Optimal Input Excitation for Parameter Estimation in Electromechanical Brake Actuators

Studiengang Mechatronik
Master thesis in the department of Electrical Engineering and Information Technology
by Shengya Guo
Date of submission: 2. December 2024

Erstprüfer: Prof. Dr.-Ing. Rolf Findeisen
Betreuer: Dr.-Ing. Eric Lenz
Betreuer: M. Sc. Daniel Stümke
Darmstadt

TECHNISCHE UNIVERSITÄT DARMSTADT

CCPS Control and Cyber-Physical Systems

Electrical Engineering and Information Technology Department

Optimal Input Excitation for Parameter Estimation in Electromechanical Brake Actuators
Studiengang Mechatronik

Master thesis in the department of Electrical Engineering and Information Technology by Shengya Guo

Date of submission: 2. December 2024

Darmstadt

Technische Universität Darmstadt
Institut für Automatisierungstechnik und Mechatronik
Fachgebiet Control and Cyberphysical Systems
Prof. Dr.-Ing. Rolf Findeisen

Task definition

With the ongoing electrification of passenger vehicles, the transition from hydraulic to electromechanical brake (EMB) systems becomes attractive. This new kind of braking system promises advantages, as it removes bulky hydraulic components, enables fast wheel individual brake actions, and integrates seamlessly with the X-by-wire paradigm, but it also poses new challenges which need to be overcome. To enable the reliable operation of EMB systems, exact knowledge about system parameters e.g., actuator motor torque constant, friction coefficients or the brake pad stiffness, is essential [1]. Due to aging and wear or manufacturing imperfections these quantities are uncertain and may vary over the component's lifetime. Therefore, a method which reliably estimates system parameters is of great importance. An example of such a method in the context of EMB is presented in [2], where a genetic algorithm is used to obtain a control input which maximizes an information gain criterion (D-Optimality). In other engineering domains, the optimal excitation problem is tackled with Reinforcement Learning (RL) [3] and Chance Constraints [4]. The aim of the offered master's thesis is to investigate how optimal excitation strategies can be realized (open- or closed-loop, with RL or other optimization techniques, etc.) for a mechatronic system such as the electromechanical brake.

Tasks:

- Literature research on parameter estimation for mechatronic systems
- Formulation of a simplified system model for analysis of excitation strategies
- Implementation of a suitable excitation strategy
- Evaluation based on simulation results and optionally also on prototype hardware

This thesis is offered in cooperation with Robert Bosch GmbH (Corporate Research).

Erklärung zur Abschlussarbeit gemäß § 22 Abs. 7 APB TU Darmstadt

Hiermit erkläre ich, Shengya Guo, dass ich die vorliegende Arbeit gemäß § 22 Abs. 7 APB der TU Darmstadt selbstständig, ohne Hilfe Dritter und nur mit den angegebenen Quellen und Hilfsmitteln angefertigt habe. Ich habe mit Ausnahme der zitierten Literatur und anderer in der Arbeit genannter Quellen keine fremden Hilfsmittel benutzt. Die von mir bei der Anfertigung dieser wissenschaftlichen Arbeit wörtlich oder inhaltlich benutzte Literatur und alle anderen Quellen habe ich im Text deutlich gekennzeichnet und gesondert aufgeführt. Dies gilt auch für Quellen oder Hilfsmittel aus dem Internet.

Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Mir ist bekannt, dass im Falle eines Plagiats (§ 38 Abs. 2 APB) ein Täuschungsversuch vorliegt, der dazu führt, dass die Arbeit mit 5,0 bewertet und damit ein Prüfungsversuch verbraucht wird. Abschlussarbeiten dürfen nur einmal wiederholt werden.

Bei einer Thesis des Fachbereichs Architektur entspricht die eingereichte elektronische Fassung dem vorgestellten Modell und den vorgelegten Plänen.

Darmstadt, 2. December 2024

S. Guo

Kurzfassung

Zusammenfassung entsprechend der Dokumentensprache. In diesem Fall Deutsch.

Abstract

Additional abstract in English.

Contents

1 Introduction

Nowadays, electrification of vehicles and autonomous driving have became the most two popular trends of the entire vehicle manufacture industry. Under the encouragement from policies in different country, the battery electric vehicle (BEV) market has exploded in recent years. More and more brand have start the turn their production into electrocution. Back in to the days, traditional motor vehicle use the 12-Voltage platform and the Electronic Control Unit (ECU) to control the whole system running, the main target of the system is to keep the vehicle functional in any circumstance. To enable new functions, such as Adaptive Cruise Control (ACC), the amount of networking within the domains of in traditional vehicle EE system architectures has been increased [5]. This leads to the result, that the need for the accomplish of new functions increase the requirement of electrical power in vehicles [6]. With the respect of this, the electric vehicle platform could brings a lot of advantages, for example the possibility to carry high performance computer, control all components on chassis with a single centralized system etc, which make the vehicle more suitable for the autonomous driving.

With the growing interest of autonomous driving and the support of electric vehicle platform, a new technology with the name of X-by-wire has show up. X-by-wire technology replace the traditional mechanical connection between operator and chassis components with electronic signal. By doing this, the advanced X-by-wire chassis could reduce the time of system reaction and improve the performance and safety of vehicles [7].

Brake-by-wire (BBW) is a part of the X-by-wire chassis. Different from the traditional hydraulic brake system, which the brake paddle connected to a hydraulic cylinder and achieve the control of the hydraulic brake system, the brake paddle in brake-by-wire system doesn't directly drive the brake system, but will generate a control signal to controller and then manipulates the actuator. brake-by-wire systems can be categorized in main types based on their sources and methods of regulating braking force: the electro-hydraulic brake (EHB) system and the electromechanical brake (EMB) system [8].

The EHB system use a electro-pump to drive hydraulic system and generate the clamping force for brakes. Due to the advantages of good compatibility and easy implementation, EHB has become a mature solution in the current market [9]. As the hydraulic fluid system is a main part in EHB systems, the drawback of hydraulic system such as slow braking response, possibility of leaks and the consequence of pollution still existed [10]. Moreover, the elimination of the high cost of hydraulic system always stands for a goal of manufactures.

In EMB systems, motor-generated torque is transmitted directly to the brake's friction components via mechanisms like gear reduction and ball screws, generating frictional braking torque [10]. Because the whole hydraulic system was saved, the disadvantages of EHB systems doesn't existed for EMB. Moreover, from a physics perspective, EMB systems can achieve faster braking responses, which are urgently needed for autonomous driving. In many dangerous scenarios, the autonomous vehicle need to made the decision in millisecond. With the help of EMB systems, it will accomplish higher driving performance and shorter braking distance than experienced human driver. EMS systems also have several other advantages, including reduced system volume and weight, integrated with a parking brake, and compatibility with active safety control systems [11], [12].

With all the advantages above, the fact is that EMB have not show up in the mass production yet. Functional safety concepts is a main reason of it [13]. EMB actuators need to provide a maximum caliper clamping force of 40 kN to provide sufficient braking force [14]. EMB needs to work under all conditions like hot, cold, strong vibration etc, which the normal motor can't functional normally.

To guarantee rapid response in all circumstance, the EMB need to have the ability to be controlled precisely in very short time. Thus, build up an accurate model for EMB is the research hard point for researchers. In order to apply the control task and simulation, a precise model of EMB is necessary. Hence, the estimation of model parameter is very important for the performance and accuracy of EMB models. Form this reason, plenty research of parameter estimation of EMB have been conducted, including nonlinear optimization, xx and xxx **<empty citation>**

With the developing of artificial intelligence (AI) and the growing of computing performance of edge device, more and more AI approaches was used in the vehicle area. Not only the most popular topic in the vehicle industry task like end to end autonomous driving and large language models (LLMs) is involved in AI approaches, in more specific area, AI was also widely used to solve the industry problem including complex and non-linear control for autonomous vehicle, optimal design of motor for electric vehicle and optimization of battery management systems (BMS) [15]–[19]. This leads to the idea that, AI approaches could might also help with searching for the optimal excitation for estimation of EMB parameters, which will be conducted in the following text in detail.

In this thesis, LLMs and translation tools are used to check grammar and make the writing more suitable for scientific purposes [20], [21].

1.1 Motivation

As the EMB could brings a lot of benefits which had already show in early text, there still existing various challenges to be solve. The most critical thing is that autonomous driving system with fast response capability puts high demands on the accuracy of EMB system model. Unlike general systems, the EMB is a dynamic system, and the system characteristics of the EMB will change accordingly with the passage of time or changes in temperature. This poses an even greater challenge to system modelling, as the system parameters determined once using conventional methods cannot be guaranteed to be correct in the long term. In addition to this, traditional methods require the assumption of an a priori value for the system parameters and the optimal design of experiments based on this a priori value [3]. The problem is that this a priori value cannot be determined in advance, and if the a priori value deviates significantly from the actual value, this significantly reduces the performance of the design results.

In contrast to traditional time-consuming parameter estimation methods, this thesis attempts to solve this problem by applying artificial intelligence, more specifically reinforcement learning, to determine the optimal input excitation for parameter estimation. The main aim of this thesis is to investigate two questions, firstly whether reinforcement learning can be used to optimise the best input excitation for parameter estimation, and secondly how reinforcement learning can be designed in such a way that the agent is free from its dependence on the a priori values of the system parameters. For the first problem, an agent with reinforcement learning based on a priori values of system parameters is trained and its performance is tested. For the second problem, a reinforcement learning framework is designed based on the improvement of existing methods and its robustness to non-deterministic parameter a priori values.

Another implication of this thesis is that its approach is not only limited to apply to EMB systems. More precisely, this thesis attempts to develop a more general approach for optimising input excitations for parameter discrimination. In this paper, EMB is used more as a platform to apply and test the method. Therefore another motivation for this topic is that, with simple modifications, the method in this thesis can also be used in other systems to optimise the input excitation for parameter discrimination.

To do the estimation, the conventional approach is do different test for different parameter. In this thesis, approaches to do the estimation of EMB model parameter was developed.

1.2 Structure of the Thesis

Chapter 2 describes the current state of development of optimal experimental design and parameter estimation methods, comparing their application in the field of EMB and the application of reinforcement learning methods in optimal input exaction.

In Chapter 3, the modelling of the electromechanical braking system is described and a simplified model of the EMB used in this thesis is presented.

Chapter 4 presents the methods for applying reinforcement learning to the optimizing experimental design in EMB, with detailed explanations of reinforcement learning framework formulation, parameter estimation, and environment setup.

Chapter 5 describes the application of the reinforcement learning framework to single-parameter optimal input excitation, where training results for reinforcement learning are shown and the performance of two observation space structures is compared.

Chapter 6 extends the application to the multi-parameter domain, describing the multi-parameter approach and presenting the training results.

Chapter 7 summarises the content of this thesis and provides an outlook of future works.

2 State of the Art

Papers on general optimal design Papers on parameter estimation and its use in EMB Papers on RL and some RL approaches already used for optimal input exaction and parameter estimation.

Compare the pros and cons of existing approaches for optimal input exaction and parameter estimation and make a clear view in the table. Came to the clue of the necessities of this thesis and which problem or drawback form the existing research are overcome in this thesis.

3 Modelling of the Electromechanical Brake System

To have the idea of the EMB system, a mathematics model is necessary. Talking with the parameter estimation, the way of modelling have direct affect of the performance of the modelling. For a functional part with high quality control task and rapid response ability like EMB, the model of the system should be as precise as possible. As EMB is a complex system, this leads to the high level of parameters to be consider of during the modelling.

Nowadays, a lot of researches have been done in electromechanical brake, but mostly focus on the disc EMB system. Compare with the disc brakes, drum brakes is not wildly used for vehicles in recent years. The reason of it including, drum brakes have lower performance facing the cooling problem and the traditional respect from customer that considering drum brakes is cheaper etc. However, drum brakes still have advantages under the more and more restrict politic restriction, especially concern about the lower piratical emission because of the closed drum [22]. Thus manufacture like Bosch start to do research on electromechanical drum brake again and in this thesis, research was conducted based on a electromechanical drum brake system from Bosch.

In this chapter, the structure of the to be modelled EMB system will be introduced first. Then, different approaches to do the modelling of the EMB system will be introduced. Later the key parameters in different point of view for the modelling of EMB will be described. Finally the simplify of the EMB model for the later use will be conducted.

The subject of different degrees of accuracy when modelling different parts of hydraulic systems has been well described by several authors, where only a simplified modelling is usually applied [23].

3.1 Structure of the Electromechanical Drum Brake

As show in Figure 3.1, the electromechanical drum brake to analysed have the following components: the brake shoes, the brake lining, return springs and the brake drum. When the ECU receives a braking signal, it will activates the electro motor, which drives the actuator and push brake shoes and brake pads against the drum. When braking is released, the motor reverses, and a spring returns the brake shoes and brake pads to its original position [24], [25].

Consider of the structure of the electromechanical drum brake, several force needs to take into consideration during the modelling. First of all, the propose of the break is to make the wheel speed lower. When the actuator force the brake shoes move against the brake drum, the drake lining will have contact with the drum and the clamping force will show up, finally this clamping force will transformed in to brake moment. Secondly, the force form the return springs is also a main part in the model. Another big part of force which need take in to consideration is the force from the actuator, namely the electro motor.

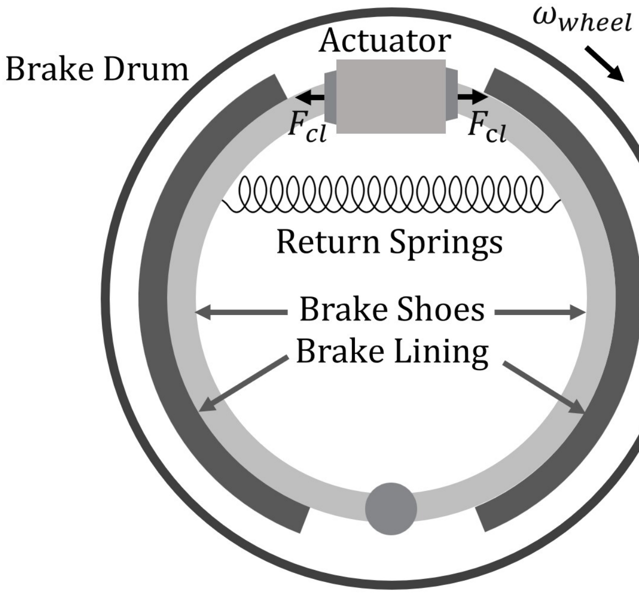

Figure 3.1: Mechanical structure of the electromechanical drum brake to be analyzed [26]

There is still a kind of torque which can't neglected, that is the friction torque. The friction torque is everywhere, from inside the motor to the actuator. In following text, these torque will be deal in detail.

3.2 Lumped parameter EMB Model

In this section, all the torque which have been mention in the above text will be treat carefully in the following text. As we can already seen, that torques happens all over the EMB, which make it difficult for the modelling. According to researches, the lumped parameter EMB model always been used [1]. In this thesis, three main torque are considered in the lumped parameter EMB Model: the motor torque T_m, the load torque T_l and the friction torque T_f. Consider of the dynamic behavior of the motor, the EMB have been modeled as:

$$J\ddot{\theta} = T_m - T_f - T_l \tag{3.1}$$

The motor torque T_m serves to accelerate the brake shoes, while the load torque T_l and frictional torque T_f act in opposition, resisting this acceleration. Assuming a linear relationship, the electromagnetic motor torque T_m cloud be regard as directly proportional to the motor torque constant k_m and the motor current i_m with the equation

$$T_m = k_m i_m \tag{3.2}$$

The load torque T_l is proportional to the clamping force F_{cl} and the overall gear ratio γ

$$T_l = \gamma F_{cl} \tag{3.3}$$

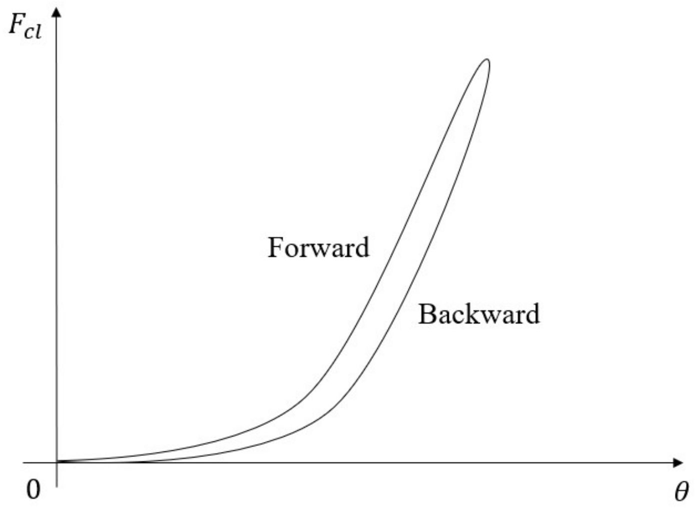

Figure 3.2: Qualitative hysteresis characteristics between clamping force and motor angle

The gear ratio γ represents the characteristics of the entire mechanical system, transforming the force from the braking shoe side to torque on the motor side. The friction torque T_f could be regard as a equation which included the motor angle θ and the motor velocity $\dot{\theta}$. The detail of calculation of clamping force F_{cl} and friction torque T_l will be conducted in the following subsection.

3.2.1 Stiffness Model

The EMB is a complex nonlinear system, and its nonlinear properties are reflected in the stiffness model. The characteristic curve, also known as the stiffness of the EMB system, describes the relationship between the motor angle θ and the clamping force F_{cl}. The clamping force is quantified in relation to the drive motor through the utilisation of force sensors positioned on the EMB. As illustrated in Figure 3.2, the stiffness curve is nonlinear. It is assumed that 0 degrees represents the point of contact between the brake pad and the brake drum. The brake force stiffness curve can be divided into two intervals. During the initial range, the brake force grows at a slow rate. However, from later on range, the clamping force grows nearly exponentially with angle. The stiffness model of the EMB exhibits hysteresis curve characteristics, as outlined in the referenced literature [27]. In comparison to the process in which the brake pads leave the brake drum due to the action of the reset spring, when the brake pads are actuated by the actuator and move towards the brake drum, the curve can reflect a higher stiffness. Furthermore, the hysteresis characteristics are also dependent on factors such as speed and temperature. Given that the parameter estimation process generally focuses more on the braking process than the release process, the hysteresis characteristics are ignored in this model to maintain a simplified representation.

3.2.2 Friction Model

For a long time, friction has been intensively studied in physical systems. The classic model of friction is static model, it including Coulomb friction F_c and viscous friction F_v. By applying this classic model to

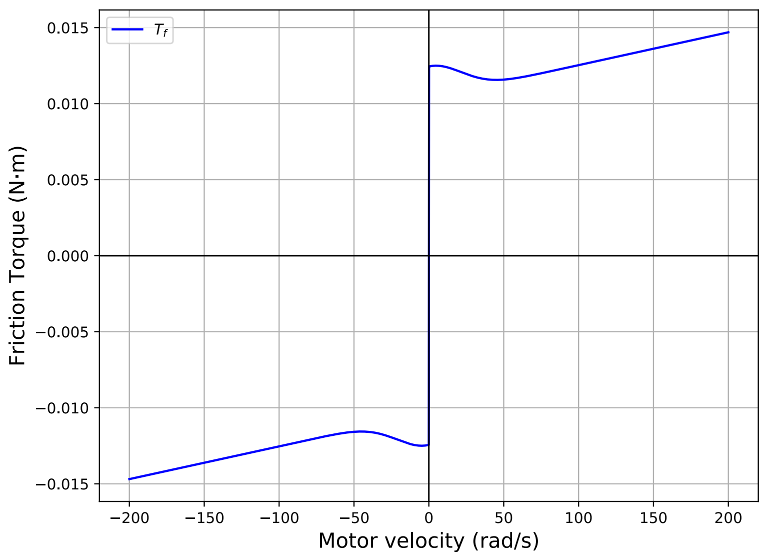

Figure 3.3: Illustration of the Stribeck friction model

the EMB system, the friction torque equal to the summation of these two frictions torque

$$T_f = T_c\,\mathrm{sign}(\dot{\theta}) + f_v\dot{\theta} \tag{3.4}$$

T_c is the friction torque, f_v is the viscous friction torque parameter and $\dot{\theta}$ is the motor velocity.

The drawback in equation (3.4) is, in this static model, the static friction which higher than the Coulomb friction is not included. Stribeck observed a continues dependence for static friction and dynamic friction [28]. One commonly used model for the non-linearity with Stribeck behavior considered in EMB is expressed as

$$T_f = T_c + (T_s - T_c)e^{-|\dot{\theta}/\dot{\theta}_\sigma|^{\delta_\sigma}} + f_v v \tag{3.5}$$

where $\dot{\theta}_\sigma$ represents the Stribeck velocity. The friction model with Stribeck effect is show in Figure 3.3 This type of model has been widely used for a long period. By holding the velocity, the friction function can be easily determined by measuring the friction force.

In EMB modelling, an additional friction component, which is dependent on the load, could also be considered. This is modeled as a linear function of the clamping force F_{cl}, with f_l as the proportionality coefficient

$$T_s = T_{s0} + f_l F_{cl} \tag{3.6}$$

One of the limitations of the aforementioned model is the necessity to ascertain the point at which the motor velocity is at its lowest. Karnopp discovered that this issue can be circumvented by incorporating a small zero-speed interval θ into the model and preventing the motor from repeatedly alternating between positive and negative equations of motion [29]. In accordance with the aforementioned model, the friction force may be either arbitrary or defined, contingent upon whether the speed is less than the low-speed interval threshold. With this theory, the EMB system friction could be built as

$$T_f = \begin{cases} T_c \operatorname{sign}(\dot{\theta}) + f_v\dot{\theta} & \text{if } |\dot{\theta}| > \varepsilon \\ T_e & \text{if } |\dot{\theta}| < \varepsilon \text{ and } T_e < T_s \\ T_s & \text{otherwise} \end{cases} \tag{3.7}$$

T_e stands for the external torque and T_f stands for the load-independent static friction torque. When the motor velocity with in the zero-speed interval and the external torque smaller than the static friction torque, the friction torque will be equal to the external torque, as the system could be seen as don't move.

However, this model is not without its limitations. When the motor velocity stay in the zero-speed interval and the external torque smaller than the static friction torque, the friction force is not explicitly given, rendering this model unsuitable for cases where the friction force must be derived.

Olsson et al.[30] said many friction phenomena do not show up in constant velocity experiments. Thus, in addition to static friction modeling systems, dynamic friction modeling has also been studied. Given the high precision of the EMB, the dynamic friction model is more complex and requires detailed modeling of the system to analyze the dynamic properties of the system. Consequently, the dynamic friction model is also considered in the modeling.

There is dynamic friction model as LuGre model [31]. The LuGre model interprets friction as the deflection of tiny bristles between surfaces. When a force is applied, these bristles bend like springs, and if they bend too much, they start to slip. In steady-state motion, the average deflection depends on speed, becoming smaller at higher velocities. This reduction in friction at higher speeds represents the Stribeck effect. The model also includes rate-dependent behaviors, capturing dynamic effects like frictional lag.

But this model is often too complex and need to be simplify. This so called steady-state approximation allows us to focus on the frictional behavior after the dynamic transients have subsided, effectively making the model act quasi-static. LuGre model in static could be described as

$$\begin{aligned} T_f &= \sigma_0 g(\dot{\theta}) \operatorname{sign}(\dot{\theta}) + \sigma_2\dot{\theta} \\ &= \left(T_c + (T_s - T_c)e^{-\left(\frac{\dot{\theta}}{\omega_s}\right)^2}\right) \operatorname{sign}(\dot{\theta}) + f_v\dot{\theta} \end{aligned} \tag{3.8}$$

As the equation showed, the LuGre model in static form will be the same as it has been shown in equation (3.5), with the static friction model and the Stribeck effect been considered. This show the unify of different ways of friction modelling.

3.3 Simplify of the Model of Electromechanical brake System

The preceding section provides a comprehensive account of the various factors that must be taken into account when constructing a nonlinear EMB system. In theory, the more detailed the modeling of a system, the more accurately it reflects the actual performance of the physical system. Nevertheless, in practice, the most detailed model is not always necessary. In the majority of cases, the model must be

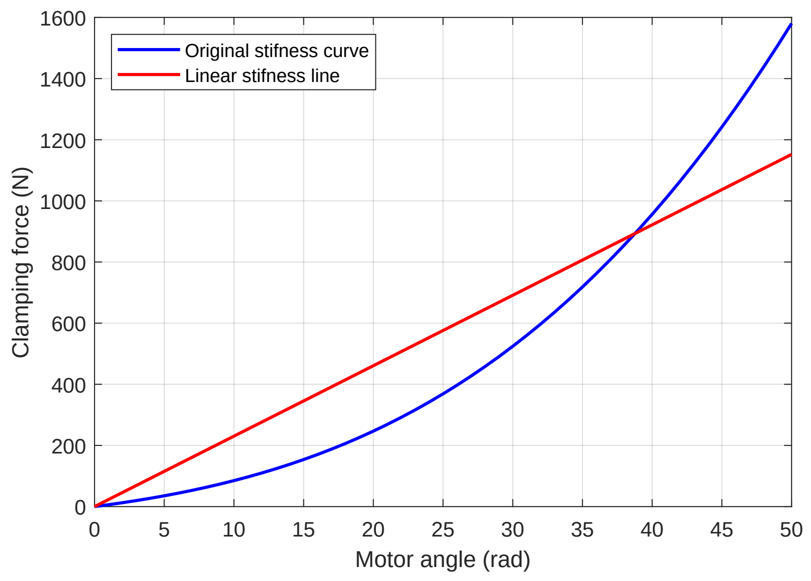

Figure 3.4: Comparison of original and linearized stiffness curves for clamping Force

simplified in order to facilitate computation. This thesis investigates the potential for optimal input excitation of reinforcement learning methods for parameter estimation. In theory, each parameter has an optimal excitation, and overly complex EMB models will have a greater number of parameters that need to be recognized. This will complicate the problem that reinforcement learning needs to solve at the outset. Furthermore, this thesis is primarily a research project on EMB systems, and its primary objective is not to solve the engineering problems of actual EMB systems. Consequently, the corresponding simplifications of the models for each of the aforementioned parts were carried out.

3.3.1 Simplify of the Stiffness Model

For the stiffness model, a polynomial is generally fitted to the experimentally collected relationship between motor angle and clamping force, and the fitted result is mostly a cubic curve

$$F_{cl} = k_1\theta^3 + k_2\theta^2 + k_3\theta \tag{3.9}$$

Figure 3.4 illustrates the real curve representing the relationship between the motor angle and the clamping force, based on data collected from an established and well-validated electromechanical drum brake model experiment. Introducing three stiffness parameters, $k_1 - k_3$, would greatly increase the dimensional complexity of the reinforcement learning problem, necessitating parameter estimation for each of these components. To mitigate this issue, a linear interpolation method is used in the modeling process. The linear clamping force is represented by

$$F_{cl} = k_1\theta \tag{3.10}$$

Although using linear interpolation increases the deviation between the system model and the real system behavior, it reduces the three stiffness parameter dimensions to one parameter dimension, which helps to reduce the learning difficulty of the reinforcement learning agent.

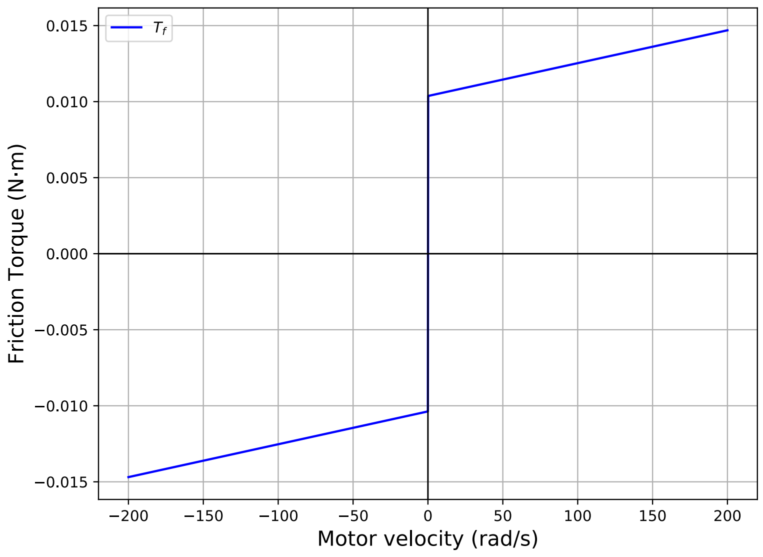

Figure 3.5: Illustration of the simplified friction model

To reduce the dimension level of the stiffness parameter, data from validated EMB model was imported into $Matlab$, and the $CurveFittingToolbox$ was used to derive the linear stiffness line (red line) through linear interpolation. The resulting red and blue curves in Figure 3.4 demonstrate the original stiffness response and the simplified linear approximation, respectively. The slope of the linear curve corresponds to the stiffness parameter k_1.

3.3.2 Simplify of the Friction Model

The modelling of friction model have talk a lot in the before pages. It can be observed that complex friction models bring high accuracy while likewise adding more parameter dimensions. Taking the Stribeck model as an example, the modeling session introduces more parameters such as Stribeck velocity according to equation (3.5). When the load-dependent static friction torque is considered, as showed in equation (3.6), it will not only brings other parameters like T_{s0} and f_l, also make the friction model relevant to the stiffness model, which will make the analyse of the whole EMB model even more complicated.

To limit the dimensional of the parameters in the reinforcement learning process, the primary consideration when modeling friction is to ensure the simplicity of the model. Consequently, the classical friction model, as presented in equation (3.4), is adopted due to its straightforward representation. Figure 3.5 illustrates the simplified friction model used in this study.

3.3.3 Summary of the EMB Model

After all the simplify of the early pages, finally a simplified lumped parameter nonlinear EMB model was introduced and showed in the state space:

$$\dot{x} = f(x, u), \quad y = h(x) \tag{3.11}$$

where $x = [x_1, x_2]^T$ is the state with the motor position x_1 and the motor velocity x_2 and the u is the motor current as the input of the system. The simplified nonlinear function is considered as:

$$f(x,u) = \begin{bmatrix} x_2 \\ \frac{1}{J}\left(k_m u - \gamma F_{cl} - T_f\right) \end{bmatrix} \tag{3.12}$$

where J is the lumped inertia of the EMB system, k_m is the motor torque constant, γ is the gear ratio, F_{cl} is the clamp force, T_f is the friction torque. In this simplified lumped parameter EMB model, we firstly consider the stiffness characteristics of the clamp force is linear function:

$$F_{cl}(x_1) = \begin{cases} k_1 x_1 & \text{if } x_1 \geq 0, \\ 0 & \text{if } x_1 < 0. \end{cases} \tag{3.13}$$

where k_1 is the stiffness characteristics parameter, which is not totally true but it is simple for the calculation. The friction model here only take coulomb and viscous friction in to consideration, given by:

$$T_f(x_2) = T_c \operatorname{sign}(x_2) + f_v x_2 \tag{3.14}$$

where T_c and f_v are the coulomb friction torque and viscous friction coefficients.

The system output also the observation is the motor position x_1 and the motor velocity x_2, hence:

$$h(x) = [x_1, x_2]^T \tag{3.15}$$

4 Methodology for applying reinforcement learning to optimize input excitation in EMB Systems

Following the development of the EMB model in Chapter 3, an investigation was initiated into the input excitation to this model. As demonstrated in Chapter 2, the optimization problem associated with identifying the input excitation of the system parameters has been extensively studied. In this process, a variety of methods, including MPC and RL, are employed. However, these methods are not without limitations. In particular, reinforcement learning methods have been applied only in localized areas and still have significant shortcomings. This chapter will highlight the methodology, propose a new reinforcement learning framework for solving optimal input excitation for system parameter estimation, and describe the different theoretical methods used in this structure from the beginning to the end. It will also describe how to perform the optimal experimental design for the above EMB model, how to combine it with reinforcement learning, and the theoretical methods used in this process.

4.1 Formulation the EMB model for the optimization of input excitation

Consider this model will be used in later research with reinforcement learning, to prevent unrealistic situation from happening, constraints must also be set. The first consideration is the input of the EMB model, which is contingent upon the technical parameters of the motor. It can be deduced that the input voltage of the motor is limited to a range of -6V to 6V. Secondly, the range of motion of the motor is constrained by the physical limitations of the system. The motor is unable to rotate at an infinite angle; therefore, the rotation angle of the motor is set to be between -10 rad and 100 rad. Meanwhile, according to the The test data obtained from the EMB model in the laboratory indicates that the maximum rotational speed of the motor is 500 rad/s. Over speed of the motor will not only leads to greater friction, generating excessive heat, but also decrease the control precision of the system. The motor may behave too fast for the response controller, leading to braking force errors and potentially compromising vehicle safety and comfort. Consequently, the motor rotational speed has been set to a range between -500 rad/s and 500 rad/s.

As motor velocity sensors are employed in the actual parameter estimation process, the measurements of the sensors are discrete sequences. Consequently, the continuous state space equations mentioned above must be discredited. There are numerous methods of discrediting the system, including Euler's method, RK-4 method, and so forth. However, to ensure the optimal simplicity of the model, Euler's method is utilized in this thesis

$$x_d(k+1) = x_d(k) + t_s f(x_d(k), u(k)) \tag{4.1}$$

where t_s is the time step. Since the sampling frequency of the sensor used in the experimental platform is 1000 Hz, t_s is determined to be 0.001s in this model accordingly.

Furthermore, the sensor noise utilized to quantify the system's input and output must be considered during the model's construction, as it will directly influence the accuracy of the parameter estimation. In accordance with the specifications of the actual task, the EMB model investigated in this thesis is equipped with a high-precision angle sensor, but lacks a speed sensor. The motor velocity is provided by a kind of state observer, which introduces a relative higher degree of error. The sensor noise is typically modeled as Gaussian noise, which obeys a Gaussian distribution $v_o \sim \mathcal{N}(0, R)$ with mean 0 and variance R. For the angle sensor, the variance of the Gaussian noise is 1×10^{-6}. For the state observer used to observe the motor velocity, the variance of the Gaussian noise is 1. The input signal is presumed to be entirely free from nose, leading to the assumption that $v_i = 0$. After the discrediting the system and considering the input noise, the EMB model in the subsection 3.3.3 will turn to be like

$$\begin{aligned} x_d(k+1) &= f_d(x_d(k), u(k)) \\ y_d(k) &= h_d(x_d(k)) \\ y_m(k) &= h_d(x_d(k)) + v_o(k), \quad k = 1, ..., n \end{aligned} \tag{4.2}$$

where the state variable $x_d(k) = [x_1(k), x_2(k)]^T$ represents the motor position and motor velocity at time step k. Here, denotes the system output $y_d(k)$ at time step k, while corresponds to the measured output $y_m(k)$ at the same time step.

In conclusion, a streamlined discrete EMB model has been developed to facilitate optimal input excitation for the investigation of reinforcement learning and parameter estimation. Based on the conventional parameter estimation approach and the simulation model outcomes, parameters that have been included in the EMB system has been identified in the table 4.1.

In these parameters, the lumped inertia of the EMB system J and total gear ratio γ could be seen as fixed and provide by the producer, so not included in the parameter estimation. The motor constant k_m and the stiffness parameter k_1 will change over the use of EMB, so it is two parameter to be estimated. The friction parameters are also critical for the EMB system, as they directly affect braking performance. These parameters can be influenced by factors such as temperature, wear, and contamination, which can alter the system's response and reliability. So at the end, the to estimated EMB parameters θ have 4 dimensional

$$\theta = [k_m, k_1, T_c, f_v]^T \tag{4.3}$$

4.2 Optimal experimental design and Fisher information matrix

Optimal experimental design is an approach to design a input trajectory to excited the system and inferring the parameter from the output trajectory. In this section, the formulation of optimal experimental design (OED) in the context of the EMB model was discussed, focusing on concepts such as Fisher Information Matrix (FIM) and sensitivity analysis.

Optimal Experimental Design is a methodology used to design an input trajectory that effectively excites the system, enabling accurate inference of system parameters from the resulting output trajectory. The

Parameter	Value	Unit
	Motor	
J	4.624×10^{-6}	kg·m^2
k_m	2.170×10^{-2}	[illegible]/rad
γ	1.889×10^{-5}	
	Stiffness	
k_1	2.304×10^{1}	N/rad
	Friction	
T_C	1.037×10^{-2}	N·m
f_v	2.160×10^{-5}	[illegible]/rad
	Constraints	
u_{max}	6	V
u_{min}	-6	V
$x_{1,max}$	100	rad
$x_{1,min}$	-10	rad
$x_{2,max}$	500	rad/s
$x_{2,min}$	-500	rad/s
	Noise	
R_{x_1}	1×10^{-6}	
R_{x_2}	1	
	Sampling time	
t_s	1×10^{-3}	s

Table 4.1: Parameters for the Simplified EMB Model

goal of OED is to maximize the information obtained from experiments, and a key metric for assessing this information is the Fisher Information Matrix (FIM), FIM takes the form [32]

$$\begin{aligned} M(t) &= \int_0^t \left(\frac{\partial \mathbf{h}}{\partial \boldsymbol{\theta}}\right)^\top R^{-1} \left(\frac{\partial \mathbf{h}}{\partial \boldsymbol{\theta}}\right) dt \\ &= \int_0^t \left(\frac{\partial \mathbf{h}}{\partial \mathbf{x}} \frac{\partial \mathbf{x}(t)}{\partial \boldsymbol{\theta}}\right)^\top R^{-1} \left(\frac{\partial \mathbf{h}}{\partial \mathbf{x}} \frac{\partial \mathbf{x}(t)}{\partial \boldsymbol{\theta}}\right) dt \end{aligned} \tag{4.4}$$

In equation (4.4), the part $\frac{\partial \mathbf{h}}{\partial \boldsymbol{\theta}}$ is called sensitivity. It reflects the relationship between system outputs and the unknown parameters. Sensitivity analysis determines how changes in the input trajectory affect the system output with respect to the parameters. If a particular input trajectory causes significant changes in the output with respect to the parameters, it implies that the parameters are highly sensitive to this input, suggesting that the input trajectory contains a high level of information about the parameters. Thus, sensitivity analysis is essential for evaluating how effective a given input is for parameter estimation.

Furthermore, FIM incorporates the effect of measurement noise. As previously stated, measurement noise was seen as Gaussian noise with a mean value of zero. In multiple-input and multiple-output (MIMO) systems, the greater the noise in the output measurements of a given dimension, the more inaccurate the measurements become. This results in a reduction in the information content of that dimension's outputs generated in the parameter estimation process. The inverse matrix of the output noise variance matrix R^{-1} is included in equation (4.4). This form amplifies the amount of information contained in the output dimension with measurements variance less than 1 and reduces the amount of information contained in the output dimension with measurements variance greater than 1.

The Fisher Information Matrix (FIM) measures the amount of information available in the output for estimating the system parameters. It quantifies how much the system output changes in response to variations in the parameters, thereby indicating the quality of the parameter estimation. Under the assumption that parameter estimates are unbiased and that measurement noise is uncorrelated, the inverse of the FIM provides an approximation of the Cramér-Rao lower bound [32]. This bound represents the theoretical lower limit for the variance of unbiased estimators, closely related to the variance-covariance matrix of the estimated parameters. By maximizing some criteria of the FIM, the lower bound is minimized , thus let the system have the possibility to get a lower variance of unbiased estimators, which means improving the precision of parameter estimation. Therefore, when designing the input excitation trajectory, criteria for input excitation are often evaluated using scalar measures of the FIM M and the inverse matrix of FIM M^{-1}, including [33]:

1) D-optimality: Minimizes the determinant of inverse of the FIM $|M^{-1}|$, ensuring that the volume of the confidence ellipsoid for parameter estimates is minimized. Also, the convex function $log|M^{-1}|$ impels the mutual information between system output and parameters [34].

2) A-optimality: Minimizes the trace of the inverse of the FIM $tr(M^{-1})$, reducing the average variance of parameter estimates.

3) E-optimality: Minimizes the biggest eigenvalue of inverse of the FIM $\lambda_{max}(M^{-1})$, minimizing the length of the largest uncertainty axis within the joint confidence region of the parameters, effectively reducing the largest possible parameter errors.

Suppose a system model has two parameters that need to be identified, according to Equation (4.4), FIM M is a i-positive definite symmetric matrix, so the inverse matrix of FIM p take the form

$$M^{-1} = \begin{bmatrix} m_{11} & m_{12} \\ m_{12} & m_{22} \end{bmatrix} \tag{4.5}$$

The estimates $\hat{\theta}_1$ and $\hat{\theta}_2$ are the unbiased estimators for the parameters. The elements m_{11} and m_{22}represent the diagonal entries of the matrix M^{-1}, while its eigenvalues are λ_1 and λ_2. Figure 4.1 intuitively shows an example for these three optimality with this parameters in two dimensions. The blue-shaded area in the figure represents the area of the confidence ellipse. Given that the two parameters constitute a two-dimensional planar space, the D-optimality is reflected in the minimization of the ellipse area. The semi-axis lengths of the two axes of the confidence ellipse correspond to the two eigenvectors of the FIM inverse matrix. Therefore, the E-optimality is reflected in the minimization of the scales of the axes of the confidence intervals. The A-optimality is aimed at minimizing the trace, i.e., $m_{11} + m_{22}$, of the FIM inverse matrix. This is equivalent to the overall reduction of the uncertainty in all directions.

In summary, OED involves designing inputs that maximize the information available for parameter estimation. The FIM is a fundamental tool in this process, the design of optimal input trajectory based on optimality criteria, while sensitivity analysis helps determine the effectiveness of these inputs in estimating the system parameters accurately.

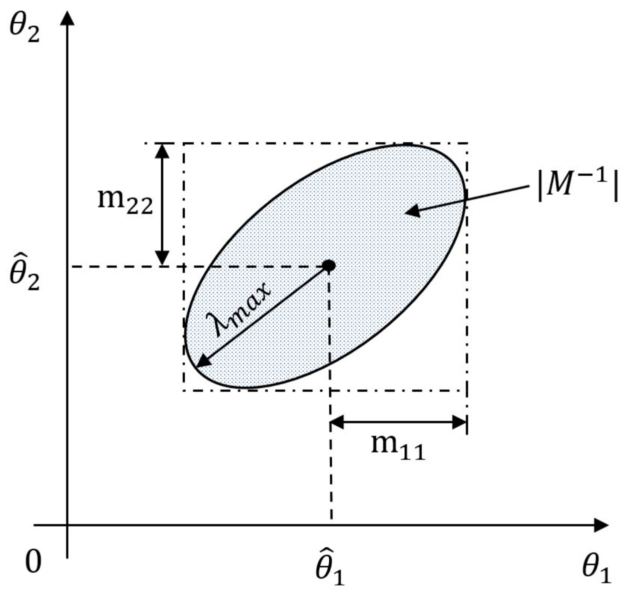

Figure 4.1: Confidence ellipsoid for parameters in two dimensions based on paper [33]

4.3 Formulation of the optimal experimental design in EMB system

In this section, the formulation of an optimal input excitation for a time-discrete EMB system is presented, incorporating the Fisher Information Matrix into an optimization problem. Assume the existence of a time-discrete EMB model, as described in Section 4.1 and Equation (4.2). Given a trajectory U over n steps, the objective is to determine the optimal input trajectory by solving the following optimization problem

$$\min_{U} \begin{cases} |M^{-1}| \text{ or } \log|M^{-1}|, & \text{for D-optimality,} \\ \text{tr}(M^{-1}), & \text{for A-optimality,} \\ \lambda_{\text{max}}(M^{-1}), & \text{for E-optimality.} \end{cases} \tag{4.6}$$

Here, M is the Fisher Information Matrix, and the optimization criteria (D-, A-, or E-optimality) are chosen depending on the desired objective: maximizing determinant-based informativeness, minimizing overall parameter uncertainty, or reducing the worst-case uncertainty.

The optimization is subject to the following boundary conditions and constraints

$$\begin{aligned} & x_d(1) = x_0, \\ & x_{1,min} \leq x_1 \leq x_{1,max}, \\ & x_{2,min} \leq x_2 \leq x_{2,max}, \\ & u_{min} \leq u \leq u_{max}, \\ & k \leq t_{max} \end{aligned} \tag{4.7}$$

where x_d is the vector of state variables, u is the system input trajectory, k is the time step and t_{max} is the maximum number of steps defined for the parameter estimation process. The constraints ensure that the system operates within physical and operational limits, such as state bounds, input saturation and time restriction.

In the optimal experimental design specific to the EMB system, further detailed constraints were considered. According to the requirement of experiment, the EMB motor will start from a zero position and velocity. The experiment will last for 500 ms, and at the end of this period, the motor should return to the zero position. These experimental setup requirements will be implemented later during the construction of the EMB environment using the Gymnasium framework and the design of the reward function for reinforcement learning.

4.3.1 Reformulating the optimization problem for reinforcement learning

In reinforcement learning, the goal is to train an agent to maximize the return over an episode. Therefore, the minimization problem in Equation (4.7) cannot directly serve as the return for the RL agent. To solve the excitation design task, this optimization problem is reformulated into a framework that can combine with reinforcement learning methods. One intuitive approach is to simply apply a negative sign to the objective function to convert it into a maximization problem. While this approach is feasible, there is still another problem that exists.

Since the FIM is changing over time during the parameter estimation experiment, as shown in Equation (4.4), the objective function still requires calculating the inverse of the FIM, M^{-1}, at each time step. If the dimension of the matrix M is large, this significantly increases computational difficulty and causes longer computation times. Therefore, the objective function for the optimization problem needs to be modified to avoid matrix inversion operations when practically applied to reinforcement learning. The approach for this modification will be presented below.

Suppose that the FIM is an n-dimensional nonsingular matrix with eigenvalues λ that satisfy the following relationship

$$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n \tag{4.8}$$

the determinant of the FIM can be expressed as

$$|M| = \prod_{k=1}^{n} \lambda_k, \tag{4.9}$$

and the determinant of the inverse of the FIM is given by

$$|M^{-1}| = \prod_{k=1}^{n} \frac{1}{\lambda_k} = \frac{1}{|M|}. \tag{4.10}$$

When the D-optimality criterion with logarithm is applied, the object function becomes

$$\log|M^{-1}| = \log|\frac{1}{M}| = -\log|M| \tag{4.11}$$

With this approach, the optimization problem for the D-optimality criterion is transformed into a maximization problem, avoid to calculate the inverse matrix at each time step. Hence, the total return G_n in the reinforcement learning framework after n steps is

$$G_n = \sum_{k=1}^{n} r_k = log|M_n| \tag{4.12}$$

From this equation, it is intuitive to define the step reward r_k for the agent during training as

$$r_k = log|M_k| - log|M_{k-1}| \tag{4.13}$$

Similarly, under the E-optimality criterion, the objective becomes

$$\min \lambda_{\text{max}}(M^{-1}) \rightarrow \max_{U} \lambda_{\text{min}}(M) \tag{4.14}$$

Thus the step reward for reinforcement learning based on the E-optimality is given by

$$r_k = \lambda_{\text{min}}(M_k) - \lambda_{\text{min}}(M_{k-1}) \tag{4.15}$$

However, for the A-optimality criterion, which involves the trace of the inverse FIM

$$\text{tr}(M^{-1}) = \sum_{k=1}^{n} \frac{1}{\lambda_k} \neq \frac{1}{\text{tr}(M)}, \tag{4.16}$$

Hence the calculation of the inverse FIM is still necessary for the A-optimality criterion. Therefore, due to the consideration of the simplicity of computation, D-optimality and E-optimality criteria are primarily considered in this thesis.

4.3.2 Normalization of parameters

The advantage of D-optimality over the other two optimality criteria is its independence from the numerical scales of the parameters. In the model studied in this thesis, as shown in Table 4.1, the magnitude of the a priori values of each parameter ranges from 10^{-5} to 10^{1}. It can be concluded from the calculation of sensitivity that the system output is generally more sensitive to parameters with smaller numerical scales. Since significantly value difference could existed between different parameters, in order to keep the variance of the different parameters comparable, the parameters need to be normalized when applying the other two optimality criteria. The normalised parameter vector for a n dimensional parameter vector could be written as:

$$\bar{\theta} = \left[\frac{\theta_1}{\theta_{nom,1}}, \frac{\theta_2}{\theta_{nom,2}}, \ldots, \frac{\theta_n}{\theta_{nom,n}}\right]^T \tag{4.17}$$

4.3.3 Normalization of outputs

Since sensitivity depends on the magnitude of the change in outputs with respect to parameter variations, different output scales in a multi-output system can significantly influence the calculation of FIM and sensitivity. For instance, if some parameters in a system only affect specific output dimensions, and the range of these output dimensions is significantly larger than others, the corresponding dimension will contribute larger values to the FIM. This implies that the input excitation will yield higher information content for the parameters affecting those output dimensions. Therefore, to ensure balanced sensitivity and an accurate representation of information content, not only do the parameters need to be normalized, but the output scales should also be normalized.

For a system with p output dimensions, represented as $y = h(x) = [y_1, \ldots, y_p]$, normalization can be applied as follows:

$$\bar{h} = D_y^{-1} h(x) \tag{4.18}$$

where D_y is a diagonal matrix defined as:

$$D_y = diag([y_{1,max}, y_{2,max}, \ldots, y_{p,max}]) \tag{4.19}$$

This normalization ensures that all outputs are brought to a comparable scale, thereby preventing any single output dimension with a larger range from disproportionately influencing the FIM. By normalizing the outputs in this way, a more balanced sensitivity analysis is achieved, which helps in obtaining an accurate estimation of the parameters and ensures that the information content from all output dimensions is represented equitably.

4.3.4 Calculation of the Fisher information matrix with discrete system

As described in the Equation (4.4), the calculation of FIM involves the sensitivity $\frac{\partial h}{\partial \theta}$. Since the EMB model is discretized using Euler's method, the equation (4.4) must also be transferred into discretized version. Consider the simplified lumped parameter nonlinear time-discrete EMB model with measurement noise as showed in (4.2)

$$\begin{aligned} x_d(k+1) &= f_d(x_d(k), u(k)) \\ y_m(k) &= h_d(x_d(k)) + v_o(k), \quad k = 1, ..., n \end{aligned} \tag{4.20}$$

The Fisher information matrix with n-dimensional parameters after the normalization of parameters and outputs, can be calculated using the following formulation [2]

$$\begin{aligned} M &= \sum_{k=1}^{n} \left(\frac{d\bar{h}_k}{d\bar{\theta}} \right)^T \text{[illegible]}^{-1} \left(\frac{d\bar{h}_k}{d\bar{\theta}} \right) = \sum_{k=1}^{n} (M_{\bar{\theta},k}), \\ \frac{d\bar{h}_k}{d\bar{\theta}} &= \begin{bmatrix} \frac{d\bar{h}_k}{d\bar{\theta}_1} & \frac{d\bar{h}_k}{d\bar{\theta}_2} & \ldots & \frac{d\bar{h}_k}{d\bar{\theta}_n} \end{bmatrix}, \end{aligned} \tag{4.21}$$

with the time steps k written in the subscripts. The sensitivity of each output with respect to each parameter is obtained as

$$\frac{d\bar{h}_k}{d\bar{\theta}_i} = \frac{\partial \bar{h}_k}{\partial x_d^T} \frac{\partial x_{d,k}}{\partial \bar{\theta}_i} \text{[illegible]} \frac{\partial \bar{h}_k}{\partial \bar{\theta}_i} \tag{4.22}$$

Here, $\frac{\partial x_{d,k}^T}{\partial \theta}$ represents the sensitivity of each state variable with respect to each parameter. In the continuous domain, this sensitivity can be calculated by integrating[4]

$$\frac{d}{dt} \frac{\partial x}{\partial \theta}(t) = \frac{\partial f}{\partial x} \frac{\partial x}{\partial \theta}(t) + \frac{\partial f}{\partial \theta} \tag{4.23}$$

with initial condition

$$\frac{\partial x}{\partial \theta}(0) = \frac{\partial x_0}{\partial \theta} \tag{4.24}$$

Transfer this function into the discrete domain, it turns int

$$\frac{\partial x_{d,k+1}}{\partial \bar{\theta}_i} = \frac{\partial f_{d,k}}{\partial x_d^T} \frac{\partial x_{d,k}}{\partial \bar{\theta}_i} + \frac{\partial f_{d,k}}{\partial \bar{\theta}_i}, \tag{4.25}$$

With the consideration of no sensitivity or information is present at the beginning of the parameter estimation experiment, the initial condition of the calculation is given by

$$\frac{\partial x_{d,1}}{\partial \theta} = 0 \tag{4.26}$$

Using this approach, the Fisher information matrix in the discrete system is computed iteratively at each time step, and the final FIM is obtained as the number of total time steps n reaches.

4.3.5 Implementation of the Calculation of the Fisher information matrix in Python

After completing the theoretical formulation of the calculation of FIM, the corresponding model in Python is implemented. Since the equations in the before section basically always calculate the gradient, automatic differentiation tools are suitable for this task. This implementation utilizes PyTorch's Autograd feature for automatic differentiation [35], which simplifies calculations of the FIM.

In the Python script, equations from the previous subsection is encapsulated in a class named FI_matrix, which simulates the EMB system using a discrete representation and estimates the FIM iteratively. Functions contained in the class including:

$f(x, u, theta)$: Defines the state update equations of the EMB model, including dynamic components such as motor torque, load torque, and friction, as showed in equation (4.21).

$h(x)$: Represents the output function that provides system outputs, such as motor position and velocity.

$jacobian(x, u, theta)$: Uses PyTorch's Autograd to compute the Jacobian of the state function f with respect to both the state variables and system parameters.

$sensitivity\_x()$ and $sensitivity\_y()$: Calculate the sensitivity of the states and outputs with respect to the system parameters recursively, as described in equation (4.22) and (4.25).

$fisher\_info\_matrix()$: Computes the FIM by accumulating the sensitivity information over all time steps.

The use of PyTorch's Autograd for automatic differentiation greatly simplify the calculation of Jacobians, ensuring accurate and efficient sensitivity computation throughout the simulation. Furthermore, as this script is incrementally updated, at each time step not only can the FIM be readily computed, but also its corresponding state variables and system outputs, as well as the sensitivity of each state variable and output with respect to each parameter. This enables the monitoring and evaluating of the trend of FIM and sensitivity at an episode, which is a valuable contribution to the next reinforcement learning framework.

4.4 Initialization and normalization of the Fisher Information Matrix

After the automated computational FIM was written, predetermined inputs were used to test the performance of the script. Different system input trajectories were selected during the testing process, such as step inputs, ramp inputs, and sinusoidal inputs. In comparison to the other two optimality criteria, the performance of the D-optimality criterion was initially evaluated, as it represents the optimality in the parameter space.

Figure 4.2 illustrates the progression of the system input u, state parameters, x_1 and x_2, the determinant of the FIM, denoted by $|M|$, the logarithm of the determinant, represented by $log(|M|)$, and the condition number of the FIM, $\kappa(M)$, over time in response to a constant input trajectory. However, unexpected behavior was noted when calculating the log determinant of the FIM. Figure 4.2(c) shows he log determinant curve is discontinuous. Further examination of the determinant values from Figure 4.2(b) revealed that this discontinuity was caused by negative values for the determinant of the FIM at certain time points, which caused the logarithm function to return a NaN (not-a-number) value. Since the FIM is a symmetric semi-positive definite matrix, its determinant should not be negative.

The erroneous negative determinant values were likely due to floating-point errors inherent in the determinant calculation, particularly when the matrix approaches singularity. The FIM was found to be approximately singular at certain time points, as evidenced by a sudden increase in the conditional number of the matrix at those points. As the red box showed in Figure 4.2(f), during the experiment, the conditional number of the FIM spiked unexpectedly at several intervals, which resulted in the matrix becoming ill-conditioned and leading to numerical instability during determinant calculation.

In reinforcement learning, where the input trajectory is determined by the agent, it is not feasible to anticipate when the FIM will become nearly singular and design the input to avoid this issue. Furthermore, the determinant of the FIM tends to increase rapidly over time, which could lead to potential overflow issues. To mitigate this problem, method was developed to scale the FIM matrix, ensuring numerical stability and preventing such issues during determinant calculation.

Furthermore, a step reward function was devised based on the characteristics of reinforcement learning in conjunction with the D-optimality criterion. The issue that needed to be addressed was that, due to the incremental nature of FIM calculation—where the FIM at each time step is equivalent to the FIM from the previous time step plus a rank-1 update—the determinant of the FIM cannot become non-zero until at least n steps have been completed. This behavior presents a challenge for calculating the reinforcement learning reward function in the initial phase. Consequently, an initialization method for the FIM was developed and demonstrated to have no adverse effect on the optimization results.

4.4.1 Initialization of the Fisher Information Matrix

In the design of the calculation of the FIM, an assumption is made

$$M_0 = 0 \tag{4.27}$$

indicating that no information is obtained at the beginning of the experiment. Since at each time step during the training, the FIM is updated by a rank-1 update, initializing the FIM as in Equation (4.27) poses an issue: at least during the initial n steps, the step reward cannot be calculated because $rank(M_k)$ is less than n. The determinant of an n-dimensional matrix with rank less than n is zero. To address this,

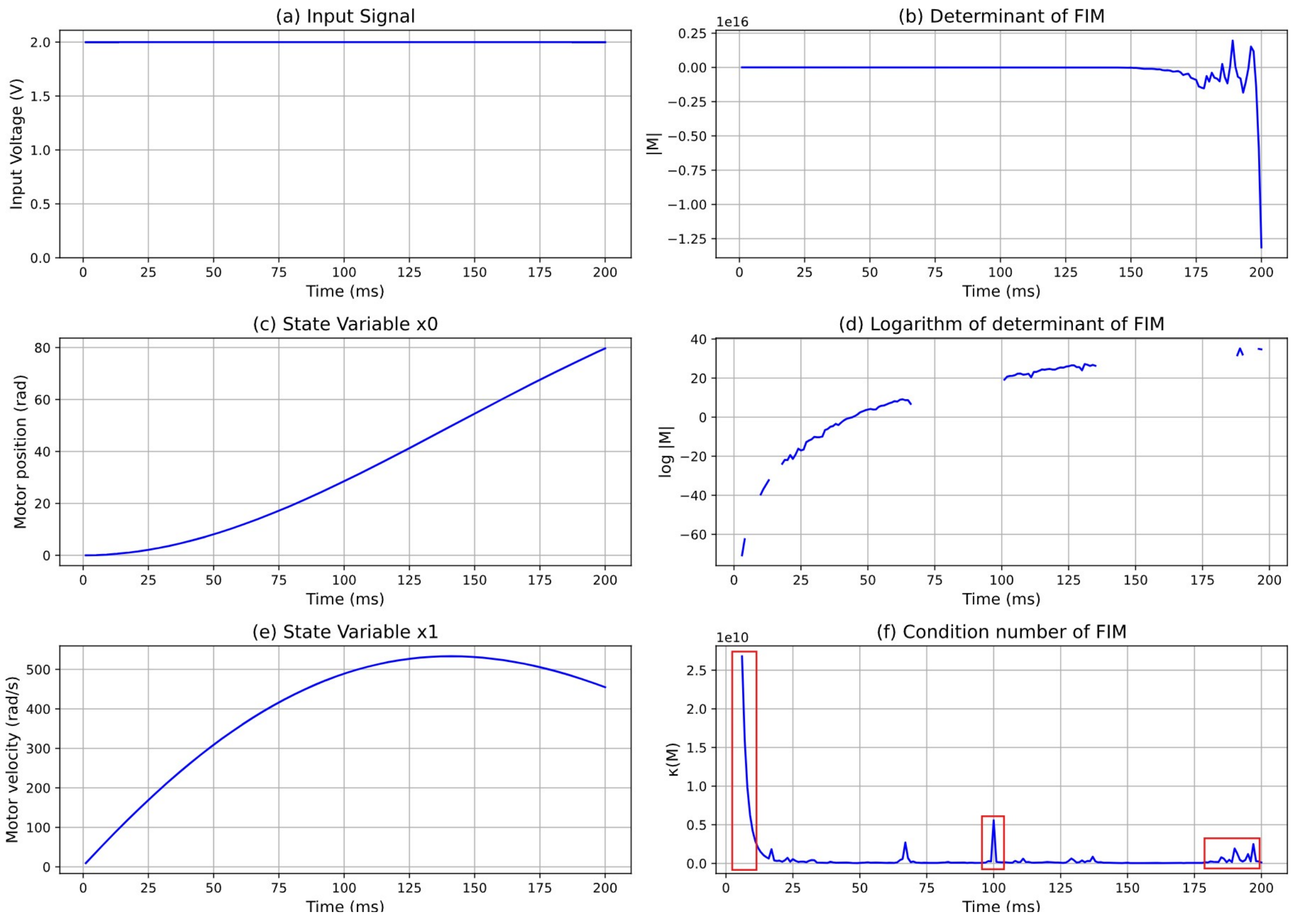

Figure 4.2: System Response Over Time to a Constant Input Trajectory: (a) System input u. (b,c) State parameters x_1 and x_2. (d) Determinant of the FIM $|M|$. (e) Logarithm of the determinant $log(|M|)$. (f) Condition number of the FIM, $\kappa(M)$

the FIM is initialized with a sufficiently small value ϵ multiplied by an identity matrix I of appropriate dimension

$$M_{init} = \epsilon I \tag{4.28}$$

As add an identity matrix will change the FIM in the end, it could be prove the affection could be neglected with Courant-Fisher theorem. To proceed, the Courant-Fischer theorem is first introduced and used to prove the following inequality

$$\lambda_i(A) + \lambda_1(B) \leq \lambda_i(A+B) \leq \lambda_i(A) + \lambda_n(B) \tag{4.29}$$

with eigenvalues $\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$

Theorem 1 (Courant-Fischer Theorem). *[36] Let* $\mathbf{M}$ *be a symmetric matrix with eigenvalues* $\mu_1 \geq \mu_2 \geq \cdots \geq \mu_n$. *Then,*

$$\mu_k = \max_{\substack{S \subseteq \mathbb{R}^n \\ \dim(S)=k}} \min_{\substack{\mathbf{x} \in S \\ \mathbf{x} \neq 0}} \frac{\mathbf{x}^T \mathbf{M} \mathbf{x}}{\mathbf{x}^T \mathbf{x}} = \min_{\substack{T \subseteq \mathbb{R}^n \\ \dim(T)=n-k+1}} \max_{\substack{\mathbf{x} \in T \\ \mathbf{x} \neq 0}} \frac{\mathbf{x}^T \mathbf{M} \mathbf{x}}{\mathbf{x}^T \mathbf{x}},$$

where the maximization and minimization are over subspace S and T of $\mathbb{R}^n$.

Consider a matrix $M = A + B$ with a unit vector $\mathbf{x}$, by using the Courant-Fisher Theorem, the i-th eigenvalue of the matrix M, λ_i can be characterized as

$$\lambda_i = \max_{\substack{S\subseteq\mathbb{R}^n \\ \dim(S)=k}} \min_{\substack{\mathbf{x}\in S \\ \mathbf{x}\neq 0}} (\mathbf{x}^T\mathbf{A}\mathbf{x} + \mathbf{x}^T\mathbf{B}\mathbf{x}) \tag{4.30}$$

because $\mathbf{x}^T\mathbf{x} = 1$. Could be noticed that

$$\mathbf{x}^T\mathbf{B}\mathbf{x} \geq \lambda_1(\mathbf{B}) \tag{4.31}$$

for any unit vector $\mathbf{x}$, because $\lambda_1(\mathbf{B})$ is the smallest eigenvalue of $\mathbf{B}$. Thus

$$\begin{aligned} \lambda_i(\mathbf{A}+\mathbf{B}) &\geq \max_{\substack{\dim(S)=i \\ \mathbf{x}\in S, \|\mathbf{x}\|=1}} \min \left(\mathbf{x}^T\mathbf{A}\mathbf{x} + \lambda_1(\mathbf{B})\right), \\ &= \lambda_i(\mathbf{A}) + \lambda_1(\mathbf{B}) \end{aligned} \tag{4.32}$$

Similarly could be proved that

$$\begin{aligned} \lambda_i(\mathbf{A}+\mathbf{B}) &\leq \max_{\substack{\dim(S)=i \\ \mathbf{x}\in S, \|\mathbf{x}\|=1}} \min \left(\mathbf{x}^T\mathbf{A}\mathbf{x} + \lambda_1(\mathbf{B})\right), \\ &= \lambda_i(\mathbf{A}) + \lambda_n(\mathbf{B}) \end{aligned} \tag{4.33}$$

Thus the inequality (4.29) is proved.

When the FIM M_n is initialized with a sufficiently small value ϵ multiplied by an identity matrix I, inequality (4.29) becomes the following equation

$$\lambda_i(M_n + M_{init}) = \lambda_i(M_n) + \epsilon \tag{4.34}$$

Thus, initializing the FIM with $M_0 = \epsilon I$ ensures that the matrix remains non-singular while having a negligible effect on its eigenvalues. This allows the objective function to be evaluated consistently from the start, ensuring stability in the early stages of the experiment without compromising the optimization process or the accuracy of the results.

4.4.2 Normalization of the Fisher Information Matrix

Another observation from Figure 4.2 is that during the experiment, the value of $det\,(M_k)$ can easily grow to an extremely large number due to the large values of the matrix elements. This can lead to a consequence where the condition number of the FIM also becomes large, making the calculation of the determinant unstable. To restrict the growth of $det\,(M_k)$, a normalization approach with normalization factor α_k is designed

$$\bar{M_k} = \alpha_k M_k \tag{4.35}$$

where $\bar{M_k}$ is the normalized FIM.

After the normalization, it is important to keep the step reward unchanged, otherwise it will change the object function of the optimization problem. This is possible due to the properties of logarithms and determinants, which allows Equation (4.13) to be rewritten in

$$\begin{aligned}
r_k &= log|M_k| - log\,det M_{k-1}| \\
&= log\frac{|M_k|}{|M_{k-1}|} \\
&= log\frac{\alpha_k|M_k|}{\alpha_k|M_{k-1}|} \\
&= log|\bar{M}_k| - log|\frac{\alpha_k}{\alpha_{k-1}}\bar{M}_{k-1}|
\end{aligned} \tag{4.36}$$

Thus, the original FIM at time step k is not needed for the calculation of step reward r_k.

Due to Equation (4.2), the FIM is actualization by rank-1 update, FIM at time step k is

$$M_k = M_{k-1} + M_{\bar{\theta},k} \tag{4.37}$$

Hence, Equation (4.35) changes into

$$\begin{aligned}
\bar{M}_k &= \alpha_k M_k \\
&= \alpha_k(M_{k-1} + M_{\bar{\theta},k}) \\
&= \frac{\alpha_k}{\alpha_{k-1}}\bar{M}_{k-1} + \alpha_k M_{\bar{\theta},k}
\end{aligned} \tag{4.38}$$

Thus, the update for the normalized FIM in this approach also have no requirement for the calculation of original FIM, avoiding the occurrence of a potentially large determinant.

One reasonable way to choose the initial normalized FIM is make it same to the initial FIM in last subsection, and to design α_k is to always keep $det\left(\bar{M}_k\right)$ related to the determinate of the initial matrix

$$\alpha_k = \left(\frac{det\left(M_{init}\right)}{det\left(\bar{M}_{k-1}\right)}\right)^{\frac{1}{n}} \tag{4.39}$$

An experiment with the same input signal after initialization and normalization was conducted and is shown in Figure 4.3. As observed from (b) and (d), the determinant of FIM is positive and logarithm is now continues. From (d), it can be seen that the condition number does not suddenly increase, but grows continuously as the experiment progresses.

4.5 Reinforcement Learning

In this thesis, reinforcement learning methods will be used to carry out input excitation optimisation used for EMB parameter estimation. Reinforcement learning is a machine learning method. The content of it can be abstracted into two parts, the agent and the environment. The agent can gain experience and learn by interacting with the environment, giving the agent the ability to accomplish set tasks autonomously. In the application of this thesis, the environment is the EMB and the agent is responsible for giving the input excitation used for parameter estimation.

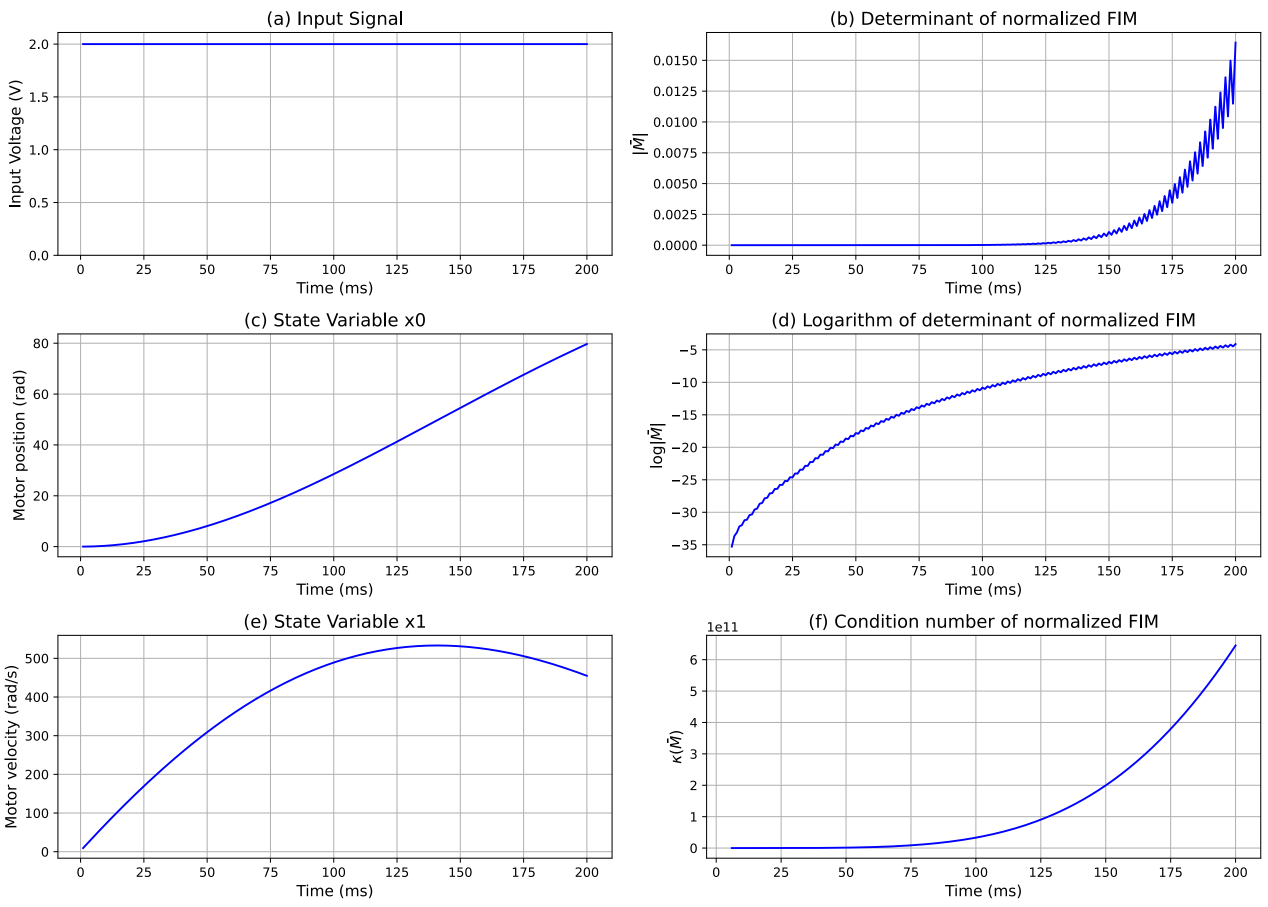

Figure 4.3: System response over time to a constant input trajectory after initialization and normalization of FIM: (a) System input u. (b,c) State parameters x_1 and x_2. (d) Determinant of the FIM $|\bar{M}|$. (e) Logarithm of the determinant $log(|\bar{M}|)$. (f) Condition number of the FIM, $\kappa(\bar{M})$

Reinforcement learning based on the Markov Decision Process (MDPs), it is an abstraction of the real world. An MDPs is defined by a tuple (S, A, P, R, γ), where S is the set of states, A is the set of actions, P represents the transition probability between states, R is the reward function, and γ is the discount factor for the future rewards [37]. Figure 4.4 illustrates this process. Starting from the initial state s_0, the agent takes an action a_1 that leads to a new state s_1 and receives a reward r_1, and then this process will continue in the same way. The goal of the agent is to find an optimal policy $\pi(a|s)$ which could based on the state s, do the action a and maximize the expected cumulative return

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \tag{4.40}$$

And the expected cumulative return based on strategy π_θ is

$$J = \mathbb{E}_\pi(G_t) \tag{4.41}$$

Action-value function and state-value function build up the core part of the reinforcement learning. The action-value function $Q_\pi(s, a)$ describes the expected return if the agent takes action a in state s with a

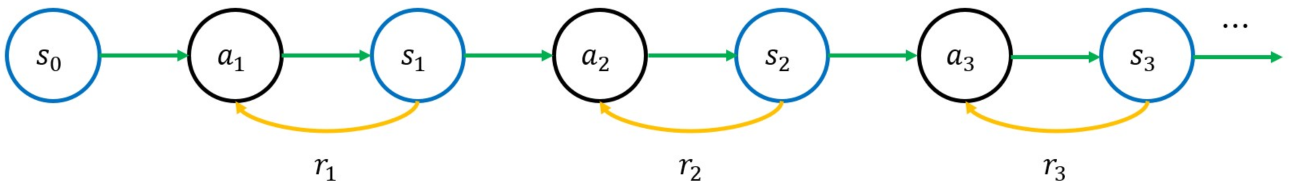

Figure 4.4: Testing Actor with EMB environment after training

policy π

$$\begin{aligned} Q_\pi(s,a) &= \mathbb{E}\left(G_t \middle| s_t = s, a_t = a, \pi\right) \\ &= \mathbb{E}\left(\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \middle| s_t = s, a_t = a, \pi\right) \end{aligned} \tag{4.42}$$

The state-value function $V_\pi(s)$ describes the expected return from a given state s under a policy π

$$\begin{aligned} V_\pi(s) &= \mathbb{E}\left(G_t \middle| s_t = s, \pi\right) \\ &= \mathbb{E}\left(\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \middle| s_t = s, \pi\right) \\ &= \sum_{a \in A} \pi(a|s) Q_\pi(s,a) \end{aligned} \tag{4.43}$$

Almost all reinforcement learning algorithms based on the calculation of action-value function and state-value function. The relationship between those two function is build up by advantage function. In reinforcement learning, the advantage function A is used to measure the advantage of the current action a over the average behavior in the current state s based on the current strategy π

$$A(s,a) = Q_\pi(s,a) - V_\pi(s) \tag{4.44}$$

Policy gradient methods is a kind of methods which directly optimize the policy π [38]. The basic idea is to increase the probability of actions that can bring about positive advantages and decrease the probability of bringing about negative actions. Advance function is employed in this method to compute the gradient of the objective function

$$\nabla J(\pi) = \mathbb{E}\left[\nabla \log \pi(s,a) A(s,a)\right] \tag{4.45}$$

Traditional reinforcement learning methods, such as Q-learning and Temporal Difference (TD) methods, optimize an agent's policy by computing Q and V values. These methods have the advantage of being simple to construct and easy to understand, but they can only handle relatively simple problems. For example, in the case of Q-learning, maintaining a Q-table action space to be discrete, and in case of huge dimension of state space will directly lead to huge dimension of the Q-table, which limits its applicability to more complex scenarios.

With the development of deep learning and neural networks, reinforcement learning has gradually merged with deep learning. Since deep neural networks have the ability to fit arbitrary functions, Deep Reinforcement Learning (DRL) is theoretically almost unlimited in observation space and action space. This gives DRL the ability to solve complex problems in complex environments, such as robotic arm grasping and robot path planning problems.

So far, several DRL methods have been developed, such as Deep Q-Networks (DQN), Deep Deterministic Policy Gradient (DDPG), Twin Delayed DDPG (TD3), and Proximal Policy Optimization (PPO) [39]–[42]. Expect DQN and its variant, all otter DRL algorithm are based on Actor-Critic methods.

Compared to other DRL algorithms, PPO is known for its robustness. Since the application scenario in this thesis is a self-developed EMB model environment, unlike well-established reinforcement learning environments used wildly for benchmarking, the custom environment may be highly sensitive to the parameters of deep reinforcement learning algorithms, making tuning challenging. Compared to DDPG or A3C, PPO is relatively less sensitive to hyperparameter settings, making it a practical choice for this EMB environment that may be unstable or difficult to tune. Another advantage for PPO is, the clipping mechanism prevents large updates, making it more stable and robust compared to other actor-critic methods, which makes it suitable for the purposes of this thesis.

4.5.1 Actor-Critic methods

As showed in the name, two main components are included in this method: the Actor and the Critic. Actor is used to output the action, while Critic is responsible for give evaluation to those actions. In DRL, this two components are represented by two separate neural networks, policy network for Actor and value network for Critic. Policy network uses the policy gradient method combined advantage function $A(s, a)$ to update the network, as show in Equation (4.45).

For the critic network, the Temporal Difference (TD) error is used to minimize the difference between the predicted value ($V(s_t)$) and the actual return (G_t). The critic loss L_V is typically computed as mean squared error (MSE) of the TD error

$$= \frac{1}{2}\left(V(s_t) - G_t\right)^2 \tag{4.46}$$

4.5.2 Proximal Policy Optimization

Proximal Policy Optimization is a reinforcement learning algorithm base on Actor-Critic methods. It is an algorithm that achieves the data efficiency and reliable performance of Trust Region Policy Optimization (TRPO), while using only first-order optimization [42]. The object function of TRPO is introduced by [43]

$$J(\theta) = \hat{\mathbb{E}}_t\left[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}\hat{A}_t\right] \tag{4.47}$$

TRPO introduces a Kullback-Leibler (KL) Divergence constraint to constrain the above objective function and uses fisher information matrix to compute the second-order approximation of the KL divergence. TRPO tries to convert this constrained optimization problem into an unconstrained optimization problem by introducing a hyperparameter β, but difficulty existing in selecting β.

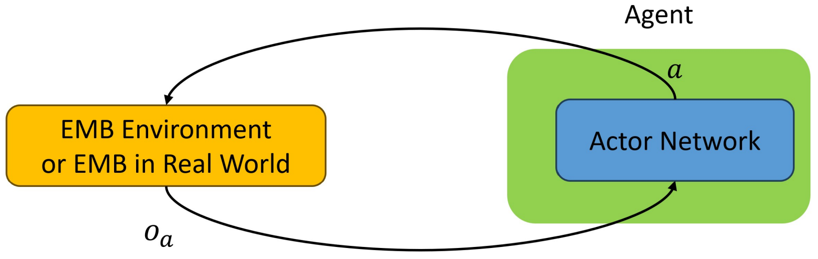

Figure 4.5: Testing Actor with EMB environment after training

The idea of PPO is to set a lower and upper bound for Equation (4.47), to represent the constraint and prevent the policy from making large updates. These bounds are converted into the following clip function with hyperparameter ϵ

$$J^{CLIP}(\theta) = \hat{\mathbb{E}}_t \left[\min \left(\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)} \hat{A}_t, \text{clip}(\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}, 1-\epsilon, 1+\epsilon) \hat{A}_t \right) \right] \tag{4.48}$$

With this clipping mechanism, no KL divergence is needed for the calculation of the gradient of the object function, and the update of the police is strongly restricted. Making it have a more robust performance.

One special feature of PPO is that the deign of memory buffer it doesn't based on episodes and stetting a maximum number of episode [44]. Instead, PPO uses vectorized environment and runs N independent environments in parallel during the training. Each of these parallel environment collects T timesteps of data in the rollout phase. If a done flag d is triggered in any environment, that environment will be automatically reset, and then continue collecting data. During the learning phase, the memory buffer with NT steps data collected in the rollout phase will be used for the loss calculation and updating the networks.

4.5.3 Partially observable environments

In the reinforcement learning problem with EMB optimal input excitation, an important feature is partial observability. The FIM or sensitivity calculations described in the previous section cannot be accurately performed during actual parameter estimation due to the lack of prior parameter values. An agent trained by reinforcement learning should only be able to acquire information from sensors to make decision and give the correct input trajectory. Thus, the gap between the theoretical simulation of optimal experimental design and its practical application needs to be taken into account in the framework design.

The goal of this reinforcement learning framework is to train the critic based on the full information from the EMB environment, which is used to accurately evaluate the behavior of the actor, thus guiding the actor to make the right decision. As showed in Figure 4.5, at the end of training, when the agent is actually used for generate the input excitation, only the actor network is employed to determine the corresponding system inputs based on the current system output measurements and time step, all the rest of the information in the environment stay unknown.

In fact, partial observability not unique to the EMB system, it also existed in many other area. For example in the robotics, robots are usually first trained in the simulator with all state information available. However, when the robot is deployed after training, it might only have images as input, which also leads to the partial observations problem. To address this problem, partially observable Markov decision processes (POMDPs) have been proposed [45]. A POMDPs is a MDP in which the agent is unable to observe the current state. Instead, it makes an observation based on the action and resulting state, and the goal of maximize expected discounted future reward remains the same. POMDPs are described by the tuple $\langle S, A, T, R, \Omega, O \rangle$ [45], while S, A, T, R stay the same meaning with MDP, Ω is a set of possible observations that the agent can experience in its environment. $O : S \times A \rightarrow \Pi(\Omega)$ is the observation function, which provides a probability distribution over observations given a state and action.

POMDPs have been used in planning and control tasks [46], [47]. Most POMDPs applications rely on offline algorithms, the disadvantage of it is that the training time is too long, also it can not deal with small changes in the environment because the entire offline policy need to recalculated. To address these issues, several online POMDPs algorithms have been proposed and studied [48]–[50].

The Asymmetric Actor-Critic framework is variant of Actor-Critic framework, where full state in observation space is used to train the critic, while partial observation is used to trains the actor [51]. This framework has been applied to image-based robot learning and contextual learning tasks [52]. Although many asymmetric methods are effective, they are often evaluated in limited scenarios. Recent research has proved an asymmetric policy gradient theorem for partially observable control, an extension of the policy gradient Actor-Critic methods theorem[53].

Since PPO also based on Actor and Critic networks, it should be consider as a good way to introduce the symmetric Actor-Critic approach into PPO to deal with the existing partially observable space problem in EMB. The Detail of the asymmetric PPO framework will be conducted in the next section.

4.5.4 Asymmetric PPO framework

Inspired by the existing researches, in this thesis, an asymmetric PPO framework is introduced. As show in the Algorithm 1, compared to the classic PPO algorithm, this framework employs an asymmetric input strategy for the actor and critic networks. As showed in Figure 4.6, while the observation o from the complete observation space is stored in memory after each step, the input to the actor network, o_a, contains only a partial observation. In contrast, the input to the critic network, o_c, contains more information and differs only slightly from the full observation o. Therefore, when designing and initializing the actor and critic networks, the different input structures must be taken into consideration.

In conjunction with the framework design specifically, the observation space of reinforcement learning contains all the information mentioned above. But the observation space of actor o_a has only three dimensions, including the measured values of motor position and speed and the time step, while the observation space of critic o_c contains the true values of motor position and speed, the a priori values of system parameters obtained from sampling the parameter distributions, as well as the elements of the time step and the elements of FIM. The difference of the observation space for those two network is listed in Table 4.2. During the development of the Asymmetric PPO framework, the PPO algorithm from $CleanRL$ is chosen as the fundamental reinforcement learning algorithm framework [54].

Network	Observation Space
Actor	$x_{1,m}, x_{2,m}, k$
Critic	$x_1, x_2, k, \theta_{sample}, FIM$

Table 4.2: Asymmetric inputs for Actor and Critic

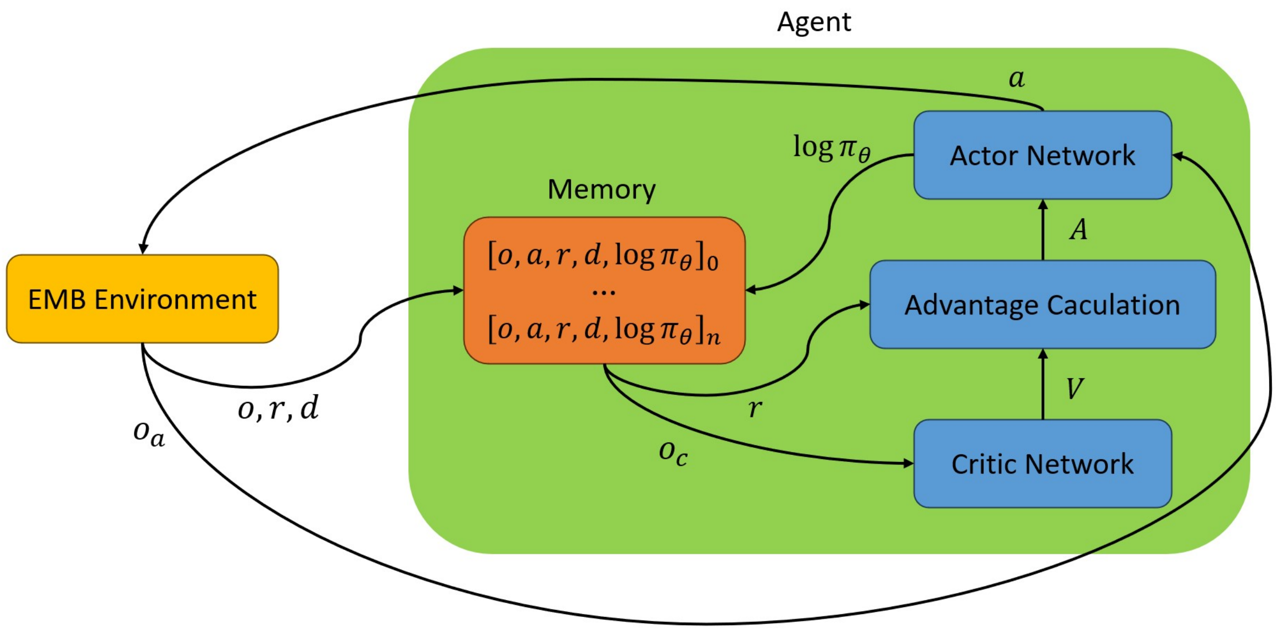

Figure 4.6: Training asymmetric PPO framework with EMB environment

4.6 EMB environment setup for reinforcement learning

After solving the computational aspects of the optimal experimental design for the EMB system, one remaining challenge is build up an environment that can interact with reinforcement learning algorithms. In the field of reinforcement learning, a widely used standard interface is Gymnasium.

Gymnasium is an open-source library that provides a standard interface for reinforcement learning environments. It has features that allow different environments and training algorithms to work together easily, making it simpler to develop and test RL algorithms [55]. Gymnasium includes many classic reinforcement learning environments, such as $CartPole$, $Pendulum$, and $LunarLander$, etc. By using these existing environments, interaction with RL algorithms can be easily achieved without the need to implement the environment, and most of the training results can be visualized.

In addition, Gymnasium provides standard tools and interfaces for reinforcement learning that allow for easy customization of environments. Since the EMB model in this thesis is a simplified simulation model based on a real test platform, building the corresponding reinforcement learning environment in Gymnasium is necessary for applying RL to the optimal experimental design.

Algorithm 1 Asymmetric PPO Framework

```
1:  Initialize Actor-Critic networks
2:  Initialize memory buffer $\mathcal{B}$
3:  Sample initial observation state $o_0$
4:  for update $= 1, 2, \ldots$ do
5:      for actor $= 1, 2, \ldots, N$ do
6:          for $t = 0, T - 1$ do
7:              Get action $a_t$ and action distribution $\log \pi_\theta$ using actor policy $\pi_\theta(o_{a,t})$
8:              Execute action $a_t$, receive reward $r_t$, next observation $o_{t+1}$, and done flag $d_t$
9:              Get value $V_t$ from critic network
10:             Store $(o_t, a_t, r_t, d_t, V_t)$ in memory buffer $\mathcal{B}$
11:             if done flag $d_t == 1$ then
12:                 Sample initial observation state $o_0$
13:             end if
14:         end for
15:     end for
16:     Compute returns $R$ and advantages $A$ with stored trajectories in memory buffer $\mathcal{B}$
17:     for epoch $= 1, 2, \ldots, K$ do
18:         Shuffle memory buffer $\mathcal{B}$ and divide into mini-batches
19:         for each minibatch do
20:             Update actor network using $(o_{a,t}, a_t, r_t, A_t)$
21:             Update critic network using $(o_{c,t}, a_t, r_t, V_t)$
22:         end for
23:     end for
24:     Clear memory buffer $\mathcal{B}$
25: end for
```

The environment is the core part in Gymnasium. It is the instantiation of the RL environment and enables interaction with RL algorithms [55]. An environment consists of observation space, action space, $step$ method, and $reset$ method. The EMB model constraints introduced in Section 4.1 and the optimal experimental design requirements described in Section 4.3 will be implemented in these components. The following part describes how each section is built in a customized EMB environment.

The input to the EMB system is the motor voltage, which ranges from -6V to 6V, hence it corresponds to a one-dimensional action space. The design of the observation space determines the amount of information that the RL agent can obtain from the environment, and needs be considered in context of the task requirements and the structure of the RL algorithm. Parameters that can be contained in the observation space include the actual and measured values of motor position and speed, the prior values of parameters to be identified, the time step and elements in the FIM, etc. The $reset$ method is used to initialize the environment, which is mainly responsible for setting the parameters in the observation space to their initial values.

The $step$ function is the core of the reinforcement learning environment. In this part, the agent's action is input into the EMB system, and the motor position and speed in next step are updated. Additionally, the function used to calculate the FIM from the previous section is called and the states in the observation space are updated. The $step$ function also includes two conditions that automatically trigger the $reset$ function. The first one is when the time step reaches 500, which means that the experiment has lasted for

500 ms and it is over. Furthermore, the agent must learn to control the motor so that it does not exceed speed and position limits. If this limit is broken in the step function, the $reset$ function is automatically triggered. This feature is implemented by the $terminated$ function within the environment. Another important part of the step function is to update the step reward based on the design of the reward function, which will directly affect the agent's training performance. The design of the reward function will be introduced in detail in the following sections.

4.7 Formulation of parameter initialization methods in observation space

With the reinforcement learning algorithm selected in Section 4.5 and the EMB environment for reinforcement learning implemented in Section 4.6, this section introduces the formulation of parameter initialization methods in the observation space. These methods aim to address the limitations of conventional approaches that rely heavily on prior parameter values for optimal experiment design.

Two parameter initialization methods are considered in this thesis: the Fixed Parameter Initialization Method and the Random Distribution-Based Parameter Initialization Method. Each method offers distinct characteristics and impacts on the learning process, as detailed below.

4.7.1 Fixed parameter initialization method

In the fixed parameter initialization method, the observation space contains all the information from the EMB environment. This includes the position of the motor, the real and measured values of the velocity, the time step, and the a priori values of the parameters and the FIM elements. By using fixed parameter values, this method provides rich information in the observation space, which may help the agent understand the environment and learn better.

However, a major drawback is its reliance on fixed prior parameter values. If the actual parameter values in the EMB model differ from these fixed values during training, the trained agent may fail to take correct actions. Despite this limitation, this method directly applies the conventional input excitation optimization approach to reinforcement learning. It serves as a baseline to verify the correctness of the reinforcement learning framework and the agent's ability to learn effectively.

4.7.2 Random distribution-based parameter initialization method

Different from the fixed parameter initialization method, the random distribution-based parameter initialization method does not rely on specific prior parameter values. Instead, it uses a distribution of parameters for training. For example, as shown in Figure 4.7, a uniform distribution from θ_{min} to θ_{max} can be used. It ensues equal probability of sampling any parameter value in this range. Depending on the training needs, other distributions, such as the normal distribution, can also be used.

During the training, when the $reset$ function is called in the EMB environment, a set of parameter values is randomly sampled from the defined distribution. These values are saved in the observation space and stay unchanged for training in that episode. This method ensures that no assumptions about the a priori values of the parameters are needed for modeling, and the agent will no longer learn optimal the

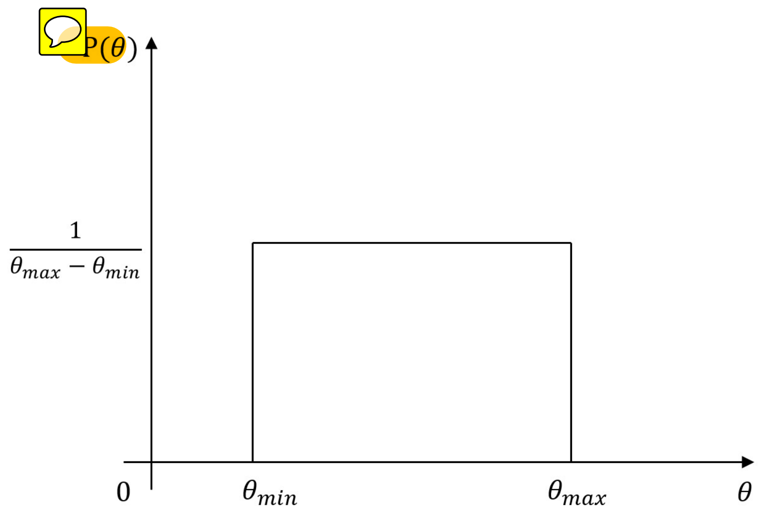

Figure 4.7: Structure of Asymmetric PPO framework with EMB environment

input base on specific parameter values during the training process, but will learn the optimal policy under the parameter distribution based on experience.

The main advantage of this method is the use of randomness in reinforcement learning. By exposing the agent to a variety of parameter values, it helps the agent learn more general strategies and adapt to the uncertainty of the parameter in the model, making it less dependent on specific parameter settings.

5 Approaches of the single parameter Excitation

In this chapter, the previously described reinforcement learning approaches are applied to the EMB system for input excitation optimisation of parameter estimation. Since the dimension of the FIM is directly related to the number of parameters to be identified, although a simplified model of the EMB system was developed in Chapter 3, a conservative research strategy has been adopted as the initial step in exploring RL applications in this field.

For the FIM, the simplest form is a one-dimensional matrix, corresponding to the scenario where only a single parameter needs to be identified while the remaining parameters are assumed to be known. Therefore, this chapter focuses on optimizing input excitation for the estimation of a single parameter. The parameter chosen for this study is the viscous friction coefficient of the EMB system, f_v.

Additionally, while the asymmetric PPO algorithm was established in Chapter 4, as well as both fixed-parameter initialization and random distribution-based parameter initialization methods were described, these methods have yet to be applied to EMB problems. Hence, the performance of the proposed framework remains uncertain. To address this, the features of the framework will be progressively expanded and compared in this chapter.

Moreover, practical considerations for parameter estimation experiments are introduced. For instance, it is essential for the motor to automatically return to its initial state at the end of the experiment. These practical requirements are incorporated into the RL framework through different reward function designs, which will also be explored and evaluated in this chapter.

5.1 Sensitivity analysis of the viscous friction parameter

As the viscous friction f_v chosen as the single parameter to be estimated. An analysis of the behaviour of the sensitivity also fisher information matrix with regard to the viscous friction parameter will prove beneficial for the implementation of reinforcement learning. The calculation of the FIM is based on the formula given in Equation (3.12) and (4.1)

$$\frac{\partial f_{d,k}}{\partial x_d^T} = 1 + t_s \begin{bmatrix} 0 & 1 \\ -\frac{\gamma k_1}{J} & -\frac{f_v}{J} \end{bmatrix} \tag{5.1}$$

Although only a single parameter is estimated here, the normalization of the parameter, as discussed in Section 4.3.2, is not strictly necessary. However, to maintain consistency with the calculations in the later multi-parameter case, the normalization of the parameter is still performed. This normalization only affects the result by introducing a factor equal to the nominal value of f_v. Using Equation (3.14) and the normalized f_v

$$\frac{\partial f_{d,k}}{\partial \bar{f}_v} = t_s \begin{bmatrix} 0 \\ -\frac{f_v x_2}{J} \end{bmatrix} \tag{5.2}$$

According to Table 4.1, in the EMB model used in this thesis, the nominal range of position is $(-10, 100)$, the nominal range of velocity is $(-500, 500)$. Because of the calculation of the FIM is strongly relevant with the measurement value, to avoid heavy weighting on the velocity, the measured outputs need to be normalized with approach in Section 4.3.3

$$\bar{h}_k = \begin{bmatrix} \frac{1}{100} & 0 \\ 0 & \frac{1}{500} \end{bmatrix} h_k \tag{5.3}$$

Hence the Jacobian matrix of $\bar{h}_k$ is

$$\frac{\partial \bar{h}_k}{\partial x_d^T} = \begin{bmatrix} \frac{1}{100} & 0 \\ 0 & \frac{1}{500} \end{bmatrix} \tag{5.4}$$

As shown in Equation (3.15), the output is not directly relevant to the viscous friction, thus

$$\frac{\partial \bar{h}_k}{\partial \bar{f}_v} = 0 \tag{5.5}$$

With the Equations above and according to Equation (4.25), the sensitivity of each state variable with respect to viscous friction is

$$\frac{\partial x_{d,k+1}}{\partial \bar{f}_v} = \frac{\partial x_{d,k}}{\partial \bar{f}_v} + t_s \left(\begin{bmatrix} 0 & 1 \\ -\frac{\gamma k_1}{J} & -\frac{f_v}{J} \end{bmatrix} \frac{\partial x_{d,k}}{\partial \bar{f}_v} + \begin{bmatrix} 0 \\ -\frac{f_v x_2}{J} \end{bmatrix} \right) \tag{5.6}$$

This equation already shows some similarity to some existed dynamic system. The analyse will be conducted in the following text.

5.1.1 Similarity with mass-spring-damper system

Consider a mass-spring-damper with a mass m, spring constant k, and damping coefficient c. The equations of motion of the system are given by

$$m\ddot{x} + c\dot{x} + kx = F(t) \tag{5.7}$$

This equation can be represented in state-space form and discretized using the Euler method, as shown in the following equation

$$x_{k+1} = x_k + t_s(\begin{bmatrix} 0 & 1 \\ -\frac{k}{m} & -\frac{c}{m} \end{bmatrix} x_k + \begin{bmatrix} 0 \\ \frac{1}{m} \end{bmatrix} u_k) \tag{5.8}$$

Comparing the equation above to Equation (5.6), it can be observed that they share the same structure. Therefore, the sensitivity of the state variable of EMB system with respect to viscous friction can be seen as analogous to the state variable of a mass-spring-damper system with mass $m = J$, spring constant $k = \gamma k_1$, and damping coefficient $c = f_v$. In this analogy, the term $-f_v x_2$ serves as the input u to the system. Consequently, the calculation of $\frac{\partial x_{d,k}}{\partial \bar{f}_v}$ can be interpreted as solving for the state variable of a mass-spring-damper system.

Using the parameter values from Table 4.1, the damping ratio ζ of the system is calculated as

$$\zeta = \frac{c}{2\sqrt{km}} = \frac{2.16 \times 10^{-5}}{2\sqrt{(23.04 \times 1.889 \times 10^{-5})(4.624 \times 10^{-6})}} \approx 0.2407 \tag{5.9}$$

Hence, the system is classified as an underdamped system ($\zeta < 1$) with an input linearly dependent on x_2. If the state variable of this system is defined by $\frac{\partial x_{d,k}}{\partial \bar{f}_v} = [a, b]^T$. Using Equations (4.22) and (5.4), the following relationship can be derived

$$\frac{d\bar{h}_k}{d\bar{f}_v} = \begin{bmatrix} \frac{1}{100} & 0 \\ 0 & \frac{1}{500} \end{bmatrix} \frac{\partial x_{d,k}}{\partial \bar{f}_v} = \begin{bmatrix} \frac{a}{100} \\ \frac{b}{500} \end{bmatrix} \tag{5.10}$$

With the inverse of the output noise variance matrix

$$R^{-1} = \begin{bmatrix} 10^6 & 0 \\ 0 & 1 \end{bmatrix} \tag{5.11}$$

Finally the increment of the Fisher information matrix at each time step can be calculated as

$$\begin{aligned} M_{\bar{f}_v,k} &= \left(\frac{d\bar{h}_k}{d\bar{f}_v}\right)^T R^{-1} \left(\frac{d\bar{h}_k}{d\bar{f}_v}\right), \\ &= \begin{bmatrix} \frac{a}{100} & \frac{b}{500} \end{bmatrix} \begin{bmatrix} 10^6 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{a}{100} \\ \frac{b}{500} \end{bmatrix}, \\ &= 100a^2 + \frac{b^2}{250000} \end{aligned} \tag{5.12}$$

This result indicates that, at each time step, the increment of the FIM $M_{\bar{f}_v,k}$ is directly related to the state variable of the mass-spring-damper system. Due to the presence of sensor noise, the first state variable a which is associated with position in the mass-spring-damper system, has a significantly stronger influence on the FIM increment.

It is intuitive to come to the conclusion that based on the target of gaining a higher information level, also higher FIM element value in this single parameter case, because the matrix is one dimensional, the approach is in crease the position of the object in the mass damping sprint system. This can be achieved by providing a higher input, which is proportional to the EMB motor velocity x_2. In summary, for the input excitation in the single parameter estimation of viscous friction f_v, a higher motor velocity will result in a greater gain of information.

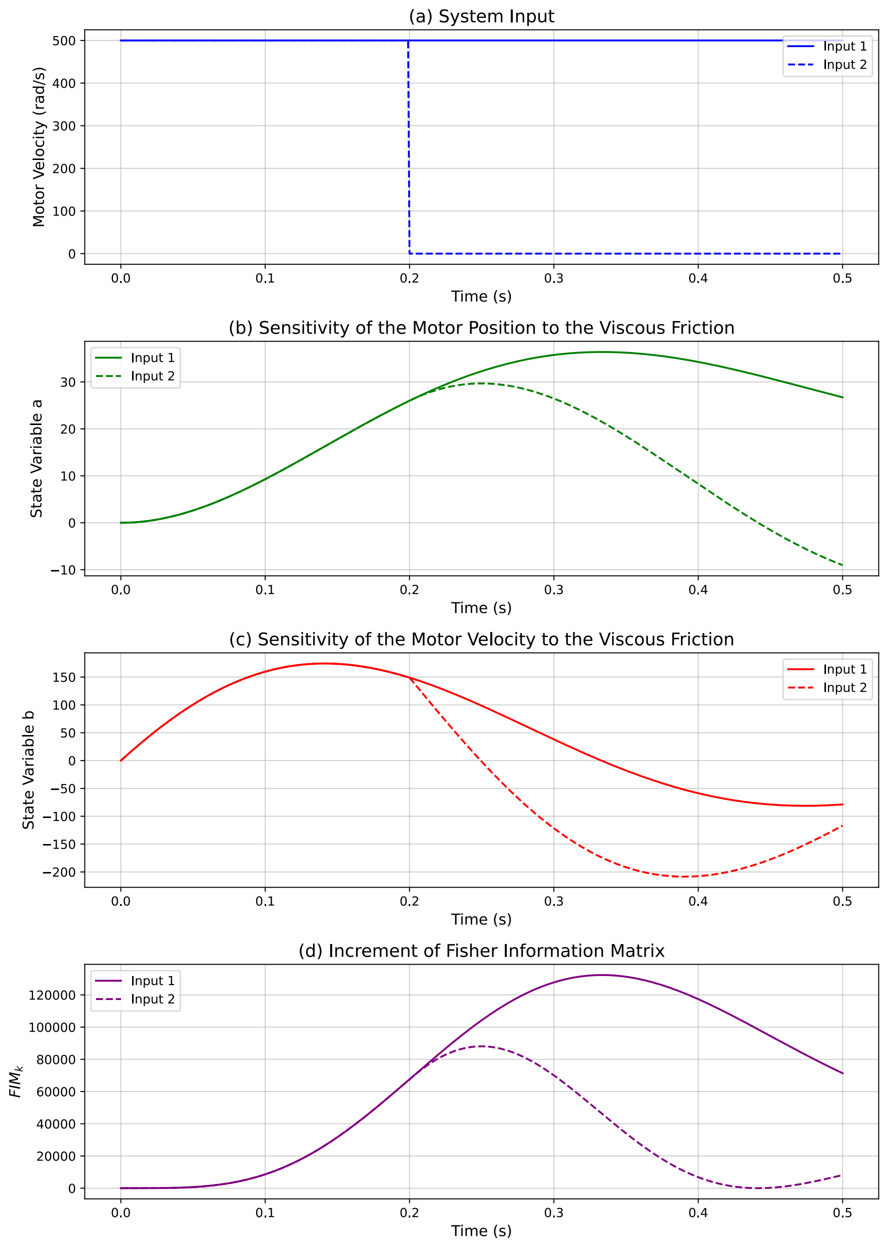

Figure 5.1: Comparison of system dynamics and increment of FIM with different motor velocity trajectory as inputs

The conclusions derived from the above FIM calculation model were validated through theoretical simulations. Figure 5.1 illustrates the effect of two different motor velocity input trajectories on a simulated mass-spring-damping system with mass $m = J$, spring constant $k = \gamma k_1$, and damping coefficient $c = f_v$ and the corresponding increments of the FIM. Input 1 represents the theoretically optimal motor velocity trajectory, where the motor velocity is maintained at a constant 500 rad/s throughout the experiment, regardless of motor angle constraints.

The system's state parameter trajectories, shown in Figures 5.1 (b) and (c), align with the step response characteristics of an underdamped system. At approximately 0.34 seconds, the system's state parameter a, which representing the sensitivity of the motor position to viscous friction, reaches its peak value before starting to decline, leading to a corresponding reduction in the FIM increment in Figure 5.1 (d). The trajectory of Input 1 in Figure 5.1 (d) represents the upper limit of the information content achievable through the system's input excitation.

In contrast, Input 2 operates at its maximum motor velocity between 0 and 0.2 seconds, after which the speed is held constant at 0 rad/s. As observed in Figure 5.1 (b), the system's state parameter a decreases rapidly after 0.2 seconds, as the input to the mass-spring-damping system becomes zero. Based on the characteristics of underdamped systems, a is expected to oscillate around zero with a gradually decreasing amplitude, which in turn causes the FIM increment to approach zero over time, as shown in Figure 5.1 (d).

Interestingly, even when the motor velocity is held at 0 for an extended period, the FIM increment temporarily increases before eventually decaying to zero. This phenomenon suggests that, for a brief duration, information about the viscous friction coefficient increases even at zero speed. However, this effect can not last for long and will finally disappear. Due to the time constraints of the experiment, this effect does not fully disappear within the observed 0.5 second duration, as shown by the trajectory of Input 2 in Figure 5.1 (d).

Although these conclusions do not directly translate into the design of a reward function or help a reinforcement learning agent learn more effectively, drawing an analogy between the FIM computation process and a mass-spring-damping system provides a deeper understanding of the system behavior of the FIM increment. This understanding proves valuable for analyzing subsequent experimental results.

5.2 Comparison of performance between parameter initialization methods

Since an important goal of this thesis is to explore the use of reinforcement learning to reduce the reliance of traditional parameter estimation optimization methods on the a priori values of system parameters, the performance of two parameter initialization methods is first compared.

The analysis of optimal experiment design begins with testing the effectiveness of the PPO algorithm and parameter initialization methods. In this subsection, the PPO algorithm utilizes a continuous action space, where the agent's action range is from -6V to 6V. The observation space of the reinforcement learning environment includes the following items: motor position, motor velocity, time step, viscous friction parameter, and the value of FIM. An idealized environment is employed here, where both the actor and critic have full access to all information in the observation space, including the true values of motor position and velocity. This idealization assumes perfect observability, which is not possible in real world scenarios. Consequently, the asymmetric PPO algorithm is not used in this section.

Given that the overall design of the parameter estimation experiment is 500 ms, this section is dedicated to the optimal input excitation optimization problem. Consequently, the return to origin task for the EMB to return to its initial state is excluded, and the experiment duration is reduced to 300 ms. In other words, the objective of the experiment is to drive the motor within 300 ms while satisfying the environmental constraints and to compare the final value of the objective function obtained by the agent trained with the two parameter initialization methods in order to determine which one has superior performance.

5.2.1 Design of the reward function

For the single parameter case, the FIM matrix simplifies to a one-dimensional matrix. In this scenario, the three optimality mentioned earlier, A-, E- and D-optimally are equivalent, as $|M| = tr(M) = \lambda_{min}(M) = M$. This greatly simplifies the optimization problem, as the objective reduces to maximizing the value of the FIM element itself. According to Equation (4.21), the FIM undergoes a rank-one update, eliminating the need for initialization and normalization. Since the agent's goal is to maximize information content by maximizing the FIM value, the total return of a full experiment is equivalent to the final FIM value. Thus, the step reward function in the EMB environment can be directly derived as

$$r_k = M_k - M_{k-1} \tag{5.13}$$

The parameters used for the system modeling are listed in Table 4.1. As described in Section 4.6, in the gym environment setup, if the system constraints are violated, for example if the motor exceeds its maximum position or speed, it is considered an unsafe state, and the current episode will be terminated. Therefore, the agent must learn to avoid entering such states, which should also be achieved by the reward function design.

However, for the system's input voltage constraints, no specific penalty is added in the reward function because the continuous actions generated by PPO is sampled from a probability distribution based on the mean value of the action. Due to the randomness of the sampling, it is not guaranteed that the action range obtained after each sampling is within the constraints. Therefore, the solution is to use the $clip$ method to clip the voltage output from the agent, ensuring it does not exceed the maximum voltage that can be input to the system.

Considering the problem of constraints on the state-space parameters of the system, a very intuitive idea is that each time the motor moves to a risky state and the episode is terminated, the agent receives a penalty to discourage it from entering such states. This approach has been validated in many classical reinforcement learning problems. According to the calculations in the previous section, the maximum single-step FIM increment for a system running continuously at full speed for 500 ms is approximately 1.3×10^5, regardless of the motor's positional constraints. Therefore, a more realistic penalty value of -4×10^4 is chosen for constraint violations.

However, this penalty constitutes a sparse reward, as the motor is only penalized when it reaches a risky state, which increases the difficulty for the agent to learn effectively. To address this, additional considerations were incorporated into the reward function design. In the EMB environment, a concept of "risky state" is introduced: when the motor position exceeds a predefined threshold $x_{0,d} = 75$, it is considered to have entered a risky state. The motor is then penalized continuously in proportion to the square of the exceeded position value. Since high motion speed is essential for recognizing viscous friction, as shown in the previous section, no penalty is applied to the motor velocity x_2. In summary, the step reward function for reinforcement learning was designed as

$$r_k = \begin{cases} -4 \times 10^4, & \text{if in unsafe state,} \\ M_k - M_{k-1} - 200(x_1 - x_{1,d})^2, & \text{if in risky state,} \\ M_k - M_{k-1}, & \text{otherwise.} \end{cases} \tag{5.14}$$

Here, r_k is the reward at time step k, $M_k - M_{k-1}$ represents the step reward corresponding to the increment of the fisher information matrix, x_1 denotes the motor position, and $x_{1,d}$ is the threshold for the risky state.

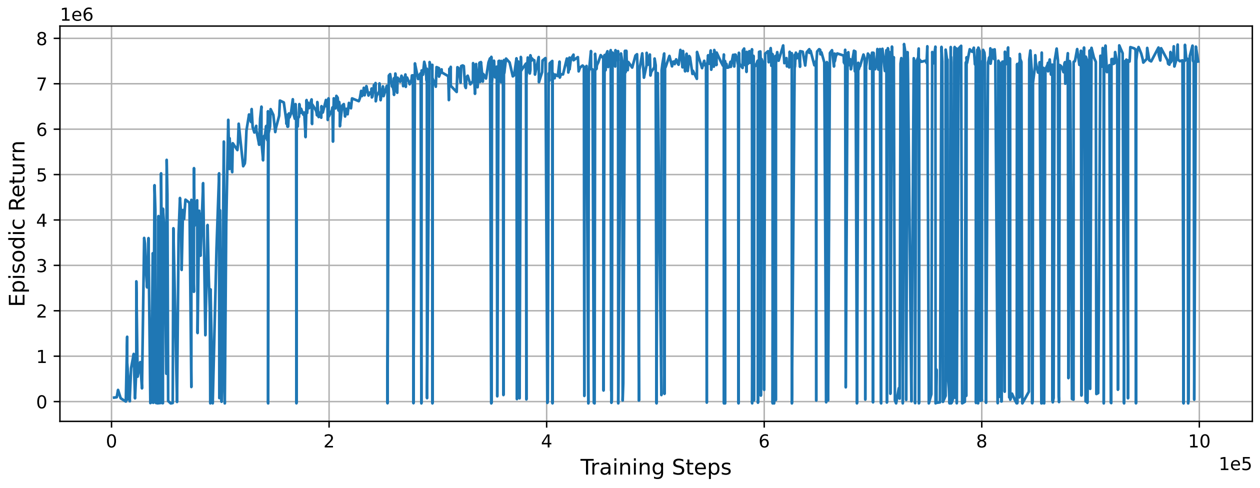

Figure 5.2: Episodic return during training with fixed parameter initialization method

5.2.2 Performance of the fixed parameter initialization method

The first training experiment was conducted using fixed parameter initialization, where the viscous friction coefficient was set to a constant value of $f_v = 2.16 \times 10^{-5}$. Since the tested FIM element in the previous section reached values as high as 10^5, which is significantly larger than the range of the viscous friction parameter, the observations space were normalized by subtracting their running mean and dividing by their variance. This reprocessing step helped the neural network learn more effectively.

The training consisted of 1 million steps, with 4 parallel EMB environments running 2048 steps per environment in each rollout phase. The collected data buffer of length 8192 was then used to update the network. A learning rate of 0.001 was applied, with the learning rate annealing enabled to reduce the rate gradually during training. The discount factor γ was set to 0.99 and Generalized Advantage Estimation (GAE) with $\lambda = 0.95$ was used to stabilize advantage estimates [56]. To ensure stable training, the key parameter of PPO, clipping coefficient ϵ was set to 0.2. The training was conducted on the NVIDIA RTX3000 and takes approximately 35 minutes to complete the 1 million steps training.

Figure 5.2 illustrates the episodic return during the training process, showing the agent's performance over time. The training process can be divided into two distinct phases. At the beginning of the training, i.e., from 0 to 3×10^5 steps, the episodic return exhibits a rapid upward trend despite experiencing small oscillations in the middle. The rapid increase in the value of the corresponding one-dimensional FIM matrix indicates that the agent is actively exploring and learning how to adjust the input voltage of the EMB to improve the recognition of the viscous friction parameter.

Starting from 3×10^5 steps, the upward trend of the episodic return slows down, and its maximum value gradually stabilizes. This indicates that the agent reduces its exploration and progressively shifts toward exploiting the learned strategy. Additionally, it can be observed that the episodic return occasionally drops suddenly during the training process. This is due to the agent's input in these episodes causing the motion state of the EMB motor to violate constraints, such as exceeding the position or speed limits. As a result, the agent is penalized according to the reward formula (5.14), the episode is terminated and this environment is reset.

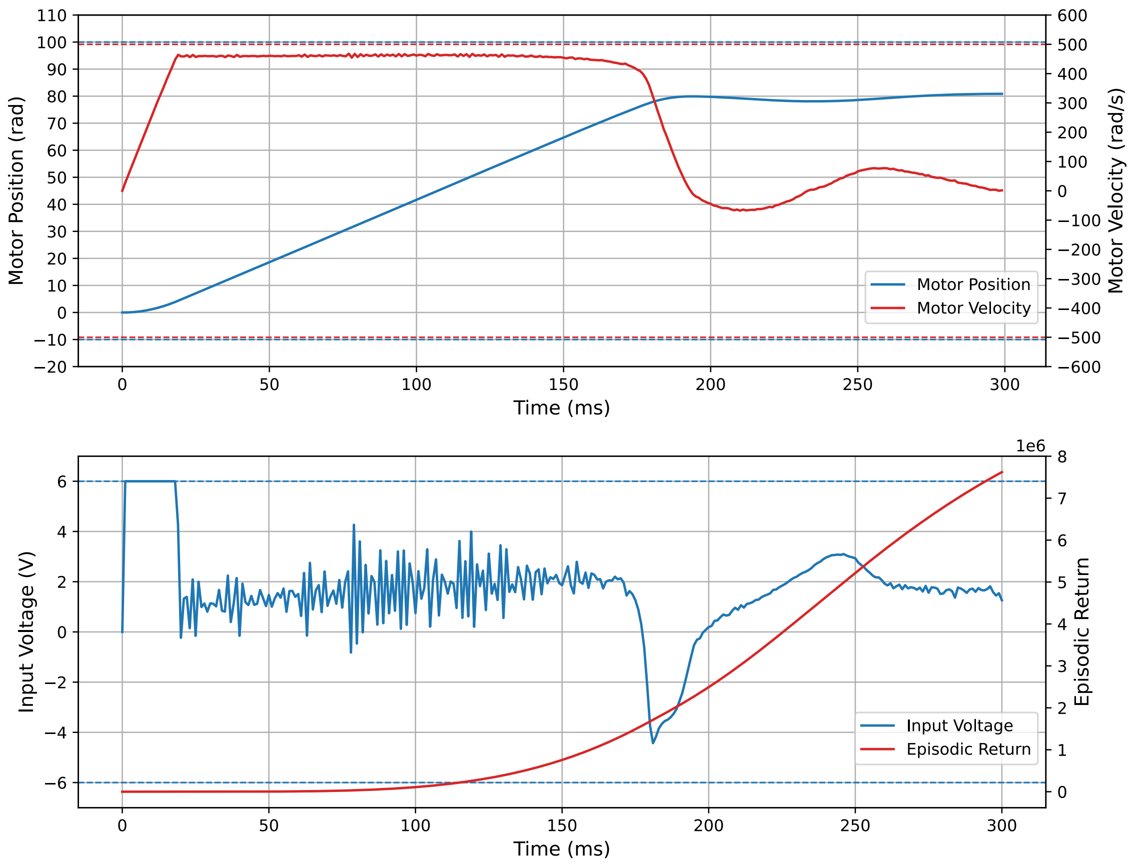

Figure 5.3: Performance of the agent trained with fixed parameter initialization method

The performance of the agent trained with fixed parameter initialization method is illustrated in Figure 5.3. At the beginning of the experiment, the agent directly inputs the maximum current to the motor, causing it to accelerate rapidly. After approximately 20 ms, the motor velocity approaches the upper speed limit of 500 rad/s. To prevent exceeding this limit, the agent reduces the system input voltage, ensuring the motor velocity remains just below the maximum allowable value. Between 20 ms and 150 ms, the motor rotates positively at a velocity close to 500 rad/s, with some minor vibrations in the agent's actions. This behaviour is consistent with the theoretical analysis conducted in the preceding subsection, which determined that maintaining the motor velocity at its maximum value is an optimal strategy for identifying the viscous friction parameter. These findings indicate that the agent has effectively learned through training, how to maximize the objective function of the optimization problem, namely the FIM element value.

At around 180 ms, the motor position exceeds the risky threshold defined in the environment. In response, the agent reduces the motor velocity to approximately zero by adjusting the input voltage to a negative value and maintains velocity around 0 until the end of the experiment. This behavior demonstrates the agent's ability to avoid violating the positional constraints of the motor.

Based on the analysis in the previous subsection, the sensitivity of the state parameter to viscous friction can be analogized to a mass-spring-damping system, where the motor velocity acts as the input to the system. At 200 ms, if the motor velocity continues to be greater than zero, the positional constraints of

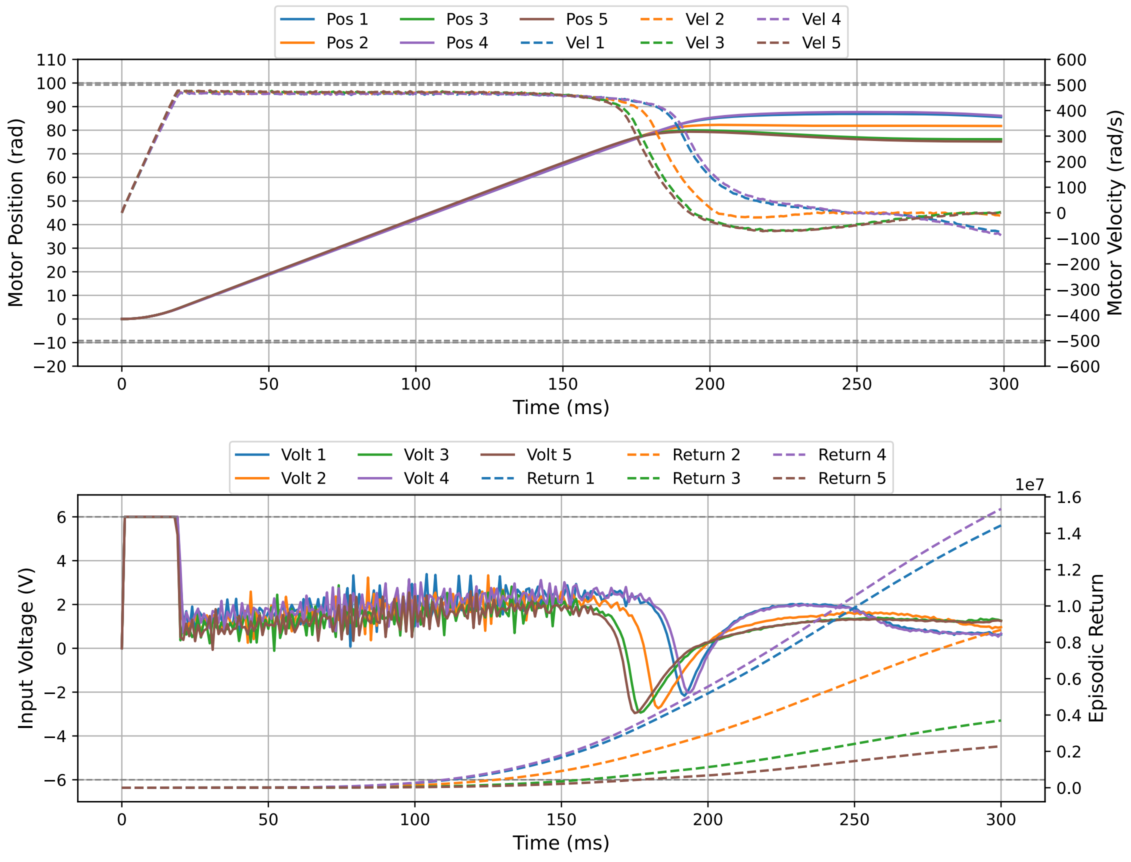

Figure 5.4: Performance of the agent trained with random distribution-based initialization method in 5 random experiments

the EMB system will be violated. On the other hand, if the motor velocity becomes less than zero, the reduction of the state parameter a in the mass-spring-damping system will accelerate. According to Equation (5.12), decreasing the value of a leads to a reduction in the incremental of the FIM, which will decrease the final FIM value that can be obtained in the end. Therefore, the agent's decision to maintain the motor velocity around zero at this stage is reasonable and aligns with the optimization objective.

In summary, the performance of the agent trained with the fixed parameter initialization method demonstrates its effectiveness in solving the input excitation optimization problem for viscous friction parameter estimation. The results show that PPO, with full information observation, can successfully perform input excitation optimization, providing a solid baseline for continued works in this thesis.

5.2.3 Performance of the random distribution-based parameter initialization method

In this section, training is conducted using the random distribution-based parameter initialization method. The training environment and hyperparameters are identical to those used in the fixed parameter initialization method, with the exception that during the training process, the viscous friction coefficient is sampled from a uniform distribution when each time the environment is reset, as shown in

Figure 4.7. Considering the a priori value of the viscous friction coefficient $f_v = 2.16 \times 10^{-5}$, a wide distribution range of $(1 \times 10^{-5}, 5 \times 10^{-5})$ was chosen for this session.

The performance of the trained agent was evaluated in the EMB environment by conducting 5 experiments. Before the start of each experiment, a random value of viscous friction was sampled from the same uniform distribution as it used for training. The experimental results are presented in Figure 5.4. It can be observed that, although each experiment corresponds to a different viscous friction value, the agents trained using the random distribution-based parameter initialization method exhibit consistent performance. During the first 180 ms, the agent initially applies the maximum action to accelerate the motor rapidly and subsequently adjusts the action based on the sampled viscous friction coefficient to maintain the motor at the critical velocity. This behavior is similar to the behavior of the agent trained with fixed parameter initialization. After 180 ms, the agent reduces the motor velocity via voltage adjustments, ensuring that the EMB system constraints are not violated.

To better compare the performance of the two parameter initialization methods, both agents were tested in 40 experiments spanning the parameter range $(1 \times 10^{-5}, 5 \times 10^{-5})$. In each experiment, the value of the viscous friction coefficient was incrementally increased by 1×10^{-6}. The test results are illustrated in Figure 5.5 . It could be seen that, for both agent, the return increases as the viscous friction coefficient grows. As the red box in the figure shows, the agent trained with the fixed parameter initialization method achieves a slightly higher return only near $f_v = 2.16 \times 10^{-5}$, corresponding to its priori parameter value used for training. However, across the remaining parameter range, its performance is worse than the agent trained using the random distribution-based parameter initialization method. Additionally, negative returns are observed in cases of low viscous friction for the fixed parameter agent, indicating constraint violations that result in the early termination of the 300 ms experiment. Thus, the random distribution-based parameter initialization method makes the agent more robust to changes in system parameters. It has better performance across nearly the entire parameter range and avoids constraint violations.

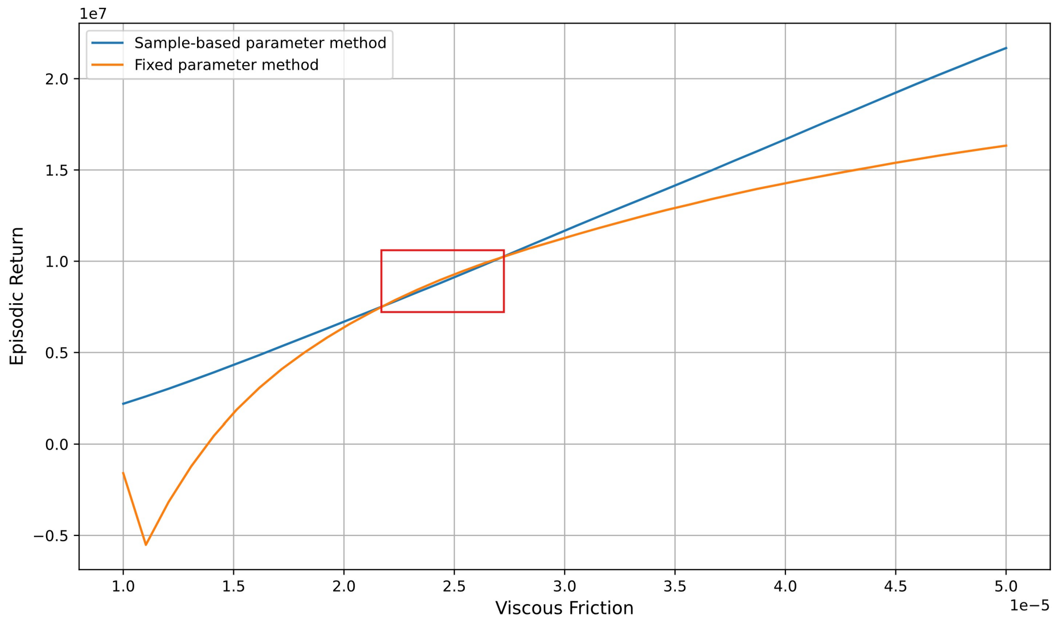

Figure 5.5: Performance comparison of two parameter initialization methods in viscous friction range tests

5.3 Results of the asymmetry PPO framework with return to origin task

In the previous section, the advantages of the random distribution-based parameter initialization method were demonstrated in the classical PPO framework. However, the assumption that the actor has access to all information in the observation space is unrealistic. Therefore, this section investigates the performance of an asymmetric PPO framework, which more closely reflects real-world scenarios.

The random distribution-based parameter initialization method is still employed. However, to match actual experimental requirements, the information available to the actor network is limited to the measured motor position x_1, motor velocity x_2, and time step k. The actor network is not provided with the viscous friction f_v, which sampled from the distribution during training or testing. In the meanwhile, the critic network have the access of all the information, as showed in Table 4.2.

Furthermore, additional content related to realistic experimental requirements has been incorporated. According to these requirements, the motor should return to the origin of velocity and position after a 500 ms experiment to prepare for the next experiment. While a simple PID controller could achieve this task, this approach is not placed at the first place in this thesis. The reason is that the process of driving the motor back to the origin position still generates valuable information for parameter estimation. While a PID controller can perform the reset efficiently, it cannot optimize the motor trajectory during the reset, leading to a loss of the potential information. Therefore, in this section, through the reasonable construction of the reward function of reinforcement learning, the learning ability of the reinforcement learning agent is fully utilised, so that the agent is able to optimize parameter estimation even under complex constraints, and to solve the optimisation problem which is closer to the actual reality.

For the consideration of the overall experiment, the parameter excitation experiment is mainly divided into two phases. Given that the total experiment duration is 500 ms, the motor's maximum velocity is 500 rad/s, and its maximum angle is 100 rad, it can be calculated that approximately 200 ms is required for the motor to return to the origin from the maximum position at maximum velocity. Therefore, in Phase 1 (0-300 ms) focuses on exploration, where the agent stimulates the EMB system within constraints to optimize the Fisher information matrix. In Phase 2 (300-500 ms), the main content of the agent is to return the motor velocity and position to the zero, in which the input excitation is also optimised to maximise the information utilisation. Using the asymmetric PPO architecture, two reward function design approaches dare explored: Adaptive reward function design and Trajectory-based reward function design.

5.3.1 Asymmetric PPO framework with adaptive reward function design

When designing the reward function for the return to origin task, the most intuitive approach is to add a penalty based on the deviation of motor speed and position from the origin at the end of the 500 ms episode, often using a quadratic penalty term. Although this idea is simple and easy to implement, the designed penalty term is sparse reward, which is not conducive to the learning of reinforcement learning agent. Therefore an adaptive reward function was developed. Since the Equitation (5.14) achieved good results in the previous task, it was used as the basis of the adaptive reward function design. The specific design idea is that when timesteps are less than 300, the reward function is consistent with Equitation (5.14). For timesteps greater than 300, with the increase of time steps, the weight of FIM in the reward function gradually decreases, and the weight of the penalty term for the deviation from the origin increases. Hence the specific reward function under this method is designed as

$$
r_k = \begin{cases} -1 \times 10^7, & \text{if in unsafe state,} \\ \left(1 - \left(\frac{k}{T}\right)^2\right) \Delta M_k - 200(x_0 - x_{0,d})^2 - 0.1 \left(\frac{k}{T}\right)^2 \left((10x_0)^2 + x_1^2\right), & \text{if in risky state and } k > 300, \\ \Delta M_k - 200(x_0 - x_{0,d})^2, & \text{if in risky state and } k \leq 300, \\ \left(1 - \left(\frac{k}{T}\right)^2\right) \Delta M_k - 0.1 \left(\frac{k}{T}\right)^2 \left((10x_0)^2 + x_1^2\right), & \text{if in safe state and } k > 300, \\ \Delta M_k, & \text{otherwise.} \end{cases} \tag{5.15}
$$

In this function, $\Delta M_k = M_k - M_{k-1}$ is the incremental change in the Fisher information matrix. The variable k is the current step count, and T indicates the total number of steps in an episode. The function adapts its behavior based on the step count. In the first 300 steps, the reward focuses on maximizing the FIM by directly linking it to ΔM_k. After 300 steps, penalization terms are introduced to guide the motor back to the origin. At the beginning the penalties is weak, only $\left(\frac{300}{500}\right)^2 = 0.36$ of the the penalties is added to the function. But it increasing quadratically as the episode progresses and quickly grow to 1 in the end.

After designing the reward function, training was conducted with the same settings and hyperparameters used in the previous section. After training, the agent was tested 5 times in EMB environments with random parameters sampled from a uniform distribution, same as what was done previously. The difference in this section is that the sampled viscous friction parameter remains unknown to the agent, and the agent's decisions are only based on sensor data, not the actual motor state parameters. The test results are illustrated in Figure 5.6. Since the reward function is designed in segments, to better analyze the reward function's effectiveness, the episodic return over time is not shown in this figure. Instead, the trend of step rewards is emphasized.

It can be observed from the figure that the agent trained with the asymmetric PPO architecture demonstrates a similar strategy to the agent trained with the classical PPO after the start of the experiment. The motor was driven at a higher velocity from 0-200 ms, but it can be seen that the velocity in this interval is around 400 rad/s, which is lower than in the previous experiment. According to the earlier analysis, lower motor velocity reduce the value of FIM. However, considering the asymmetric network architecture, the actor network can only access measurements of motor position and velocity in the new experiment. Since the measurements are noisy, in extreme cases, even if the actor network perceives motor velocity below 500 rad/s, the system may still terminated because the actual motor velocity exceeds the constraint. Therefore, it is reasonable for the agent to adopt a more conservative strategy.

Starting from 200 ms, as the motor position enters the risky interval, the agent reduces motor velocity by reducing the voltage input, during which the penalty of the risky interval begins to take effect. In experiment 2, the step reward becomes negative due to the low viscous friction value, resulting in the penalty between 200 ms and 300 ms exceeding the reward gained from FIM. From 300 ms, the penalty term in the reward function for returning to origin task is triggered, with increasing weight over time. It can be observed that the motor velocity starts to increase, reducing the deviation from zero velocity. However, this also causes a slower return of motor position to zero, resulting in a final motor position of around 35 rad.

An overall trend can be observed in the figure is that the agent's actions drive the EMB motor to running with similar position and velocity trajectories regardless of the difference between sampled viscous

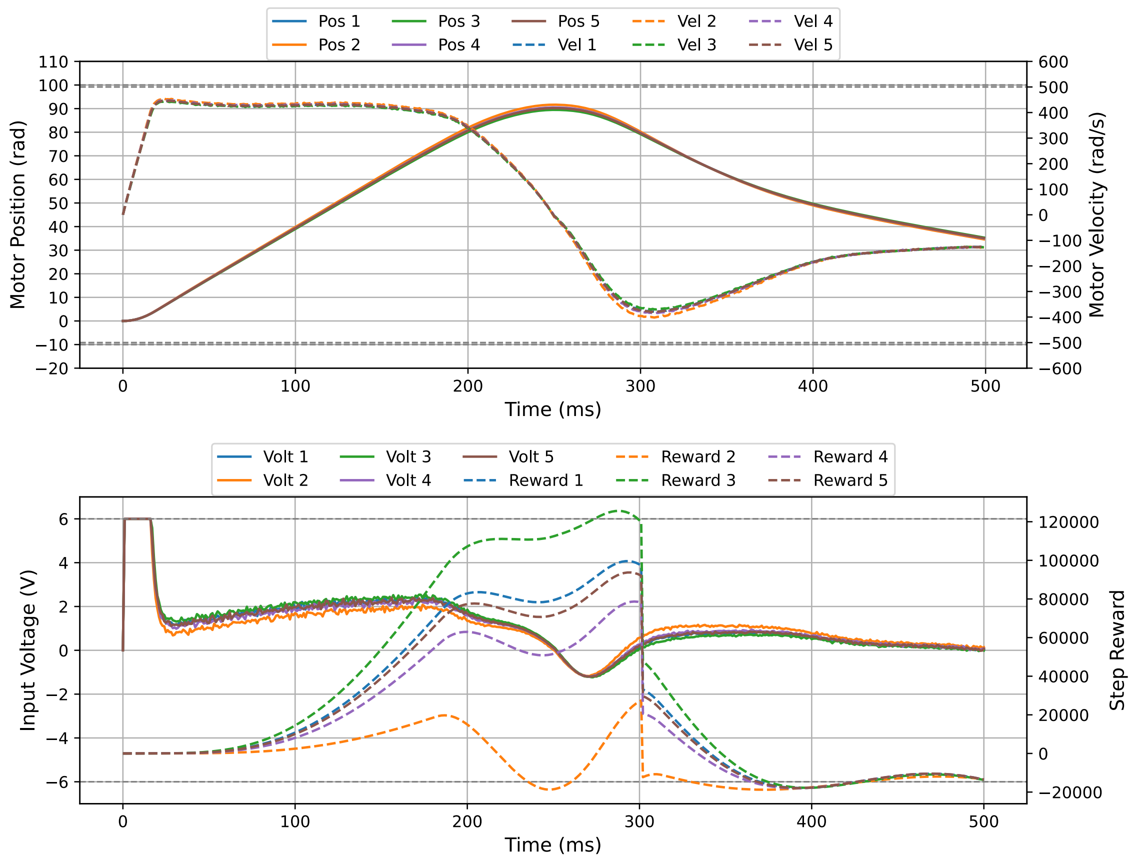

Figure 5.6: Performance of the agent trained with adaptive reward function in 5 random experiments

friction value. The agent's strategy during the exploration phase is consistent with the previous optimal strategy analyses. This demonstrates the robustness of the agent trained using the asymmetric PPO network to changes in system parameters and shows the model's strong generalization capability.

On the other hand, there is conflict between velocity and position in penalty term in the reward function design during the return-to-zero phase. Specifically, rapidly decreasing the motor position requires maintaining a high drive-back velocity, but higher velocity also causes bigger penalties in the reward function, as mentioned earlier. Therefore, the individual coefficients in the reward function are modified many times, the trained agent could not achieve better performance in the return to origin task. Hence, a new approach, trajectory-based reward function design will be presented in the following text.

5.3.2 Asymmetric PPO framework with trajectory-based reward function design

In previous training, the agent always operated at maximum velocity at the beginning of the viscous friction parameter estimation experiment, leading to similar motor velocities and positions at the end of the first stage. Based on this observation, a new reward function design approach, called trajectory-based reward function design, is proposed.

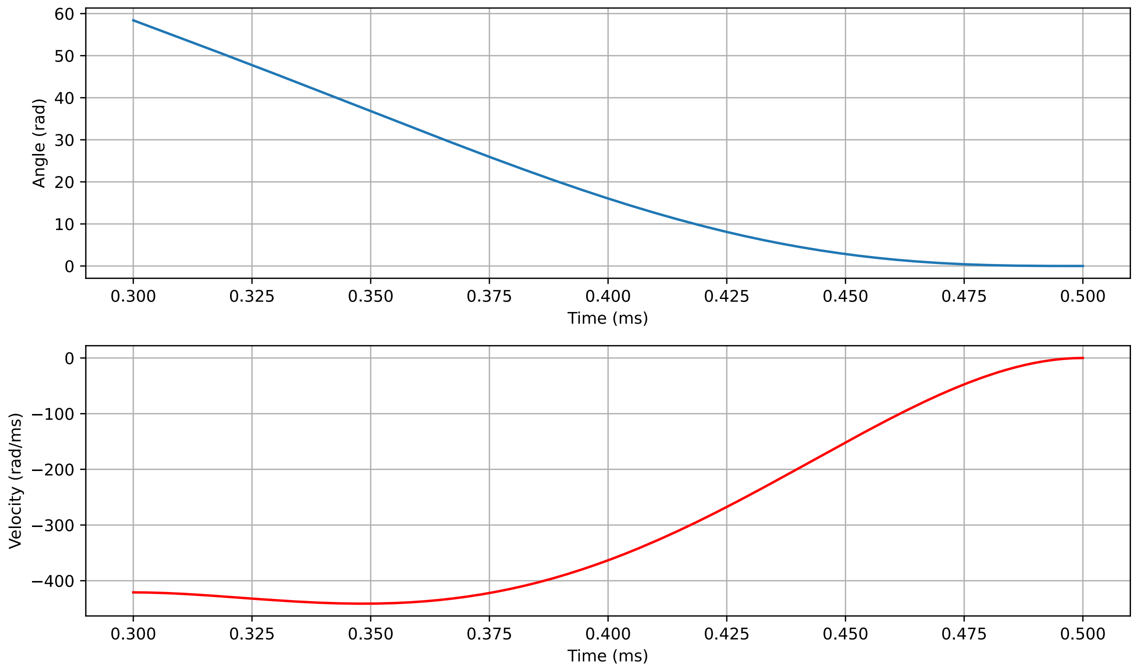

Figure 5.7: Quintic polynomial trajectory for return to origin task

In the trajectory-based reward function design, the 500-ms experiment is still divided into two phases, with the first phase of the reward function design remaining the same as in the previous section. The difference lies in the second phase of the return to origin task: instead of penalizing the agent based on the deviation from the zero point, a quintic polynomial trajectory is designed, which the agent attempts to follow the trajectory and finally returns smoothly to the zero point.

Quintic polynomial trajectory planning is a method that uses a quintic polynomial based on six boundary conditions: position, velocity, and acceleration at the start and end points, to generate smooth, continuous trajectories. This method is widely used in robotics, automotive, and aerospace trajectory planning [57]–[59]. For the motor return to origin task, when each experiment is carried out to 300ms, the boundary conditions of the quintic polynomial are set based on the current motor velocity, position, and acceleration, as well as the target boundary conditions at the end of the experiment, where the motor velocity, position, and acceleration are all zero. The following polynomial equations are solved using the $np.linalg.solve()$ function

$$\begin{aligned} \theta(t) &= k_0 + k_1 t + k_2 t^2 + k_3 t^3 + k_4 t^4 + k_5 t^5 \\ \dot{\theta}(t) &= k_1 + 2k_2 t + 3k_3 t^2 + 4k_4 t^3 + 5k_5 t^4 \end{aligned} \tag{5.16}$$

where $\theta(t)$ represents motor angle trajectory and $\dot{\theta}(t)$ represents motor velocity trajectory. Whit this function, a smooth EMB motor motion trajectory can be obtained. Figure 5.7 illustrate an example of the trajectory based on this design.

During the second phase of the experiment, penalties are applied based on the deviation of the EMB motor velocity and position from the preset trajectory. The increment of the FIM value during this

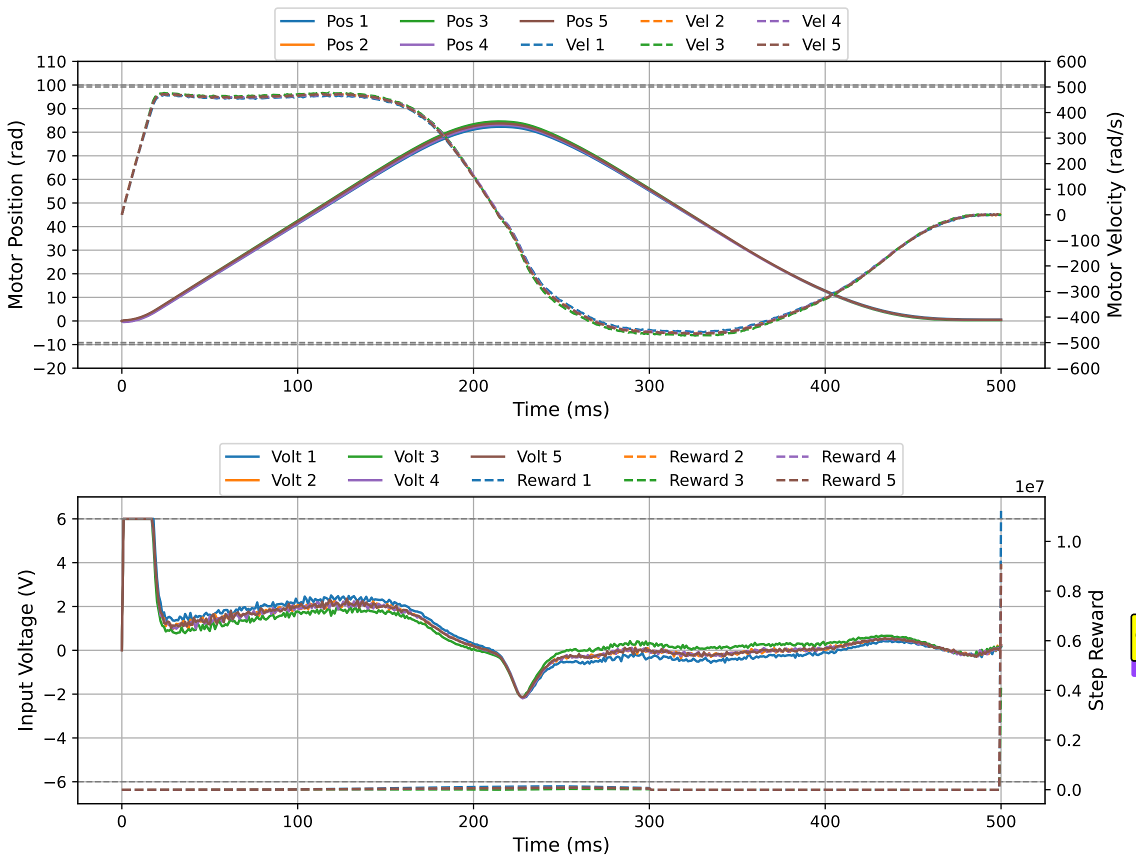

Figure 5.8: Performance of the agent trained with trajectory-based reward function in 5 random experiments

interval is also recorded, but it is not immediately fed back to the agent as a step reward. Instead, the FIM reward from 300 ms to 500 ms is provided to the agent at the end of the experiment, if the motor successfully returns to the origin point. The purpose of this design is to resolve the conflict between exploring FIM information and returning to origin, where the agent prioritizes following the trajectory to return to the zero point. Only when this task is achieved, then the agent will consider maximizing the incremental FIM value obtained during the return phase. Ideally, the agent first learns to follow the trajectory back to zero and then try to deviates from the trajectory to maximize the potential FIM, with the guarantee that it can return to origin.

After designing the reward function, training was carried out. Consider of the potential two learning phases of the agent in return to origin task, which significantly increase the training difficulty, the total number of training steps was increased to 2 million. To monitor the agent's learning progress, the network parameters of both the actor and critic were saved at regular intervals.

The trained results are displayed in Figure 5.8. It can be observed that the trained agent drives the motor to follow the same trajectory during the parameter estimation experiment, demonstrating high robustness to system parameters as well as good generalization ability. Starting from 300 ms, the agent's actions guide the motor along a quintic polynomial trajectory, eventually bringing the velocity and position of the motor to zero regardless of the system's viscous friction parameters. According to the

Agent	Phase 1	Phase 2	End Step
Agent A	1.13×10^7	-1.41×10^4	8.05×10^6
Agent B	1.16×10^7	-4.62×10^4	8.66×10^6

Table 5.1: Accumulated mean reward for agents in different phases of viscous friction range tests

reward function setup, the agent is rewarded with the cumulative FIM increment in the second phase if it successfully completes the return-to-zero task, which explains why the step reward at 500 ms in Figure 5.8 is significantly higher than at other times.

To better observe the effect of training on trajectory optimization during the second phase of the experiments, training was repeated using the same parameters and random seed, and Agent A, which saved after 1 million steps training, was compared with Agent B, which trained for 2 million steps. Table 5.1 shows the performance of the two agents in viscous friction range tests, with 40 experiments spanning the parameter range $(1 \times 10^{-5}, 5 \times 10^{-5})$. It can be seen that Agents A and B perform similarly in Phase 1, both successfully driving the EMB motor at high velocity. However, Agent B incurs a higher penalty in Phase 2, indicating that while Agent B had trained for longer time, its trajectory-tracking ability is lower compared to Agent A. Nonetheless, Agent B receives a higher end step reward than Agent A, meaning that it achieves a trade-off between trajectory tracking and maximizing FIM information by deviating from the preset trajectory to enhance FIM performance and improve parameter estimation.

Since the actions during training are sampled from the policy and the viscous friction value for each episode is also sampled from the distribution, it is important to ensure that the improved training results

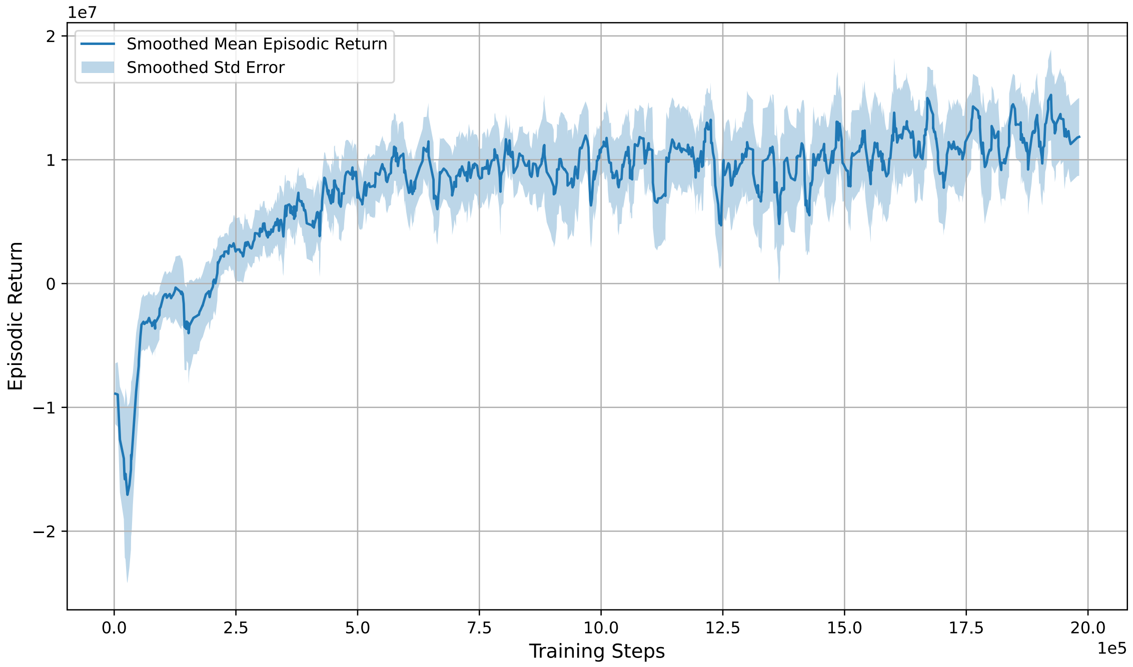

Figure 5.9: Training curve of asymmetric PPO framework with trajectory-based reward function across 5 random seeds

mentioned above are not obtained by chance. This was done by conducting multiple experiments using multiple random seeds, with results shown in Figure 5.9. It can be seen that the training curve rises rapidly in the early stage and gradually slows in the middle and late stages. Since the episodic return is directly related to the parameter values, the curve has been smoothed and the small oscillations are acceptable. The standard error of the five training sessions also shows the robustness of the asymmetric PPO network to different seeds, reflecting the good learning ability of the agent in different random situations.

In summary, the agent trained using the asymmetric PPO framework with trajectory-based reward function design demonstrates excellent performance. It not only achieves input excitation optimization for parameter estimation with limited information as in the actual experimental environment, but also shows robustness to variations in system parameters and effectively performs the return to origin task.

6 Approaches of the multi parameter excitation

The previous chapter demonstrated that reinforcement learning is fully capable of optimizing input excitation for system parameter estimation. In this chapter, the approach was extend to multi parameter excitation optimization. Specifically, in addition to identifying the viscous friction parameter of the EMB system, the stiffness characteristic parameter is also introduced. Reinforcement learning is now used to solve the input excitation optimization problem for estimating multiple system parameters simultaneously.

As the number of parameters increases, the dimension of the Fisher information matrix also rises, which can make the learning process more challenging for the agent. Therefore, in this chapter, different approaches are investigated to better address the problem of multi-parameter input excitation.

6.1 Sensitivity analyse with multi parameter

FIM stays always at the core of parameter estimation and excitation optimization and the construction of the FIM based on the sensitivity calculation. Therefore, before directly applying reinforcement learning to this problem, the properties of the second-order FIM matrix are analyzed. Since significant differences in parameter values may exist, the parameters need to be scaled when calculating the FIM. Referring to the analytical calculation in Chapter 5.1, the calculation is performed by replacing $\bar{f}_v$ with $\bar{\theta} = [\bar{f}_v, \bar{k}_1]^T$. Thus, in the case of both viscous friction and stiffness characteristics parameters included, Equation 5.6 becomes

$$\frac{\partial x_{d,k+1}}{\partial \bar{\theta}} = \frac{\partial x_{d,k}}{\partial \bar{\theta}} + t_s \left(\begin{bmatrix} 0 & 1 \\ -\frac{\gamma k_1}{J} & -\frac{f_v}{J} \end{bmatrix} \frac{\partial x_{d,k}}{\partial \bar{\theta}} + \begin{bmatrix} 0 & 0 \\ -\frac{f_v x_2}{J} & -\frac{k_1 \gamma x_1}{J} \end{bmatrix} \right) \tag{6.1}$$

Note that, according to the EMB model setup, the motor position is considered to be at the contact point when it is 0. The stiffness curve only has values when the motor angle is greater than 0, i.e., when the brake lining is in contact with the brake drum. Therefore, the corresponding term in Equation (6.1) only exists when x_1 is greater than zero; otherwise, the term is zero.

If $\frac{\partial x_{d,k}}{\partial \bar{\theta}}$ is defined as

$$\frac{\partial x_{d,k}}{\partial \bar{\theta}} = \begin{bmatrix} a & c \\ b & d \end{bmatrix} \tag{6.2}$$

Similar to Equation (5.12), the increment of the Fisher information matrix at each time step in two-parameters case becomes

Parameter	Range
f_v	$[1.0, 5.0] \times 10^{-5}$
k_1	$[25, 50]$

Table 6.1: Parameter ranges for the uniform distribution

$$
\begin{aligned}
M_{\bar{\theta},k} &= \left(\frac{d\bar{h}_k}{d\bar{\theta}}\right)^T R^{-1} \left(\frac{d\bar{h}_k}{d\bar{\theta}}\right), \\
&= \begin{bmatrix} \frac{a}{100} & \frac{b}{500} \\ \frac{c}{100} & \frac{d}{500} \end{bmatrix} \begin{bmatrix} 10^6 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{a}{100} & \frac{c}{100} \\ \frac{b}{500} & \frac{d}{500} \end{bmatrix}, \\
&= \begin{bmatrix} 100a^2 + \frac{b^2}{250000} & 100ac + \frac{bd}{250000} \\ 100ac + \frac{bd}{250000} & 100c^2 + \frac{d^2}{250000} \end{bmatrix}
\end{aligned}
\tag{6.3}
$$

From analyzing the above equations, it can be observed that the main diagonal elements of the FIM increment matrix $M_{\bar{\theta},k}$ correspond to two mass-spring-damping systems with identical mass $m = J$, spring constant $k = \gamma k_1$, and damping coefficient $c = f_v$. But these two system have different input: one related to viscous friction with a system input of $-f_v x_2$, and the other related to the stiffness parameter with a system input of $-k_1 \gamma x_1$. Based on the analysis in Section 5.1, for the stiffness parameter related mass-spring-damping system, it can be deduced that increasing the motor position x_1 results in a larger increment of the FIM. Therefore, the stiffness parameter can be estimated more effectively at higher motor position.

Unlike the previous single parameter case, the increase in the dimension of the FIM means that the individual element values cannot be directly used as the objective function for optimization. The FIM of the EMB system also shows correlation between the state parameters of the two mass-spring-damping systems, as terms like $100ac + \frac{bd}{250000}$ show up in Equitation (6.3). Even though the system identification involves adding only one additional parameter, it significantly increases the complexity of theoretical FIM analysis. In such cases, reinforcement learning's ability to train based on sampling helps balance the relationship between EMB motor speed and position during system parameter estimation, thereby optimizing the FIM-based objective function and providing optimal input trajectory.

6.2 Performance of the reward function design based on D-Optimally

After analyzing the theoretical calculations of the sensitivity and second-order Fisher information matrix, D-optimality was chosen as the objective function for reinforcement learning due to its ability to optimize multiple parameter dimensions simultaneously and its invariance to linear transformations. For the multi-parameter estimation problem, the random distribution-based parameter initialization method was still adopted. The viscous friction and stiffness parameters were each initialized according to normal distributions, with their respective ranges shown in Table 6.1.

The increase in the number of the to be estimated parameters also directly affects the observation space's dimension. While the observation space for the actor network remained unchanged, the critic network observes an additional system parameter due to the increased dimension of the FIM, as well as three additional FIM elements. However, since the FIM is a symmetric matrix, it is not necessary to store all the elements in the observation space. Only store the upper-triangular elements already provides the critic network with full information.

6.2.1 Uderpeformance of the original step reward function

The step reward, calculated as shown in Equation (4.13), was used directly for the reward function in reinforcement learning. At the beginning of the two parameter case, in order to let the agent focus on the parameter estimation problem, the return-to-origin task was temporarily removed, reducing the experiment duration to 300 ms. Training was conducted, but the results were poor. The reinforcement learning agent did not learn how to excite the EMB system properly and frequently encountered issues with system constraints being violated.

6.2.2 Modification of the step reward function

Upon further analysis, the reward function's structure for the current task differed significantly from the previous one-parameter problem. In the previous task, the value of the step reward was very small at the beginning and gradually increased as the experiment progressed. However, for the $\log|M|$ function, since the M matrix was initialized with a small diagonal value, even a slight change in FIM elements can led to big change in $|M|$, cause a significant step reward due to the nature of the logarithmic function. With the steps in the experiment growing, the logarithmic function reduced its impact on the step reward, even with further increases in $|M|$, because $|M|$ had already grown large.

Therefore, the reward function construction based on D-optimality was reconsidered, and two methods were proposed. Both ideas focus on amplifying the single-step rewards in Equation (4.13) to increase the impact of changes in $|M|$ on reinforcement learning rewards.

In Method 1, a coefficient $\alpha = 1 \times 10^4$ was used to amplify the reward in Equation (4.13)

$$r_k = \alpha(log|M_k| - log|M_{k-1}|) \tag{6.4}$$

while Method 2, move one step more than Method 1, an additional parameter $\beta = 2$ to amplify the original reward quadratically

$$r_k = \alpha(log|M_k| - log|M_{k-1}|)^{\beta} \tag{6.5}$$

The training results using the two modified reward functions are shown in Figure 6.1. It can be observed that, regardless of whether Method 1 or Method 2 was used, the agent's behavior varied periodically, similar to a square ware inputs. In Method 1, the action of the agent had a shorter duration at the peak, resulting in lower motor position and velocity amplitudes compared to Method 2. In addition, the action amplitude of Method 1 decreased towards the end of the the experiment.

In order to fairly compare the performance of these two methods, the step reward calculations shown in the Figure 6.1 are based on the original Equation (4.13), without scaling based these two method. Comparing the step reward trajectories of the two methods, it is evident that the step reward values at the beginning and end of the experiment were relatively similar. At the start, the calculation of FIM based on logarithm of the determinant produced a high step reward value, and the difference between the two methods was almost negligible at the end. The primary difference in optimizing $|M|$ occurred between 20 ms and 100 ms, resulting in lower performance for Method 1 compared to Method 2. This phenomenon was due to the more conservative behavior of the agent trained using Method 1, which avoided taking high voltage inputs for longer periods, leading to lower speed and position of the EMB motor compared to Method 2.

Based on the behaviour of the agent with both methods, it can be suggested that periodic system inputs can efficiently help the parameter estimation for the viscous friction and stiffness parameters. Therefore, a manually designed input trajectory in square ware form, with a period of 20 ms and an amplitude of -6 to 6 was introduced for comparison.

From the figure 6.1, it can be observed that as the experiment continued, the step rewards generated by the manually designed input trajectory gradually exceeded those of the other two agents. This indicates that, although the two trained agents obtained similar strategies and the strategy based on Method 2 performed better than that based on Method 1, their performance was still lower than the manually designed trajectory. This result shows that D-optimally criterion has limitations in the multi-parameter identification problem of EMB systems, and further research base on other optimally criterion is necessary.

6.3 Performance of the reward function design based on E-Optimally

Compare the result between the different reward functions.

Also compare to the different input trajectories, such as step inputs, ramp inputs, and sinusoidal input etc.

Problem founded, the way try to fix them. Sth works, But still open problem

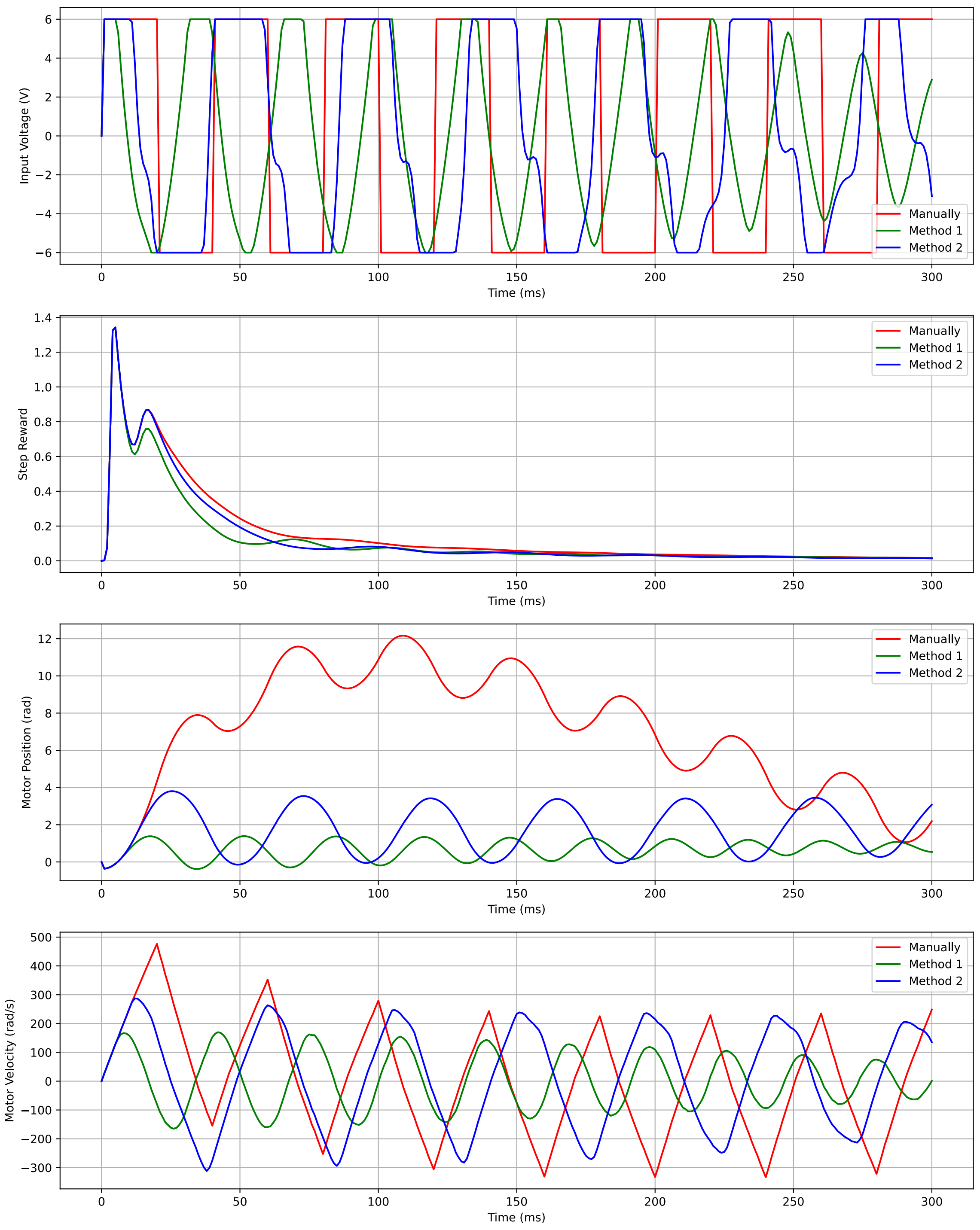

Figure 6.1: Performance comparison of agent based on reward-modified method and manually designed input trajectories

7 Conclusion and Outlook

Conclusion: compare to the related works, we are doing with mechanical area, we have some result But an estimator is still needed.

Bibliography

[1] C. L. J. Line, *Modelling and control of an automotive electromechanical brake*. University of Melbourne, Department of Mechanical and Manufacturing Engineering, 2007, pp. 63–79.

[2] C. F. Lee and C. Manzie, "Rapid parameter identification for an electromechanical brake", in *2013 Australian Control Conference*, IEEE, 2013, pp. 391–396.

[3] R. Huang, J. Fogelquist, and X. Lin, "Reinforcement learning of optimal input excitation for parameter estimation with application to li-ion battery", *IEEE Transactions on Industrial Informatics*, vol. 19, no. 11, pp. 11 160–11 170, 2023.

[4] A. Mesbah and S. Streif, "A probabilistic approach to robust optimal experiment design with chance constraints", *IFAC-PapersOnLine*, vol. 48, no. 8, pp. 100–105, 2015.

[5] K. Reif, *Automotive Mechatronics*. Springer, 2014, p. 13.

[6] H.-P. Schoner and P. Hille, "Automotive power electronics. new challenges for power electronics", in *2000 IEEE 31st Annual Power Electronics Specialists Conference. Conference Proceedings (Cat. No. 00CH37018)*, IEEE, vol. 1, 2000, pp. 6–11.

[7] L. Zhang, Z. Zhang, Z. Wang, J. Deng, and D. G. Dorrell, "Chassis coordinated control for full x-by-wire vehicles-a review", *Chinese Journal of Mechanical Engineering*, vol. 34, pp. 1–25, 2021.

[8] D. Li, C. Tan, W. Ge, J. Cui, C. Gu, and X. Chi, "Review of brake-by-wire system and control technology", in *Actuators*, MDPI, vol. 11, 2022, p. 80.

[9] C.-F. Zong, G. Li, H.-Y. Zheng, L. He, and Z.-X. Zhang, "Study progress and outlook of chassis control technology for x-by-wire automobile", *Zhongguo Gonglu Xuebao(China Journal of Highway and Transport)*, vol. 26, no. 2, pp. 160–176, 2013.

[10] C. Li, G. Zhuo, C. Tang, *et al.*, "A review of electro-mechanical brake (emb) system: Structure, control and application", *Sustainability*, vol. 15, no. 5, p. 4514, 2023.

[11] M. Zhang and J. Song, "A review of electromechanical brake (emb) system", *Mechanical Science and Technology*, vol. 24, no. 2, pp. 208–211, 2005.

[12] Y. Zhao, "Research and development of electro-mechanical brake system actuator", Ph.D. dissertation, Tsinghua University Beijing, China, 2010.

[13] S. Schrade, X. Nowak, A. Verhagen, and D. Schramm, "Short review of emb systems related to safety concepts", in *Actuators*, MDPI, vol. 11, 2022, p. 214.

[14] C. Line, C. Manzie, and M. C. Good, "Electromechanical brake modeling and control: From pi to mpc", *IEEE Transactions on Control Systems Technology*, vol. 16, no. 3, pp. 446–457, 2008. DOI: `10.1109/TCST.2007.908200`.

[15] L. Chen, P. Wu, K. Chitta, B. Jaeger, A. Geiger, and H. Li, "End-to-end autonomous driving: Challenges and frontiers", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

[16] S. Kuutti, R. Bowden, Y. Jin, P. Barber, and S. Fallah, “A survey of deep learning applications to autonomous vehicle control”, *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 2, pp. 712–733, 2020.

[17] B. Desai and K. Patil, “Secure and scalable multi-modal vehicle systems: A cloud-based framework for real-time llm-driven interactions”, *Innovative Computer Sciences Journal*, vol. 9, no. 1, pp. 1–11, 2023.

[18] Y.-m. You, “Multi-objective optimal design of permanent magnet synchronous motor for electric vehicle based on deep learning”, *Applied Sciences*, vol. 10, no. 2, p. 482, 2020.

[19] R. R. Ardeshiri, B. Balagopal, A. Alsabbagh, C. Ma, and M.-Y. Chow, “Machine learning approaches in battery management systems: State of the art: Remaining useful life and fault detection”, in *2020 2nd IEEE International conference on industrial electronics for sustainable energy systems (IESES)*, IEEE, vol. 1, 2020, pp. 61–66.

[20] DeepL, *Deepl translator*, Accessed: 2024-11-18, 2024. [Online]. Available: `https://www.deepl.com/translator`.

[21] OpenAI, *Chatgpt (gpt-4)*, Accessed: 2024-11-18, 2024. [Online]. Available: `https://chat.openai.com/`.

[22] T. Grigoratos, M. Mathissen, R. Vedula, *et al.*, “Interlaboratory study on brake particle emissions—part i: Particulate matter mass emissions”, *Atmosphere*, vol. 14, no. 3, p. 498, 2023.

[23] D. Lovrec and M. Kastrevc, “Modelling and simulating a controlled press-brake supply system”, *International Journal of Simulation Modelling*, vol. 10, no. 3, pp. 133–144, 2011.

[24] J. Kim, C. Jo, Y. Kwon, *et al.*, “Electro-mechanical brake for front wheel with back-up braking”, *SAE International Journal of Passenger Cars-Mechanical Systems*, vol. 7, no. 2014-01-2538, pp. 1369–1373, 2014.

[25] J. S. Cheon, J. Kim, and J. Jeon, “New brake by wire concept with mechanical backup”, *SAE International Journal of Passenger Cars-Mechanical Systems*, vol. 5, no. 2012-01-1800, pp. 1194–1198, 2012.

[26] C. Qi, “Kontaktereigniserkennung für ein elektromechanisches system”, Master’s Thesis, TU Berlin, 2024, p. 8.

[27] G. Park, S. Choi, and D. Hyun, “Clamping force estimation based on hysteresis modeling for electro-mechanical brakes”, *International Journal of Automotive Technology*, vol. 18, pp. 883–890, 2017.

[28] R. Stribeck, “Die wesentlischen eigenschaften der gleit-und rollenlager. zeitschrift des vereines deutscher ingenieure”, *vol*, vol. 46, pp. 1341–1348, 1902.

[29] D. Karnopp, “Computer Simulation of Stick-Slip Friction in Mechanical Dynamic Systems”, *Journal of Dynamic Systems, Measurement, and Control*, vol. 107, no. 1, pp. 100–103, Mar. 1985, ISSN: 0022-0434. DOI: `10.1115/1.3140698`. [Online]. Available: `https://doi.org/10.1115/1.3140698`.

[30] H. Olsson, K. J. Åström, C. C. De Wit, M. Gäfvert, and P. Lischinsky, “Friction models and friction compensation”, *Eur. J. Control*, vol. 4, no. 3, pp. 176–195, 1998.

[31] C. C. De Wit, H. Olsson, K. J. Astrom, and P. Lischinsky, “A new model for control of systems with friction”, *IEEE Transactions on automatic control*, vol. 40, no. 3, pp. 419–425, 1995.

[32] Y. Bard, *Nonlinear parameter estimation*. Academic press New York, 1974, vol. 1209.

[33] R. Mehra, “Optimal input signals for parameter estimation in dynamic systems–survey and new results”, *IEEE transactions on automatic control*, vol. 19, no. 6, pp. 753–768, 1974.

[34] S. Arimoto and H. Kimura, “Optimum input test signals for system identification—an information-theoretical approach”, *International Journal of Systems Science*, vol. 1, no. 3, pp. 279–290, 1971.

[35] A. Paszke, S. Gross, S. Chintala, *et al.*, “Automatic differentiation in pytorch”, 2017.

[36] D. A. Spielman, *Spectral and algebraic graph theory*, `http://cs-www.cs.yale.edu/homes/spielman/sagt/sagt.pdf`, Unpublished draft, 2019.

[37] R. Bellman, “A markovian decision process”, *Journal of mathematics and mechanics*, pp. 679–684, 1957.

[38] R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour, “Policy gradient methods for reinforcement learning with function approximation”, *Advances in neural information processing systems*, vol. 12, 1999.

[39] V. Mnih, “Playing atari with deep reinforcement learning”, *arXiv preprint arXiv:1312.5602*, 2013.

[40] T. Lillicrap, “Continuous control with deep reinforcement learning”, *arXiv preprint arXiv:1509.02971*, 2015.

[41] S. Fujimoto, H. Hoof, and D. Meger, “Addressing function approximation error in actor-critic methods”, in *International conference on machine learning*, PMLR, 2018, pp. 1587–1596.

[42] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, “Proximal policy optimization algorithms”, *arXiv preprint arXiv:1707.06347*, 2017.

[43] J. Schulman, “Trust region policy optimization”, *arXiv preprint arXiv:1502.05477*, 2015.

[44] S. Huang, R. F. J. Dossa, A. Raffin, A. Kanervisto, and W. Wang, “The 37 implementation details of proximal policy optimization”, in *ICLR Blog Track*, https://iclr-blog-track.github.io/2022/03/25/ppo-implementation-details/, 2022. [Online]. Available: `https://iclr-blog-track.github.io/2022/03/25/ppo-implementation-details/`.

[45] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra, “Planning and acting in partially observable stochastic domains”, *Artificial intelligence*, vol. 101, no. 1-2, pp. 99–134, 1998.

[46] M. Hauskrecht, “Value-function approximations for partially observable markov decision processes”, *Journal of artificial intelligence research*, vol. 13, pp. 33–94, 2000.

[47] J. Pineau, G. Gordon, S. Thrun, *et al.*, “Point-based value iteration: An anytime algorithm for pomdps”, in *Ijcai*, vol. 3, 2003, pp. 1025–1032.

[48] D. A. McAllester and S. Singh, “Approximate planning for factored pomdps using belief state simplification”, *arXiv preprint arXiv:1301.6719*, 2013.

[49] G. Shani, R. Brafman, and S. Shimony, “Adaptation for changing stochastic environments through online pomdp policy learning”, in *Proc. Eur. Conf. on Machine Learning*, 2005, pp. 61–70.

[50] S. Ross, J. Pineau, S. Paquet, and B. Chaib-Draa, “Online planning algorithms for pomdps”, *Journal of Artificial Intelligence Research*, vol. 32, pp. 663–704, 2008.

[51] L. Pinto, M. Andrychowicz, P. Welinder, W. Zaremba, and P. Abbeel, “Asymmetric actor critic for image-based robot learning”, *arXiv preprint arXiv:1710.06542*, 2017.

[52] W. Yue, Y. Zhou, X. Zhang, Y. Hua, Z. Wang, and G. Kou, “Aacc: Asymmetric actor-critic in contextual reinforcement learning”, *arXiv preprint arXiv:2208.02376*, 2022.

[53] A. Baisero and C. Amato, “Unbiased asymmetric reinforcement learning under partial observability”, *arXiv preprint arXiv:2105.11674*, 2021.

[54] S. Huang, R. F. J. Dossa, C. Ye, *et al.*, “Cleanrl: High-quality single-file implementations of deep reinforcement learning algorithms”, *Journal of Machine Learning Research*, vol. 23, no. 274, pp. 1–18, 2022. [Online]. Available: `http://jmlr.org/papers/v23/21-1342.html`.

[55] M. Towers, A. Kwiatkowski, J. Terry, *et al.*, “Gymnasium: A standard interface for reinforcement learning environments”, *arXiv preprint arXiv:2407.17032*, 2024.

[56] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel, “High-dimensional continuous control using generalized advantage estimation”, *arXiv preprint arXiv:1506.02438*, 2015.

[57] Z. Tan, J. Wei, and N. Dai, “Real-time dynamic trajectory planning for intelligent vehicles based on quintic polynomial”, in *2022 IEEE 21st International Conference on Ubiquitous Computing and Communications (IUCC/CIT/DSCI/SmartCNS)*, IEEE, 2022, pp. 51–56.

[58] Y. Li and B. Mo, “The trajectory planning of spacecraft based on optimal quintic polynomial”, in *Proceedings of 2013 2nd International Conference on Measurement, Information and Control*, IEEE, vol. 2, 2013, pp. 865–868.

[59] S. Fang, X. Ma, Y. Zhao, Q. Zhang, and Y. Li, “Trajectory planning for seven-dof robotic arm based on quintic polynormial”, in *2019 11th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC)*, IEEE, vol. 2, 2019, pp. 198–201.