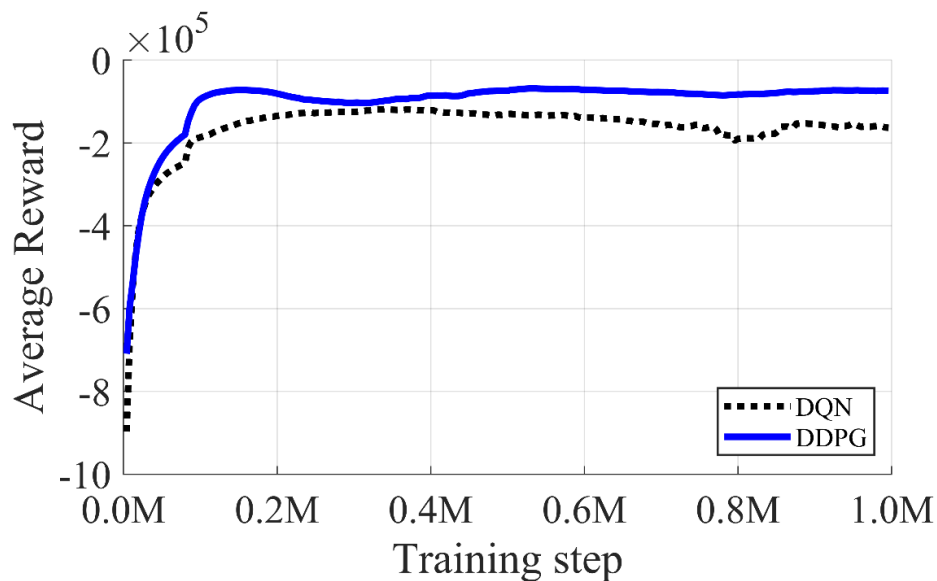
**end for**

---

## 6.2.3 Simulation and comparison

In this part a comparison has been conducted between DQN and DDPG to analyze whether a continuous action space is a key item which impacts to the performance. To guarantee a fair comparison, the neural network architecture is set to be identical with a triple layer fully

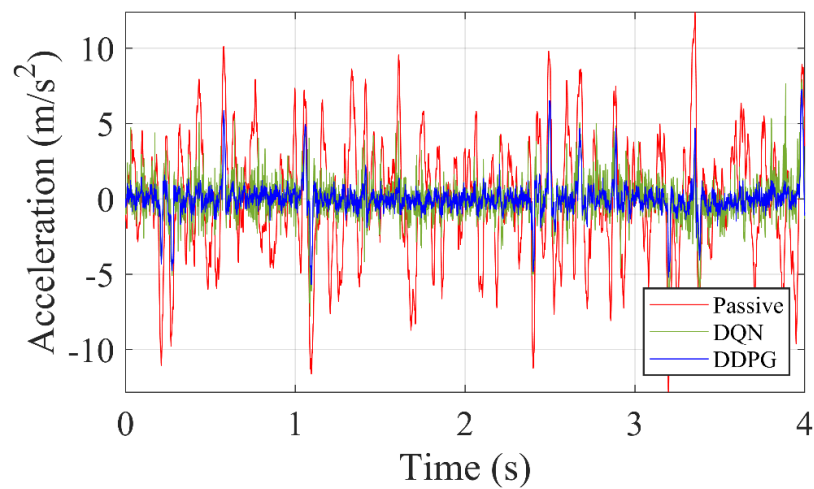
connected structure which contains 96 neurons in each layer. This structure is proved to be a best balance between performance and training time which is based on the research result from in Section 6.1. Similarly, the state space, reward signal, input pavement signal, learning rate and other RL parameters are set to be the same. The action space is obviously different where DQN uses a discrete space and DDPG applies a continuous space. However, in this test, the limits of the action space are defined to be the same which is from -4000N to +4000N. Both DQN and DDPG are trained over 1 million steps. One episode contains 4000 steps to guarantee sufficient pavement features are included in the signal for controllers to learn. The learning rate is set to be  $1e-3$ , greedy rate is set to be 0.99, sample time is set to be 0.001s, mini batch size is set to be 64 and discount factor is set to be 0.9.



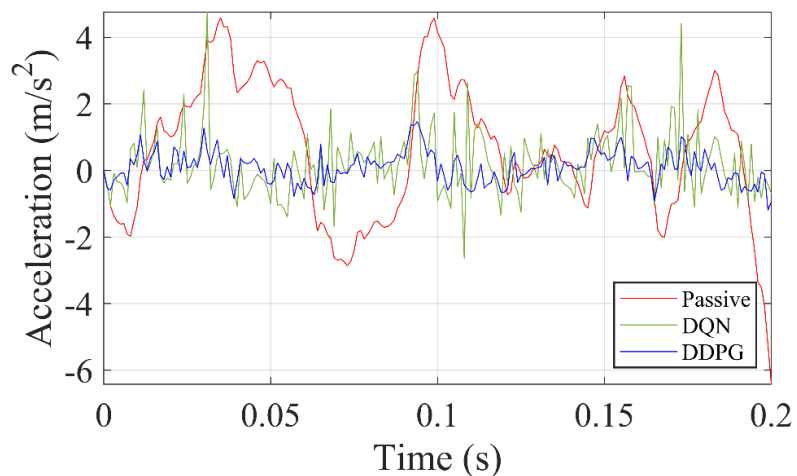
**Figure 6-26 Learning Curve DQN Vs DDPG**

The training curve is shown in Figure 6-26. From the graph, both two algorithms are converged while DDPG has demonstrated a higher reward value than that from DQN. From Figure 6-27 and Figure 6-28, it can be observed that the acceleration performance from a DDPG algorithm is much better than that from a DQN controller. Both DQN and DDPG are able to reduce the peak value from over 10 m/s<sup>2</sup> to around 5 m/s<sup>2</sup> where DDPG can provide less peaks in quantity and overall delivers a smoother oscillation in acceleration. Figure 6-29 demonstrates another

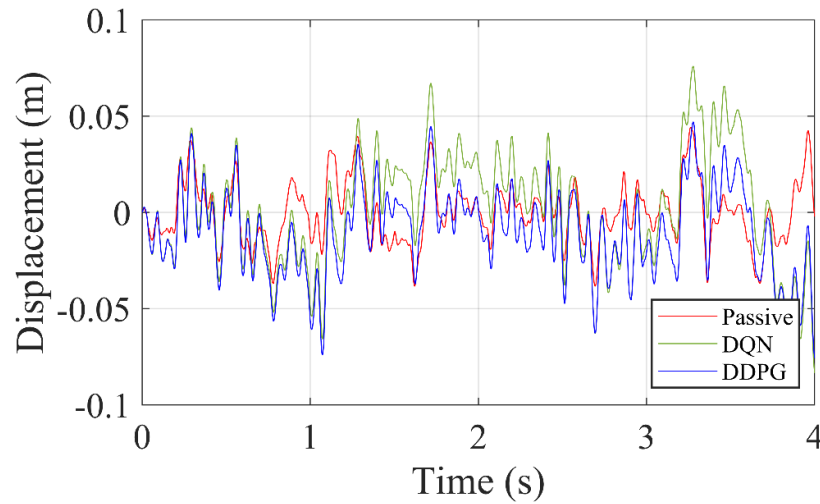
value which is the displacement of suspension, it can be seen that the displacement is increased in an active control situation however this might not be a big issue due to the vehicle running in a normal speed. Overall, it can be concluded that a PPDG algorithm is better than a DQN controller in the simulation which proves that a continuous action space has a better performance than a discrete action space regarding to an actuator control in a dynamic environment.



**Figure 6-27 Acceleration Comparison DQN Vs DDPG**



**Figure 6-28 Acceleration Comparison DQN Vs DDPG Clipped**



*Figure 6-29 Displacement Comparison DQN Vs DDPG*

## 6.3 TD3 Algorithm Application

### 6.3.1 Introduction

Twin Delayed Deep Deterministic Policy Gradients (TD3) [68] has been considered as one of the most powerful and cutting edge algorithms in reinforcement learning. TD3 is basically the next generation of Deep Deterministic Policy Gradient (DDPG) which up to now, has been one of the most widely used algorithms with continuous action space. However, like a lot of other RL algorithms, DDPG can sometimes become unstable and hard to find a solution to converge. This is due to an over estimation to the Q values from the critic network, which can be piling up over time and eventually cause the agent falling on a local optimum or experience catastrophic forgetting. Trying to improve those drawbacks that have been mentioned above, TD3 has applied some changes. First, instead of using just one critic network like DDPG, it uses a dual structure critic network. Second, the actor network is updated less frequently than the critic network, which is called a delayed update. And the last but not least, a regularization technique is applied for adding noise.

### 6.3.1.1 Twin Critic Networks

The first improvement comparing with DDPG is the implementation of a twin critic network. The idea is coming from a method which uses a separated target value function for Q value estimation to reduce the bias [70]. Unfortunately, the attempt is not very much in line with an actor and critic structure due to the reason that the networks are updated so slow that they look similar to each other.

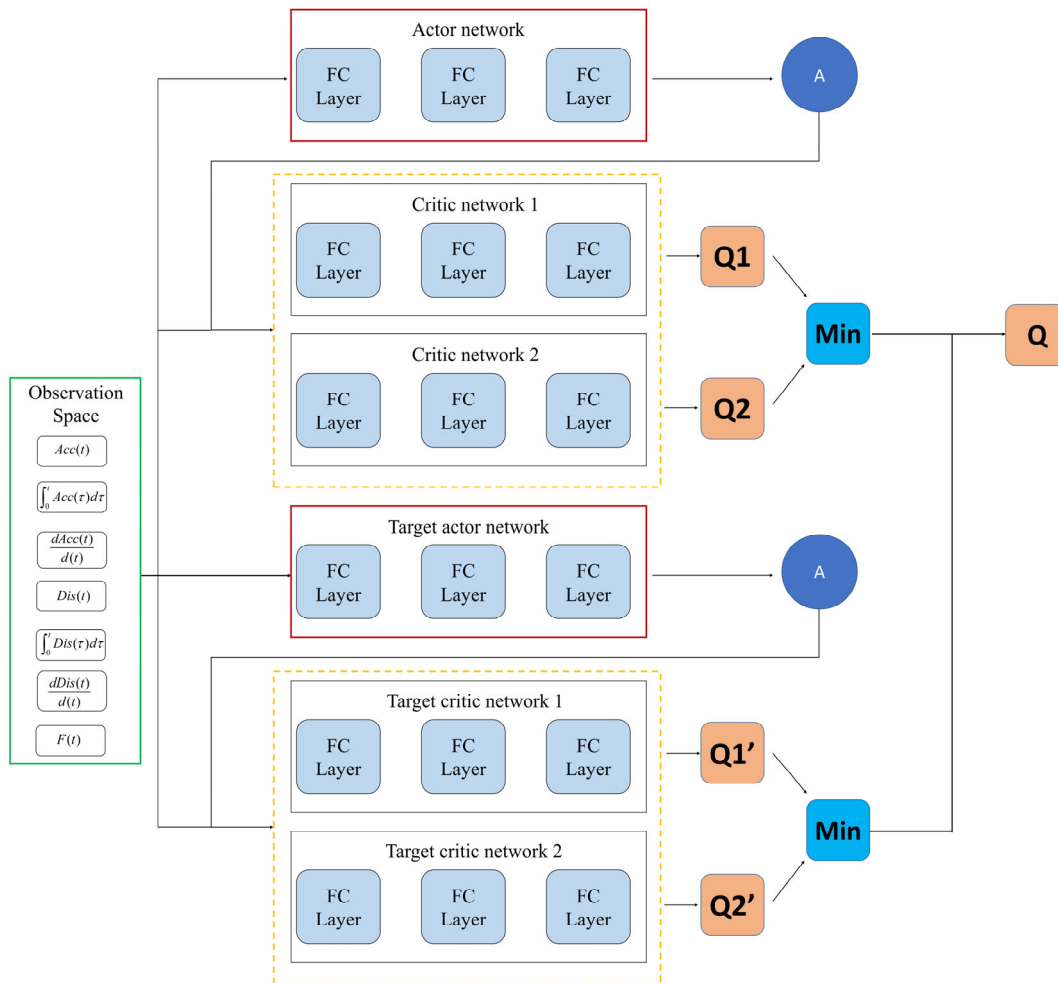
However, in TD3, for the sake of a stable converge from deep function approximators, the 2 target networks have been retained. At the same time, two sets of twin critic networks have been applied to estimate the Q value, and the one with the smaller value is set to be used to update two critic networks. Under this design, there are totally six neural networks that need to be maintained in the TD3 RL frame. The twin critic network strategy also created a bias of low Q value preference. However, being different from an overestimated value which may be spread through the algorithm, an underestimation bias stays where it is, therefore, provides a generally more stable performance. Network structure of a TD3 Algorithm is demonstrated in Figure 6-30.

### 6.3.1.2 Delayed Updates

The interaction between actor and critic network is also a major concern when talking about learning process convergence. Since the deep Q network is constantly updated, when a bad policy is applied and overestimated, in some cases, it might cause a blind iteration from actor and then lead to a continuous deterioration in learning. To be more precisely, when the actor network is updated, it may reach the highest Q value according to the current moment, however in the next step, it is found not to be the optimal one. In this case, the result can sometimes be trapped at a second highest point.

In order to provide a solution towards this issue, the idea of delayed policy updates is implemented to adjust the updates of critic network more frequently than the actor network.

By doing this, a policy with less frequent updates use a value estimate in lower variance will result in higher quality policy updates [68], thus the critic network is able to provide a more stable result which can be used to update the actor network.



**Figure 6-30 TD3 Network Structure**

### 6.3.1.3 Noise Regularization

A problem that can be found in deterministic policies is over fitting to spikes in value estimation. Learning process using deep deterministic policies is easily impacted by function approximation errors when updating the critic, which increasing the variance of the objective. To resolve this problem, TD3 demonstrates a regularization technique that is known as target policy smoothing. The idea assumes that similar actions shall deliver similar value. TD3 adds

a small amount of random noise to the target policy and average them in mini batches. Trying to keep the target policy in an acceptable region, the added noise is clipped. Though above process, a higher value is returned from target to make the policy more stable and more robust to noise and interference.

### 6.3.2 TD3 Algorithm

---

#### TD3 algorithm

---

Initialize critic network  $Q_{\theta_1}, Q_{\theta_2}$  and actor network  $\pi_\phi$ .

With random parameters  $\theta_1, \theta_2, \phi$

Initialize target networks  $\theta'_1 \leftarrow \theta_1, \theta'_2 \leftarrow \theta_2, \phi' \leftarrow \phi$

Initialize replay buffer  $\beta$

**for**  $t = 1$  to  $T$  **do**

Select action with exploration noise  $a \sim \pi(s) + \varepsilon$ ,

$\varepsilon \sim \mathcal{N}(0, \sigma)$  and observe reward  $r$  and new state  $s'$

Store transition tuple  $(s, a, r, s')$  in  $\beta$

Sample mini-batch of  $N$  transitions  $(s, a, r, s')$  from  $\beta$

$\tilde{a} \leftarrow \pi_\phi(s) + \varepsilon, \varepsilon \sim \text{clip}(\mathcal{N}(0, \tilde{\sigma}), -c, c)$

$y \leftarrow r + \gamma \min_{i=1,2} Q_{\theta'_i}(s', \tilde{a})$

Update critics  $\theta_i \leftarrow \min_{\theta_i} N^{-1} \sum (y - Q_{\theta_i}(s, a))^2$

**if**  $t \bmod d$  **then**

Update  $\phi$  by the deterministic policy gradient:

$$\nabla_\phi J(\phi) = N^{-1} \sum \nabla_a Q_{\theta_1}(s, a) \Big|_{a=\pi_\phi(s)} \nabla_\phi \pi_\phi(s)$$

Update the target networks:

$$\theta'_i \leftarrow \tau \theta_i + (1 - \tau) \theta'_i$$

$$\phi' \leftarrow \tau \phi + (1 - \tau) \phi'$$

**end if**

**end for**

---

### **6.3.3 Simulation and Comparison**

#### **6.3.3.1 System Simulation Setup**

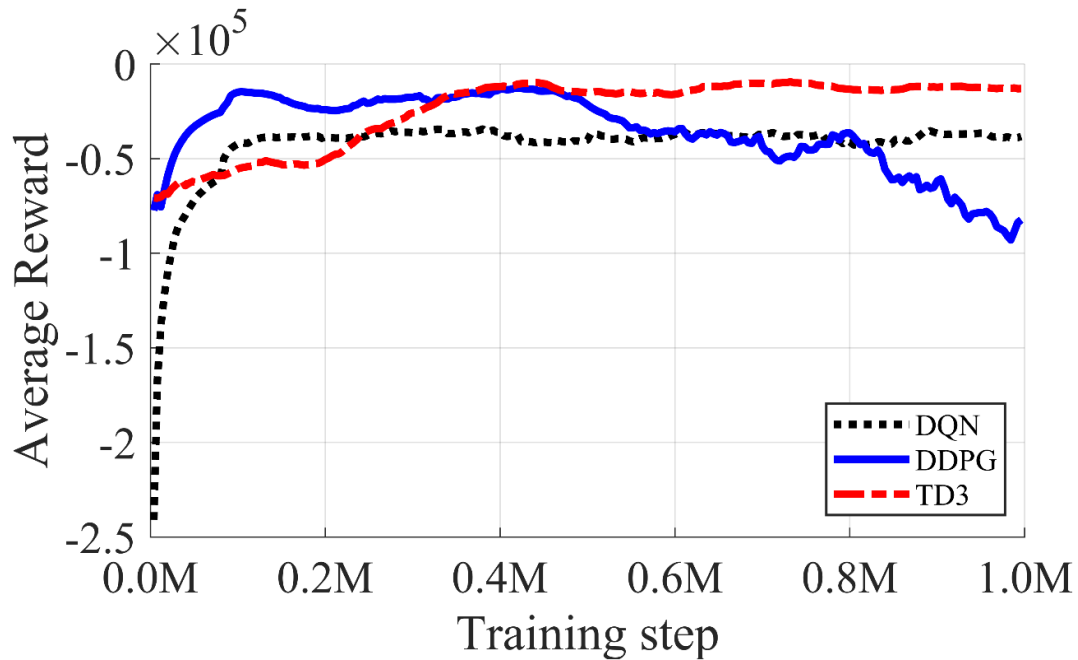
To illustrate the optimal and reliable performance of TD3 algorithm in vehicle suspension control, it is necessary to compare it with other reinforcement learning controllers. A Deep Q Learning algorithm and a DDPG algorithm are selected as benchmarks. To guarantee a fair comparison, the neural network architecture is constructed to be identical across all these three controllers. Based on the same concept, the action space, state space, reward signal and input road pavement signal are set to be the same.

All three algorithms are trained over 1 million steps, which is about 250 episodes. One episode contains 4000 steps to guarantee sufficient pavement features are included in the signal for the controller to learn. The learning rate is set to be  $1e-3$ , greedy rate is set to be 0.99, sample time is set to be 0.001s, mini batch size is set to be 64, and discount factor is set to be 0.9

#### **6.3.3.2 Class B Pavement Signal Test**

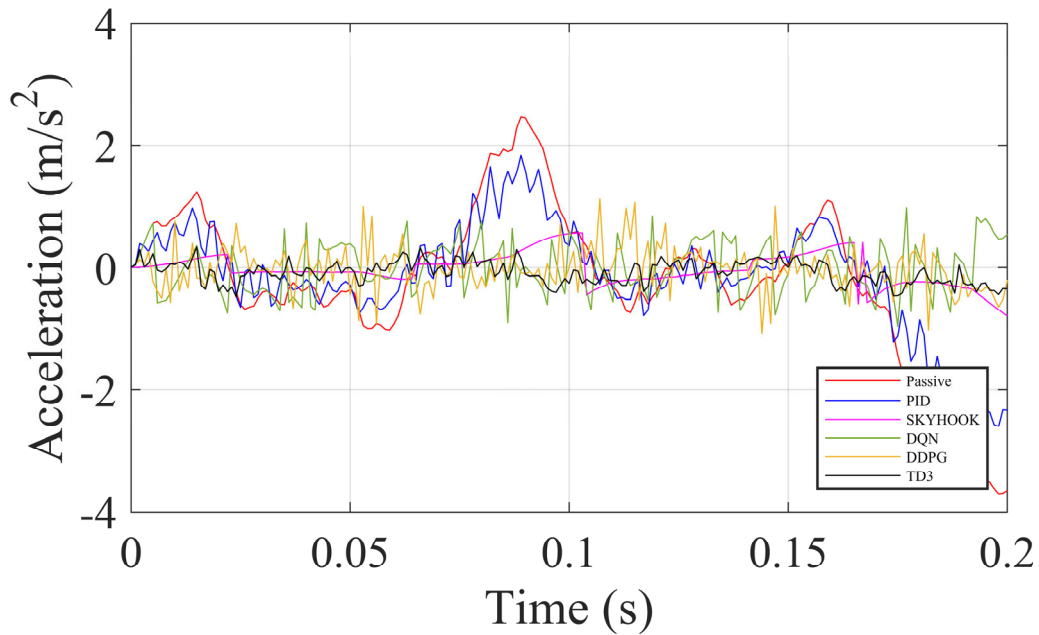
The training curve under a Class B pavement input signal has been illustrated in Figure 6-31 where each line shows the average reward. Both DQN and TD3 are able to converge while DDPG is failed to do so. As explained in the TD3 introduction, this is probably due to the unstable nature from PPDG algorithm where an overestimated Q value error can be accumulated throughout the whole learning progress and misleading the result towards a local peak. By comparing the curve between TD3 and DQN, it is easy to conclude that they both converge quickly and then move stably until then end of learning, however, TD3 can achieve a higher average value in comparing the reward.



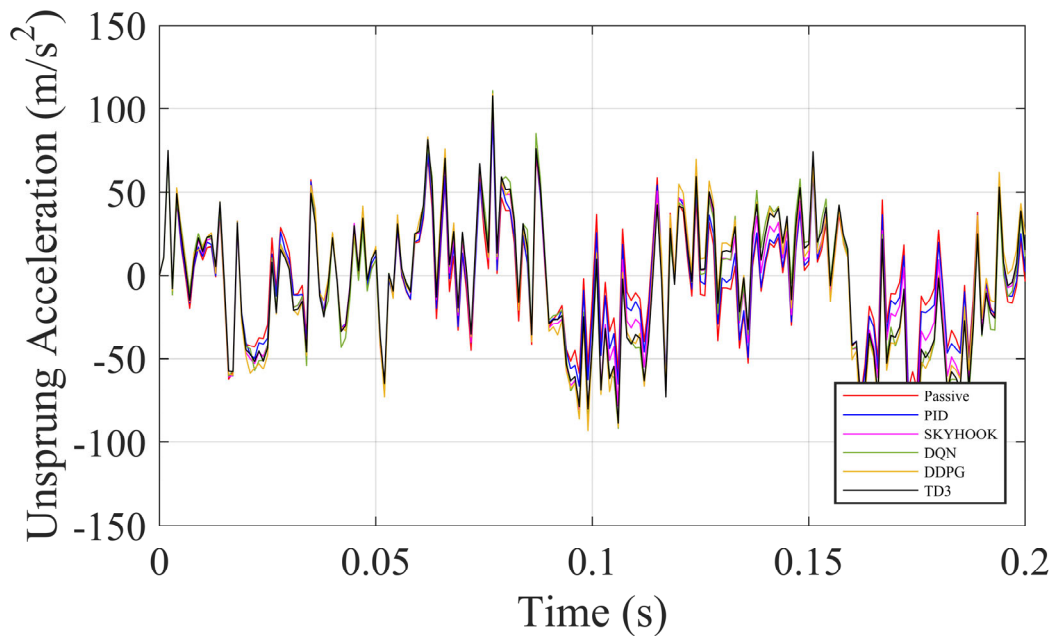


**Figure 6-31 B Class Road Learning Curve DQN Vs DDPG Vs TD3**

By looking into the body acceleration performance which is illustrated in Figure 6-32, PID controlling can provide a certain level of improvement but is still far from satisfactory, while Skyhook control method can reduce the oscillation of acceleration but create sharp steps from time to time. Among machine learning control algorithms, TD3 has outperformed other algorithms. It brings down the acceleration from maximum in  $2m/s^2$  passive control to around  $0.5m/s^2$  and keeps it smooth and stable, while DQN and DDPG are landing on a local optimal which is about  $1m/s^2$ . The diagram has been clipped to show only the interesting area. Figure 6-33 illustrated the comparison results of unsprung mass acceleration, as can be seen from the graph, there is no significant difference between each of the controlling method and the passive control.



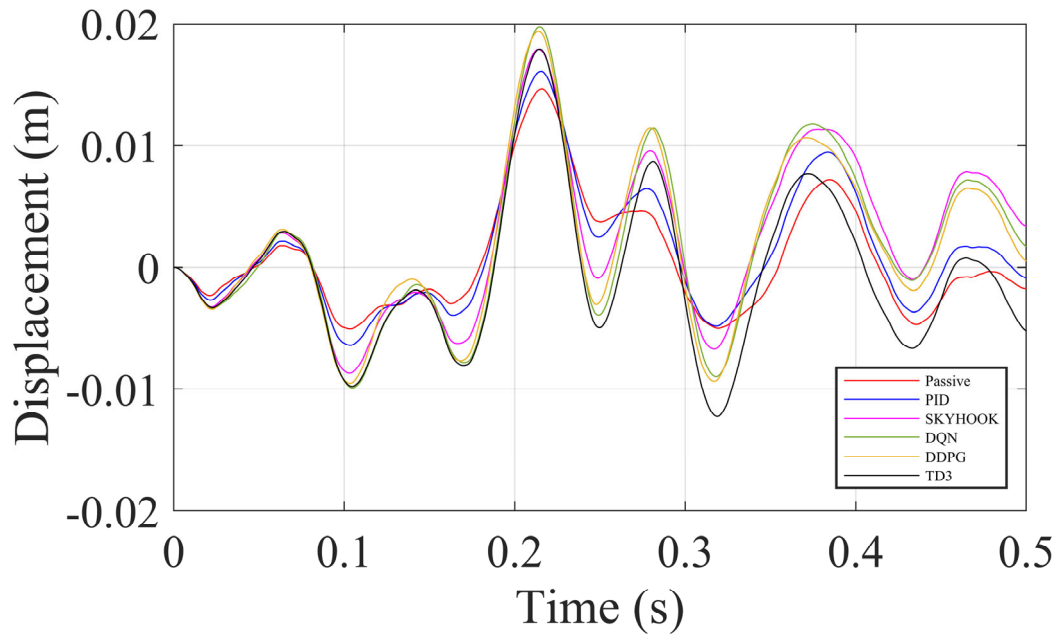
**Figure 6-32 B Class Road Body Acceleration Comparison Clipped**



**Figure 6-33 B Class Road Unsprung Acceleration Comparison Clipped**

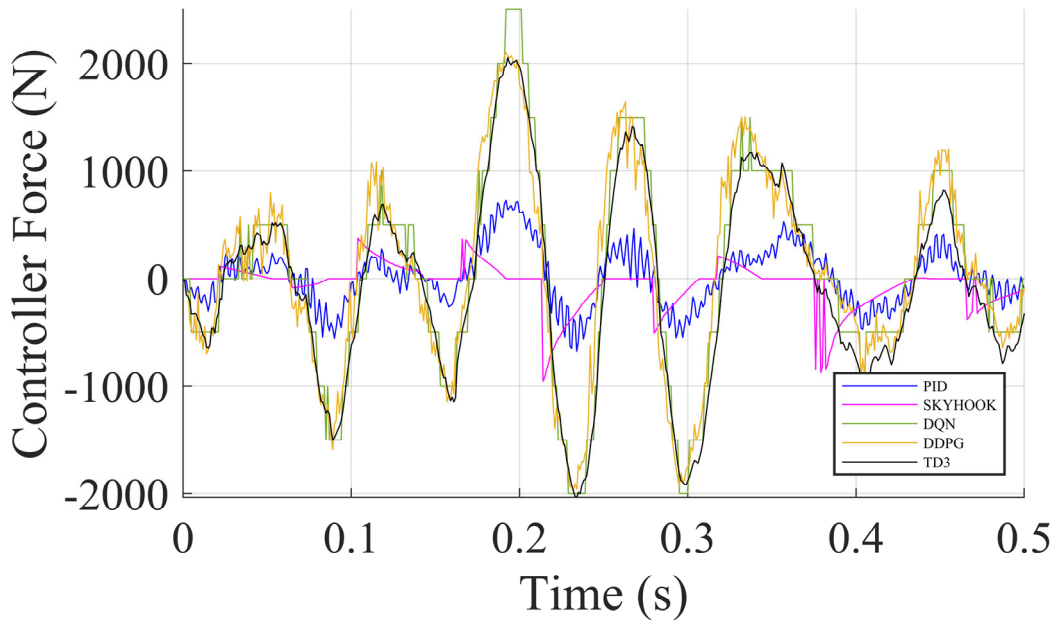
Figure 6-34 demonstrates the comparison of another judgment value, i.e. suspension displacement. As can be seen from the diagram, all controllers increase displacement a bit compared with the performance from a passive suspension. This is due to the nature of an active control phenomenon where the actuator poses an extra force to the spring and damper system. This might slightly influence the vehicle grip performance, which may be neglected

considering the fact that this study is based on a normal Class C vehicle in an urban and rural environment.

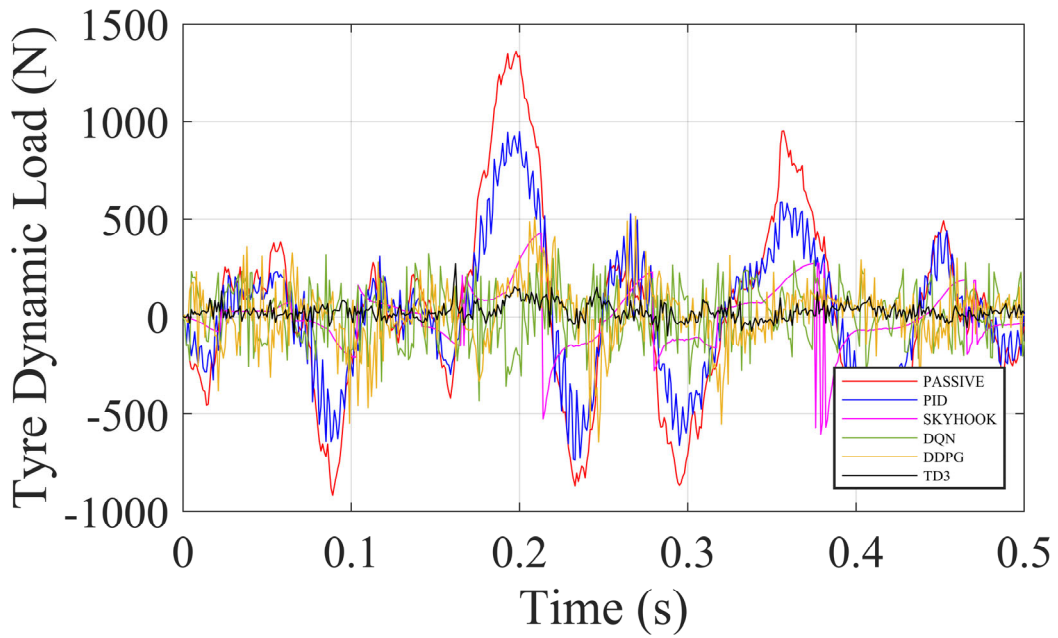


**Figure 6-34 B Class Road Suspension Displacement Comparison Clipped**

Figure 6-35 provides the controller force comparison among different algorithms. PID and Skyhook are not able to provide very accurate control force and the force range is between +1000 N and -1000 N. DQN may improve the accuracy but the performance is still not satisfying due to the nature of only output discrete actions. Both DDPG and TD3 algorithms are delivering a better performance with very accurate controlling force, while the force curve from TD3 is smoother which demonstrates its superiority comparing to other methods. Figure 6-36 illustrates the tyre dynamic load comparison result, from which it is clear that the TD3 algorithm is outperformed significantly all other controllers.



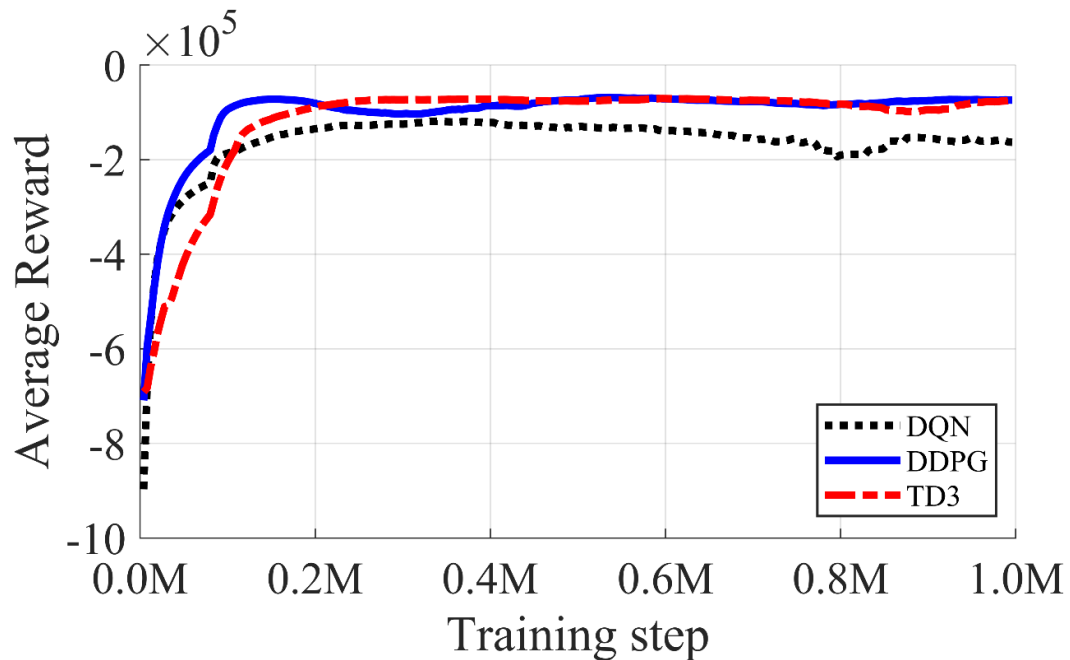
*Figure 6-35 B Class Road Controller Force Comparison Clipped*



*Figure 6-36 B Class Road Tyre Dynamic Load Comparison Clipped*

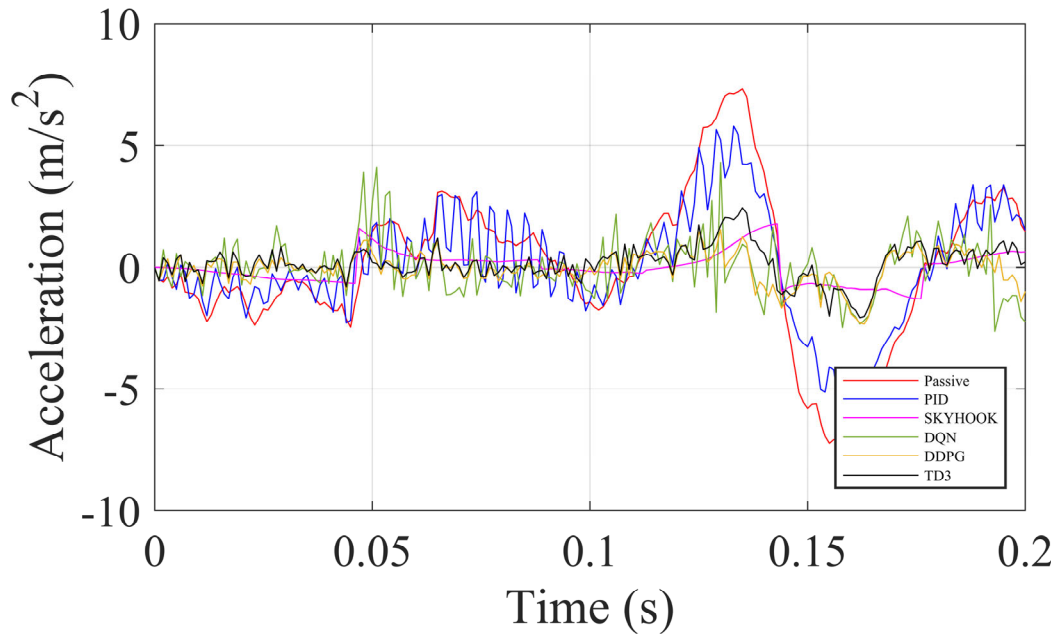
### 6.3.3.3 Class D Pavement Signal Test

The training curve under a Class D pavement input signal is shown in Figure 6-37. From the graph, all three algorithms are able to converge while DDPG and TD3 demonstrate a similar level of average reward. DQN on the other hand provides a less value which indicates limited capability in dealing with a road pavement signal environment.

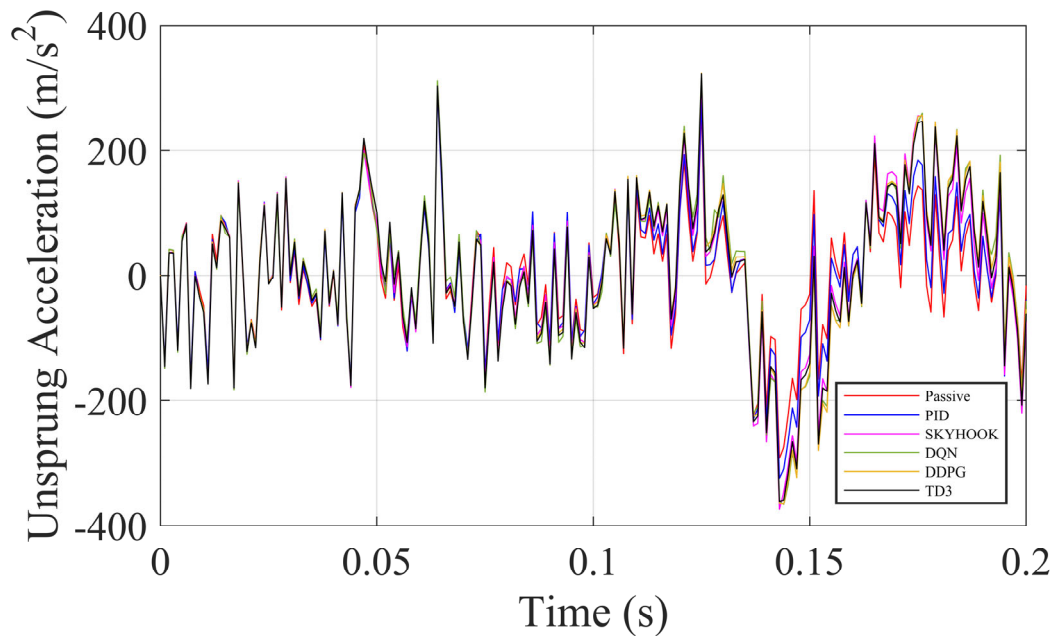


*Figure 6-37 D Class Road Learning Curve DQN Vs DDPG Vs TD3*

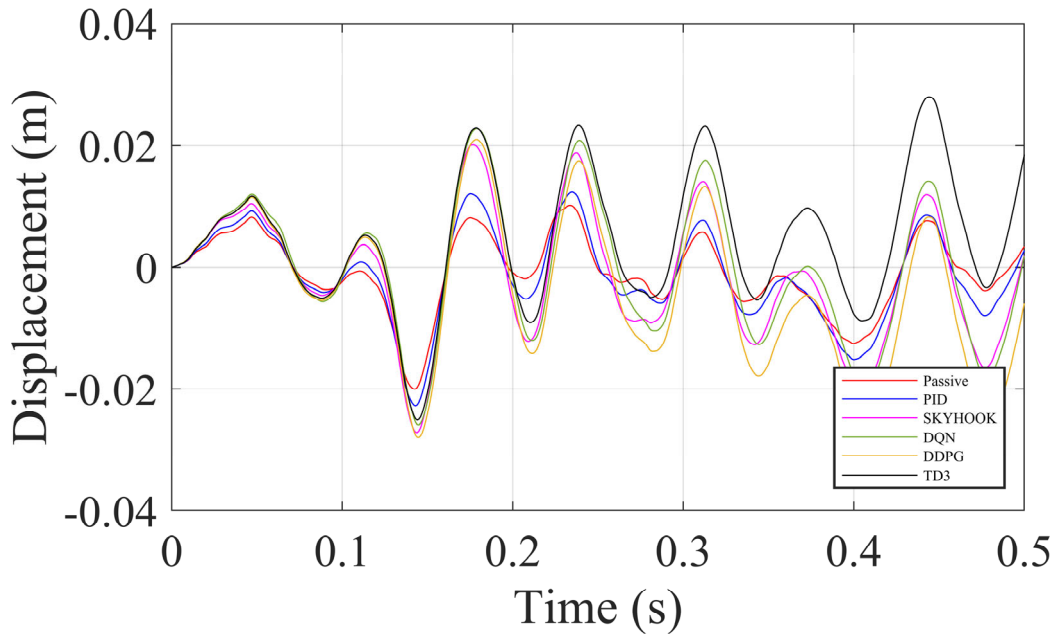
Figure 6-38 to Figure 6-42 illustrate a fairly similar phenomenon in a Class B pavement. A TD3 algorithm is dominating the best performance among all the other controlling method in a bumpy pavement condition which confirms its superiority in active suspension application.



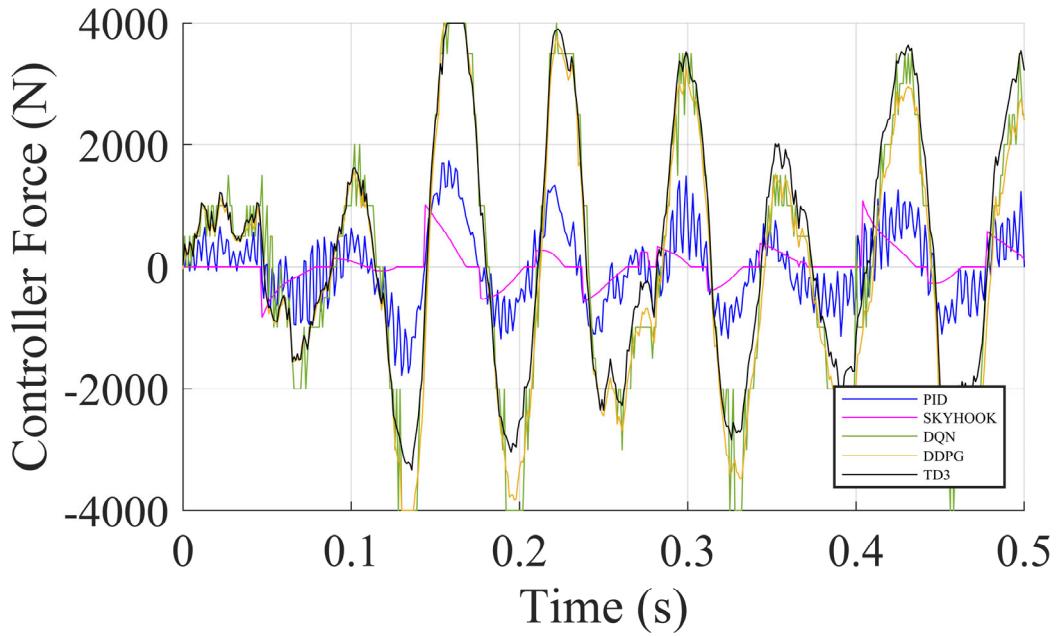
**Figure 6-38 D Class Road Acceleration Comparison Clipped**



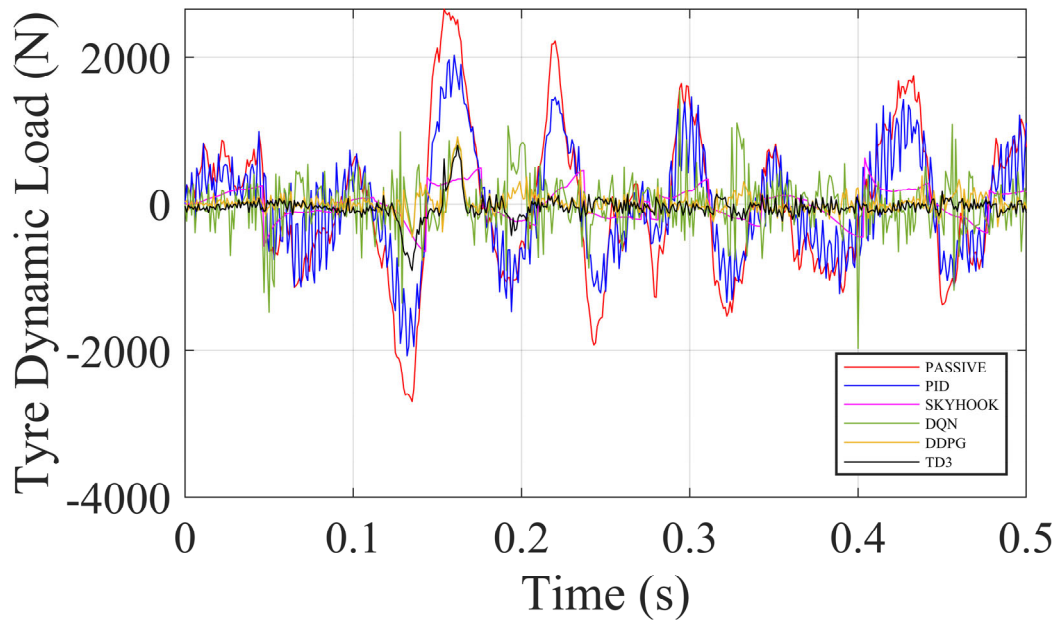
**Figure 6-39 D Class Road Unsprung Acceleration Comparison Clipped**



*Figure 6-40 D Class Road Displacement Comparison Clipped*



*Figure 6-41 D Class Road Controller Force Comparison Clipped*



*Figure 6-42 D Class Road Tyre Dynamic Load Comparison Clipped*



## **Chapter 7. Conclusions and future work**

### **7.1 Summary**

In this paper, a combination of suspension model, a road pavement signal source and a reinforcement learning control system has been established and researched. Comparing with other research in this area, this paper significantly takes the quality of the pavement signal into account, trying to bridge the gap as much as possible between a real road scenario and a simulation circumstance. Also discussed the impact from a variance in neural network architecture and other hyper-parameters towards the performance of the system. Eventually, through the analysis across different state of the art algorithms, it is concluded that a TD3 algorithm has shown its superiority in suspension control under either a smooth or bumpy road condition.

The research also provides a review of the current research status regarding to the active suspension system control. It then looks into a group of the traditional controlling method and latest control method with A.I, followed by a comparison towards their performance. Through these studies, it actually reveals the reason why a machine learning can provide a better controlling result considering its self-learning capability. The review also illustrated that still, there quite some problems that have not been solved yet. For instance, in road pavement simulation, how to guarantee the simulation signal is in line with the current standard. In reinforcement learning, how will the hyper-parameters change the final learning performance when controlling system. How to properly choose a neural network that best fit into the current assembly.

In this study, a dynamic system analysis and modeling establishment towards a suspension system, through state space method, the dynamic property is able to be derived and simulated in Simulink. At the same time, two interfaces have also been created for the input signal from

road pavement and controlling algorithm. It can also be concluded from the study that a reinforcement learning frame is suitable in a suspension control scenario, by referring to multiple current applications inside or outside automotive industry. The study also presented a method in creating an accurate road pavement signal in regarding to the industrialized standard. Therefore, in training the learning algorithm, different levels of roads can be tested to classify the performance of each type of algorithm.

Eventually, a deep study has been done by evaluating the structure of the whole assembly. It starts by analyzing the performance of the learning progress when applying different neural networks followed by a comparison of their controlling result. After that, a group of hyper-parameters is researched to evaluate their impact towards the stability and performance of the system. From there, the key impact factors are revealed and recorded to benefit the further research of the project. Then, three state-of-art algorithms are applied and tested to find out the optimized solution in regarding to a changing environment of road pavement.

## **7.2 Prospective of future work**

Through the simulation work from this study, it has been proved that a reinforcement learning controlling unit has an optimal performance regarding to its self-study capability in dealing with a changing environment. Of course, there are some works need to be carried out for a further study.

In regarding to the current project, a quarter car model is utilized to represent a suspension model. The next step can be to replace that quarter car model with a half car or full car model to demonstrate a better simulation result regarding to the reality. Also, model parameters uncertainty [75] can be involved when designing the suspension model to check stability and robust performance of a machine learning controlling unit.

Considering the active suspension design, the current method is using an actuator to create an additional force to intervene the vibrating process of the system. Also, it can be considered as a solution to utilize a damper with adjustable damping coefficient. In that case, the damping force is controllable in reacting to the different amplitude of oscillation. Or even a combination of both method in comparing with only one method applied. It is definitely possible for a reinforcement learning agent to deal with that situation by adding one more action space in the frame.

Hardware validation needs to be carried out, in which sample frequency is a key parameter that impacts the final performance of the suspension system. According to the simulation result, a 1000 Hz frequency is needed to guarantee a smooth control of the acceleration. This requirement applies to both the acceleration sensor and the actuator of a spring damper system. By looking into the current industrialized product, there are couple of types of parts that can meet this requirement, therefore clears the roadblocks on the job of a hardware in loop setup. Also, by dealing with a hardware in loop system, time delay is also a significant area to be considered to guarantee the efficiency of the whole system. Once the test and verification work are done, the system can be ready for a real vehicle test afterwards.

## Reference

- [1] F. D. A. F. P. Fael, "Software for Simulation of Vehicle-Road Interaction," in *New Advances in Information Systems and Technologies*, vol. 1: Springer, 2016, pp. 681-690.
- [2] G. J. Stein, R. Zahoranský, T. P. Gunston, L. Burström, and L. Meyer, "Modelling and simulation of a fore-and-aft driver's seat suspension system with road excitation," *International Journal of Industrial Ergonomics*, vol. 38, no. 5-6, pp. 396-409, 2008, doi: 10.1016/j.ergon.2007.10.016.
- [3] B. Chander, "Modelling and Analysis of Half Car Model Using Fuzzy Logic Control," *Research Gate*, 2009.
- [4] S. L. Rajesh Kattiboina, "Neuro PID Speed Controller for Motor," 2015.
- [5] Q. Tong *et al.*, "Combining Machine Learning Potential and Structure Prediction for Accelerated Materials Design and Discovery," *J Phys Chem Lett*, vol. 11, no. 20, pp. 8710-8720, Oct 15 2020, doi: 10.1021/acs.jpcllett.0c02357.
- [6] E. A. Gustafsson, "Comparison\_of\_Design\_Automation\_and\_Machine\_Learning\_algorithms\_for\_creation\_of\_easily\_modifiable\_sp," 2020.
- [7] M. A. Bessa and S. Pellegrino, "Design of ultra-thin shell structures in the stochastic post-buckling range using Bayesian machine learning and optimization," *International Journal of Solids and Structures*, vol. 139-140, pp. 174-188, 2018, doi: 10.1016/j.ijsolstr.2018.01.035.
- [8] Y. Mao, "Designing complex architected materials with generative adversarial networks," 2020.
- [9] C. Lin, "Modeling 3D Shapes by Reinforcement Learning," 2020.
- [10] V. Colla, S. Cateni, A. Maddaloni, and A. Vignali, "A Modular Machine-Learning-Based Approach to Improve Tensile Properties Uniformity Along Hot Dip Galvanized Steel Strips for Automotive Applications," *Metals*, vol. 10, no. 7, 2020, doi: 10.3390/met10070923.
- [11] F. Danglade, J.-P. Pernot, and P. Véron, "On the use of Machine Learning to Defeature CAD Models for Simulation," *Computer-Aided Design and Applications*, vol. 11, no. 3, pp. 358-368, 2013, doi: 10.1080/16864360.2013.863510.
- [12] K. Hayashi and M. Ohsaki, "Reinforcement learning for optimum design of a plane frame under static loads," *Engineering with Computers*, 2020, doi: 10.1007/s00366-019-00926-7.
- [13] H. Jung, J. Jeon, D. Choi, and J.-Y. Park, "Application of Machine Learning Techniques in Injection Molding Quality Prediction: Implications on Sustainable Manufacturing Industry," *Sustainability*, vol. 13, no. 8, 2021, doi: 10.3390/su13084120.
- [14] S. Lee, Y. Cho, and Y. H. Lee, "Injection Mold Production Sustainable Scheduling Using Deep Reinforcement Learning," *Sustainability*, vol. 12, no. 20, 2020, doi: 10.3390/su12208718.
- [15] C.-T. Chen and G. X. Gu, "Machine learning for composite materials," *MRS Communications*, vol. 9, no. 2, pp. 556-566, 2019, doi: 10.1557/mrc.2019.32.
- [16] S. T. H. Akcay, "Aspects of achievable performance of quarter-car active suspensions," *Journal of Sound and Vibration*, vol. 322:15-28, 2008.
- [17] S. T. H. Akcay, "Influence of tire damping on mixed H<sub>2</sub>-H<sub>∞</sub> synthesis of half-car active suspensions," *Journal of Sound and Vibration*, vol. 322:15-28, 2009.
- [18] Z. Qiang, Y. Jiaying, and L. Decai, "Intelligent Compound Control of Vehicle Active Suspension Based on RBF Neural Network," presented at the 2011 Third International Conference on Measuring Technology and Mechatronics Automation, 2011.
- [19] Y. H. N. Yagiz, "Back stepping control of a vehicle with active suspensions," *Control Engineering Practice*, vol. 16: 1457-1467, 2008.
- [20] S.-J. Kim, "Vibration Control of a Vehicle Active Suspension System Using a DDPG Algorithm," 2018.
- [21] J. S. M. Witters, "Black-box model identification for a continuously variable, electro-hydraulic semi-active damper," *Mechanical Systems and Signal Processing*, vol. 24: 4-18, 2010.

- [22] L. Ming, L. Yibin, R. Xuewen, Z. Shuaishuai, and Y. Yanfang, "Semi-Active Suspension Control Based on Deep Reinforcement Learning," *IEEE Access*, vol. 8, pp. 9978-9986, 2020, doi: 10.1109/access.2020.2964116.
- [23] S. S. A. M.M. Fateh, "Impedance control of an active suspension system," *Mechatronics*, vol. 19:134-140, 2009.
- [24] J. Liu, "Advanced PID control and its Matlab Simulation," 2007. Beijing Publishing House of Electronics Industry.
- [25] Y. Z. A. Aziz Sezgin, "Analysis of the vertical vibration effects on ride comfort of vehicle driver," 2012.
- [26] M. Palermo, "Effects of a large unsprung mass on the ride comfort of a lightweight fuel-cell urban vehicle," 2009.
- [27] H. P. Heping Wang, "Study of Passenger Comfort Relating to Vehicle Vibration," 2009.
- [28] T. Z. Chuanyin Tang, "A Study of Vehicle Vibrating Comfort Evaluation," 2008.
- [29] P. W. Shilei Zhou, Yang Tian\*, Nong Zhang, "Study on the energy economy and vibration characteristics of an in-wheel drive electric hydraulic hybrid vehicle."
- [30] W. Schiehlen, "White noise excitation of road vehicle structures," *Sadhana*, vol. 31, 4, pp. 487-503, 2006.
- [31] S. C. Fan Lu, "Modelling and Simulation of Road and surface excitation on vehicle in time domain," *qiche gongcheng*, vol. 5, no. 37, 2015 2015.
- [32] F. T. a. Y.-F. Hong, "Generation of Random Road Profiles," *Collections of the 24th National Academic Symposium of Chinese Mechanical Engineering Society*, vol. B04-0001, 2008.
- [33] I. 8608, "Mechanical Vibrations-Road Surface Profiles-Reported Of Measured Data, ISO 8608," *International Organization For Standardization*, 1995.
- [34] B. J. Goenaga, L. G. Fuentes Pumarejo, and O. A. Mora Lerma, "Evaluation of the methodologies used to generate random pavement profiles based on the power spectral density: An approach based on the International Roughness Index," *Ingeniería e Investigación*, vol. 37, no. 1, 2017, doi: 10.15446/ing.investig.v37n1.57277.
- [35] Y. Zhang, "Non-stationary\_Random\_Vibration\_Analysis\_of\_Vehicle," *13th National Conference on Mechanisms and Machines*, vol. NaCoMM-2007-77, 2008.
- [36] K. B. Singh and S. Taheri, "Estimation of tire-road friction coefficient and its application in chassis control systems," *Systems Science & Control Engineering*, vol. 3, no. 1, pp. 39-61, 2014, doi: 10.1080/21642583.2014.985804.
- [37] Y. Wang *et al.*, "Tire Road Friction Coefficient Estimation: Review and Research Perspectives," *Chinese Journal of Mechanical Engineering*, vol. 35, no. 1, 2022, doi: 10.1186/s10033-021-00675-z.
- [38] C. Bragança *et al.*, "Calibration and validation of a freight wagon dynamic model in operating conditions based on limited experimental data," *Vehicle System Dynamics*, pp. 1-27, 2021, doi: 10.1080/00423114.2021.1933091.
- [39] F. Cunto and F. F. Saccomanno, "Calibration and validation of simulated vehicle safety performance at signalized intersections," *Accid Anal Prev*, vol. 40, no. 3, pp. 1171-9, May 2008, doi: 10.1016/j.aap.2008.01.003.
- [40] N. A. Seegmiller, "Dynamic Model Formulation and Calibration for Wheeled Mobile Robots," Doctor of Philosophy, The Robotics Institute, Carnegie Mellon University, 2014.
- [41] D. Ribeiro, R. Calçada, R. Delgado, M. Brehm, and V. Zabel, "Finite-element model calibration of a railway vehicle based on experimental modal parameters," *Vehicle System Dynamics*, vol. 51, no. 6, pp. 821-856, 2013, doi: 10.1080/00423114.2013.778416.
- [42] J. Fender, F. Duddeck, and M. Zimmermann, "On the calibration of simplified vehicle crash models," *Structural and Multidisciplinary Optimization*, vol. 49, no. 3, pp. 455-469, 2013, doi: 10.1007/s00158-013-0977-7.

- [43] C. G. Bonin G., Loprencipe G., Sbrolli M., "RIDE QUALITY EVALUATION 8 DOF VEHICLE MODEL CALIBRATION," presented at the 4th INTERNATIONAL SIIV CONGRESS, PALERMO (ITALY), 2007.
- [44] B. Zhao, T. Nagayama, M. Toyoda, N. Makihata, M. Takahashi, and M. Ieiri, "Vehicle Model Calibration in the Frequency Domain and its Application to Large-Scale IRI Estimation," *Journal of Disaster Research*, vol. 12, no. 3, pp. 446-455, 2017, doi: 10.20965/jdr.2017.p0446.
- [45] A. Fares and A. Bani Younes, "Online Reinforcement Learning-Based Control of an Active Suspension System Using the Actor Critic Approach," *Applied Sciences*, vol. 10, no. 22, 2020, doi: 10.3390/app10228060.
- [46] W. Wang, K. Tian, and J. Zhang, "Dynamic Modelling and Adaptive Control of Automobile Active Suspension System," *Journal Européen des Systèmes Automatisés*, vol. 53, no. 2, pp. 297-303, 2020, doi: 10.18280/jesa.530218.
- [47] P. Zheng and J. W. Gao, "Damping force and energy recovery analysis of regenerative hydraulic electric suspension system under road excitation: modelling and numerical simulation," *Math Biosci Eng*, vol. 16, no. 6, pp. 6298-6318, Jul 8 2019, doi: 10.3934/mbe.2019314.
- [48] G. L. David Silver, Nicolas Heess, Thomas Degris, Daan Wierstra, Martin Riedmiller, "Deterministic Policy Gradient Algorithms," presented at the 31st International Conference on Machine Learning, 2014.
- [49] J. Zhang, F. Chen, Z. Cui, Y. Guo, and Y. Zhu, "Deep Learning Architecture for Short-Term Passenger Flow Forecasting in Urban Rail Transit," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 11, pp. 7004-7014, 2021, doi: 10.1109/tits.2020.3000761.
- [50] H. Zheng, F. Lin, X. Feng, and Y. Chen, "A Hybrid Deep Learning Model With Attention-Based Conv-LSTM Networks for Short-Term Traffic Flow Prediction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 11, pp. 6910-6920, 2021, doi: 10.1109/tits.2020.2997352.
- [51] T. Chu, J. Wang, L. Codeca, and Z. Li, "Multi-Agent Deep Reinforcement Learning for Large-Scale Traffic Signal Control," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 3, pp. 1086-1095, 2020, doi: 10.1109/tits.2019.2901791.
- [52] J. J. Q. Yu, W. Yu, and J. Gu, "Online Vehicle Routing With Neural Combinatorial Optimization and Deep Reinforcement Learning," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 10, pp. 3806-3817, 2019, doi: 10.1109/tits.2019.2909109.
- [53] G. Krummenacher, C. S. Ong, S. Koller, S. Kobayashi, and J. M. Buhmann, "Wheel Defect Detection With Machine Learning," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 4, pp. 1176-1187, 2018, doi: 10.1109/tits.2017.2720721.
- [54] O. G. Bowen Baker, Nikhil Naik, Ramesh Raskar, "DESIGNING NEURAL NETWORK ARCHITECTURES USING REINFORCEMENT LEARNING," presented at the ICLR 2017, Cambridge MA, USA, 2017.
- [55] C. Urrea, F. Garrido, and J. Kern, "Design and Implementation of Intelligent Agent Training Systems for Virtual Vehicles," *Sensors (Basel)*, vol. 21, no. 2, Jan 12 2021, doi: 10.3390/s21020492.
- [56] G. B. M.Y Rafiq, D.J Easterbrook, "Neural network design for engineering applications," *Computers & Structures*, vol. 79, no. 17, pp. 1541-1552, 2001. ELSEVIER.
- [57] W. S. Kyuyeon Hwang, "Fixed-point feedforward deep neural network design using weights +1, 0, and -1," presented at the 2014 IEEE Workshop on Signal Processing Systems (SIPS), Belfast, UK, 20-22 Oct. 2014, 2014.
- [58] C. Lo, Y.-Y. Su, C.-Y. Lee, and S.-C. Chang, "A Dynamic Deep Neural Network Design for Efficient Workload Allocation in Edge Computing," presented at the 2017 IEEE International Conference on Computer Design (ICCD), 2017.

- [59] T. N. S. O. V. A. S. H. Sak, "Convolutional, Long Short-Term Memory, fully connected Deep Neural Networks," presented at the 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), South Brisbane, QLD, Australia, 2015.
- [60] J. Zhao, F. Deng, Y. Cai, and J. Chen, "Long short-term memory - Fully connected (LSTM-FC) neural network for PM2.5 concentration prediction," *Chemosphere*, vol. 220, pp. 486-492, Apr 2019, doi: 10.1016/j.chemosphere.2018.12.128.
- [61] M. Kohler and S. Langer, "On the rate of convergence of fully connected deep neural network regression estimates," *The Annals of Statistics*, vol. 49, no. 4, pp. 2231–2249, 2021, doi: 10.1214/20-aos2034.
- [62] Z. Yonglin and Z. Jiafan, "Numerical simulation of stochastic road process using white noise filtration," *Mechanical Systems and Signal Processing*, vol. 20, no. 2, pp. 363-372, 2006, doi: 10.1016/j.ymsp.2005.01.009.
- [63] L. Sun, "Computer simulation and field measurement of dynamic pavement loading," *Mathematics and Computers in Simulation*, vol. 56, no. 3, pp. 297-313, 2001. ELSEVIER.
- [64] L. Sun, "Simulation of pavement roughness and IRI based on power spectral density," *Mathematics and Computers in Simulation*, vol. 61, no. 2, pp. 77-88, 2003. ELSEVIER.
- [65] S. C. Hongbin Ren, "Model-of-Excitation-of-Random-Road-Profile-in-Time-Domain-for-a-Vehicle-with-Four-Wheels," 2011.
- [66] J. Chen, "Modeling and Simulation on Stochastic Road Surface Irregularity Based on Matlab / Simulink," *Transactions of the Chinese Society for Agriculture Machinery*, 2010, Art no. 1000-1298( 2010) 03-0011-05. China Academic Journal Electronic Publishing House.
- [67] P. W. Daoyu Shen, Shilei Zhou, Nong Zhang, "Signal\_Simulation\_of\_Stochastic\_Road\_Excitation," presented at the The 19th Asia Pacific Vibration Conference, October 2021, 2021, Conference Paper.
- [68] S. Fujimoto, H. Hoof, and D. Meger, "Addressing Function Approximation Error in Actor-Critic Methods," presented at the Proceedings of the 35th International Conference on Machine Learning, Proceedings of Machine Learning Research, 2018. [Online]. Available: <https://proceedings.mlr.press/v80/fujimoto18a.html>.
- [69] T. P. Lillicrap\*, J. J. H. , A. P. , Nicolas Heess,, and Y. T. Tom Erez, David Silver & Daan Wierstra, "CONTINUOUS CONTROL WITH DEEP REINFORCEMENT LEARNING," presented at the ICLR 2016, 2016.
- [70] A. G. Hado van Hasselt, David Silver, "Deep Reinforcement Learning with Double Q-Learning," presented at the Proceedings of the AAAI Conference on Artificial Intelligence, 2016.
- [71] Y. Hou, "The Study of Combining Deep Reinforcement Learning with Prioritized Experience Replay," 2017.
- [72] H. Wei *et al.*, "Deep reinforcement learning based direct torque control strategy for distributed drive electric vehicles considering active safety and energy saving performance," *Energy*, vol. 238, 2022, doi: 10.1016/j.energy.2021.121725.
- [73] A. G. Xi Chen, John Folkesson, M°arten Bj°orkman and Patric Jensfelt, "Deep\_Reinforcement\_Learning\_to\_Acquire\_Navigation\_Skills\_for\_Wheel-Legged\_Robots\_in\_Complex\_Environments," presented at the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 2018.
- [74] M. R. Roland Hafner, "Neural\_Reinforcement\_Learning\_Controllers\_for\_a\_Real\_Robot\_Application," presented at the IEEE International Conference on Robotics and Automation, Roma, Italy, 2007.
- [75] W. Gao, N. Zhang, and J. Dai, "A stochastic quarter-car model for dynamic analysis of vehicles with uncertain parameters," *Vehicle System Dynamics*, vol. 46, no. 12, pp. 1159-1169, 2008, doi: 10.1080/00423110701884575.