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.

Stuttgart, 2. December 2024

Shengya Guo

Abstract

With the development of brake-by-wire technology, electromechanical brakes (EMB) are becoming increasingly popular. To accurately control the clamping force, precise estimation of the EMB system parameters is crucial. While optimal experimental design (OED) ensures that the outputs during parameter identification experiments contain rich parameter information by optimizing the system input excitation, it struggles to handle parameter uncertainty effectively. This thesis presents a reinforcement learning (RL) approach to address this challenge and achieve optimal input excitation for EMB actuators.

To reduce the complexity of the research, a simplified EMB model is first developed. The D-optimal and E-optimal criteria, based on the Fisher Information Matrix (FIM), are reformulated and integrated to the reward function design of reinforcement learning. An asymmetric Proximal Policy Optimization (PPO) framework is introduced to optimize the input during experiments without direct access to parameter values, by allowing the critic network to acquire additional information during training in a simulated environment. A distribution-based parameter initialization method is employed to sample parameters during reinforcement learning training, enhancing the robustness of parameter estimation experiments against uncertainties in system parameters. Additionally, EMB system constraints and real-world experimental requirements are considered in the reinforcement learning framework.

The proposed method was tested on a simulated EMB system to evaluate its performance. For single parameter input excitation optimization, the method demonstrated an effective excitation policy with robustness to unknown parameters and strong generalization ability. This result was further validated in multi-parameter experiments, with no constraint violations under significant changes in unknown system parameters. The method proposed in this thesis provides a new perspective for input excitation in EMB systems and has the potential to be extended to other systems.

Kurzfassung

Mit der Entwicklung der Brake-by-Wire-Technologie gewinnen elektromechanische Bremsen (EMB) zunehmend an Bedeutung. Für eine präzise Regelung der Klemmkraft ist eine genaue Schätzung der EMB-Systemparameter von entscheidender Bedeutung. Während das Optimal Experimental Design (OED) sicherstellt, dass die Ausgaben während der Parameteridentifikationsexperimente reich an Parameterinformationen sind, indem die Systemeingangsanregungen optimiert werden, hat es Schwierigkeiten, die Parameterunsicherheit effektiv zu bewältigen. In dieser Arbeit wird ein Reinforcement Learning (RL) Ansatz vorgestellt, um diese Herausforderung zu meistern und eine optimale Eingangsanregung für EMB-Aktoren zu erzielen.

Um die Komplexität der Untersuchung zu reduzieren, wird zunächst ein vereinfachtes EMB-Modell entwickelt. Die D-optimalen und E-optimalen Kriterien des OED werden umformuliert und in das Belohnungsfunktionsdesign des Reinforcement Learning integriert. Ein asymmetrisches Proximal Policy Optimization (PPO) Framework wird eingeführt, um die Eingabe während der Experimente ohne direkten Zugriff auf die Parameterwerte zu optimieren, indem das Kritiker-Netzwerk zusätzliche Informationen während des Trainings in einer simulierten Umgebung erwirbt. Eine verteilungsbasierte Parameterinitialisierungsmethode wird verwendet, um Parameter während des Reinforcement Learning

Trainings zu sampeln, wodurch die Robustheit der Parameteridentifikationsexperimente gegenüber Unsicherheiten in den Systemparametern erhöht wird. Zusätzlich werden Systembeschränkungen des EMB sowie Anforderungen aus realen Experimenten im Reinforcement Learning Framework berücksichtigt.

Die vorgeschlagene Methode wurde an einem simulierten EMB-System getestet, um ihre Leistung zu evaluieren. Für die Optimierung der Eingangsanregung eines einzelnen Parameters zeigte die Methode eine effektive Anregungsstrategie mit Robustheit gegenüber unbekannten Parametern und einer starken Generalisierungsfähigkeit. Dieses Ergebnis wurde in Mehrparameter-Experimenten bestätigt, ohne dass Einschränkungen verletzt wurden, selbst wenn sich die unbekannten Systemparameter signifikant änderten. Die in dieser Arbeit vorgeschlagene Methode bietet eine neue Perspektive für die Eingangsanregung in EMB-Systemen und hat das Potenzial, auf andere Systeme übertragen zu werden.

Contents

1 Introduction

Nowadays, electrification of vehicles and autonomous driving have become the two most popular trends in the vehicle manufacturing industry. Encouraged by policies in different countries, the battery electric vehicle (BEV) market has exploded in recent years. More and more brands have started to turn their production towards electrification. Traditionally, motor vehicles used a 12-volt platform and an Electronic Control Unit (ECU) to control the vehicle system, with the primary goal of ensuring vehicle functionality under all circumstances. To enable new functions, such as Adaptive Cruise Control (ACC), the amount of networking within the domains of traditional vehicle EE system architectures has increased [5]. This has led to increased electrical power requirements in vehicles to accomplish new functions [6]. In this respect, the electric vehicle platform offers several advantages, such as the ability to carry high-performance computers and control all chassis components with a single centralized system, making the vehicle more suitable for autonomous driving.

With the growing interest in autonomous driving and the support of electric vehicle platforms, a new technology called X-by-wire has show up. X-by-wire technology replaces traditional mechanical connections between the operator and chassis components with electronic signals. By doing so, advanced X-by-wire chassis systems can reduce system reaction time and improve the performance and safety of vehicles [7].

Brake-by-wire (BBW) is part of the X-by-wire chassis. Unlike traditional hydraulic brake systems, where the brake pedal is connected to a hydraulic cylinder, in a BBW system the brake pedal is not directly connected to the brake system. Instead, it generates a control signal and uses a controller to manipulate the brake actuator. BBW systems can be divided into two main types based on the source and method of brake force control: electrohydraulic brake (EHB) systems and electromechanical brake (EMB) systems. [8].

The EHB system uses an electric pump to drive the hydraulic system and generate clamping force for the brakes. Due to the advantages of good compatibility and ease of implementation, EHB has become a mature solution in the current market [9]. However, since hydraulic fluid systems are a main component of EHB systems, the drawbacks of hydraulic systems, such as slow braking response, the possibility of leaks, and the potential for pollution, still exist [10]. Moreover, eliminating the high cost associated with hydraulic systems has always been a goal for manufacturers.

In EMB systems, actuator-generated torque is transmitted directly to the brake's friction components via mechanisms like gear reduction and ball screws [10]. Because the hydraulic system is eliminated, the disadvantages of EHB systems do not apply to EMB systems. Moreover, from a physics perspective, EMB systems can achieve faster braking responses, which are urgently needed for autonomous driving. In many dangerous scenarios, autonomous vehicles need to make decisions in milliseconds. With the help of EMB systems, autonomous driving can achieve higher driving performance and shorter braking distances than an experienced human driver. EMB systems also have several other advantages, including

reduced system volume and weight, integration with a parking brake, and compatibility with active safety control systems [11], [12].

Despite these mentioned advantages, EMB systems have not yet been implemented in mass production. Functional safety concepts are a major reason for this [13]. EMB actuators for disc brakes need to provide a maximum caliper clamping force of 40 kN to ensure sufficient braking force [14]. EMB systems must function under all conditions, including extreme temperatures and strong vibrations, where standard motors may not perform reliably.

To guarantee rapid response under all circumstances, EMB systems require very precise control and fast response. Hence, accurate modeling of EMB is a major research challenge. Estimating model parameters is crucial for ensuring the performance and accuracy of EMB models. For this reason, several studies on parameter estimation for EMB systems have been conducted, including traditional signal parameter experiments and rapid experimental procedures [1], [2].

With the development of artificial intelligence (AI) and the increasing computing power of edge devices, more and more AI approaches are being applied in the automotive industry. Not only are popular topics such as end-to-end autonomous driving and large language models (LLMs) included in AI applications, but AI is also widely used to address industry problems such as complex and non-linear control for autonomous vehicles, optimal motor design for electric vehicles, and optimization of battery management systems (BMS) [15]–[19]. This suggests that AI approaches could also be helpful in estimating EMB parameters, which will be explored in detail in this thesis.

In this thesis, LLMs and AI-based tools are used to refine grammar and fine-tune writing [20], [21].

1.1 Motivation

Since the parameters of the EMB model have to be identified through experiments, the proper design of the experiment is crucial. Optimal experimental design (OED) is an important approach to ensure that the experiment is conducted effectively and that the system output contains high information content. However, conventional OED approaches are often limited by the challenge of parameter uncertainty, as the optimization relies on prior estimates of the parameters to be identified. If the prior values differ significantly from the actual parameter values, the information content of the output is reduced, leading to less accurate parameter estimation results.

This disadvantage is even more pronounced in EMB systems. Unlike typical systems, EMB is a dynamic system and its characteristics change over time and with temperature variations. Therefore, the parameters identified using conventional methods cannot be guaranteed to remain accurate in the long term.

In contrast to traditional time-consuming OED methods, this thesis attempts to address these issues by applying artificial intelligence, specifically reinforcement learning (RL), to determine the optimal input excitation for parameter estimation. The primary objective of this thesis is to investigate two questions: first, how reinforcement learning can be used to make experimental performance robust to parameter changes; second, how the RL framework can be constructed so that it remains effective without requiring knowledge of the system parameters after training, while ensuring that the constraints of the EMB system are not violated.

Another motivation for this thesis is that its approach is not limited to EMB systems alone. More specifically, the thesis aims to develop a general approach for optimizing input excitations for parameter estimation. The EMB system serves as a platform for applying and testing the method, but the proposed approach can be applied to other systems with simple modifications to optimize input excitations for parameter estimation.

1.2 Structure of the Thesis

Chapter 2 provides an overview of the current state of the art in parameter estimation and optimal experimental design (OED) methods. The strengths and weaknesses of different OED approaches are compared, and their applications in the field of EMB systems, as well as their use in reinforcement learning to optimize input excitation, are highlighted.

In Chapter 3, the modeling of the electromechanical braking system (EMB) is described and a simplified model of the EMB used in this thesis is introduced.

Chapter 4 discusses the methods for applying reinforcement learning to optimize experimental design in EMB systems. Detailed explanations of the reinforcement learning framework, Fisher information matrix calculation procedures, and environment setup are included.

Chapter 5 describes the application of the reinforcement learning framework to optimize input excitation in the single-parameter case. The performance of two different parameter initialization methods is compared, and results from the asymmetric reinforcement learning framework are presented.

Chapter 6 extends the application to the multi-parameter domain, where different approaches for the multi-parameter case are described and the results obtained are presented.

Chapter 7 summarizes the results of the thesis and gives an outlook on future work.

2 State of the Art

Parameter estimation is the foundation of system modeling because the accuracy of estimation of unknown system parameters directly affects the performance of system modeling and affects system response and control performance. Therefore, accurate system parameters are essential for the development of reliable system models. Optimal Experimental Design (OED) determines the information content of the data generated during system parameter estimation experiments, specifically in EMB systems, it means optimizing the system input excitations to make the parameter identification results more accurate. Due to its importance, a significant amount of research has been conducted on parameter estimation and OED. Therefore, an introduction to the current research is necessary before the specific implementation of EMB actuators.

2.1 Parameter Estimation

Two main methods are used for parameter estimation: output error method (OEM) and prediction error method (PEM). These two methods identify model parameters by minimizing the error between the actual output and the predicted output of the model. While OEM assumes that there is no error in the input signal and error only exists in the system output, PEM takes both input and output errors into account. It has been shown that despite the simplicity and computational ease of OEM, it is generally less effective than PEM [22], [23].

OEM defines the objective function based on the least squares criterion and adjusts the system parameters by minimizing the sum of squares of the output errors. Iterative and recursive methods, such as Recursive Least Squares (RLS), are commonly used in this process [24], [25]. Specifically, in practical applications of OEM, several improvement methods have been proposed to meet the actual requirements. In [26], a gradient-based recursive algorithm for parameter estimation of dual-rate stochastic systems is presented. To improve computational efficiency, a filtering and auxiliary model based recursive least squares algorithm is also proposed [27].

In the field of mechatronic systems and braking systems, several studies have used OEM for parameter estimation. In the field of robotics, a closed-loop OEM is used for robot dynamics parameter estimation [28]. In [29], a polynomial regression method using the least squares criterion is used to estimate the temperature parameters of a braking system. Due to the inefficiency of the recursive least squares method, a low-complexity recursive estimator for tracking resolver parameters has been developed for EMB systems [30].

Despite the parameter estimation approaches mentioned above, all methods are based on parameter estimation experiments. Taking EMB as an example, specific experiments are conducted to estimate each system parameter [1], such as using constant speed tests to identify Coulomb friction. However,

traditional experimental methods can be time consuming. For EMB systems, data collection can take over nine hours [2]. Therefore, designing the experiments correctly to make estimating the parameters more efficient is crucial.

2.2 Optimal Experimental Design

The objective of optimal experimental design is to maximize the information content of the data generated during the parameter identification session by correctly designing the system input excitations. The information content of the data directly affects the accuracy of the parameter identification, which is crucial for systems with high reliability requirements [31]. Although OED cannot achieve parameter identification by itself, it is an essential part of effective system parameter identification and has attracted significant research interest in recent years.

The Fisher information matrix and Kullback-Leibler divergence are commonly used to measure OED performance [3]. Traditional OED is based on FIM, which transforms OED into an optimization problem by constructing an objective function based on A, E, and D-optimality criteria related to FIM. In [32], multivariable excitations based on weighted A-optimality criterion are used for the identification of MIMO mechatronic systems. A comparison of these three criteria with geometrical calibration on an industrial robot suggests that D-optimality yields a smaller error [33]. OED is also used for the parameter estimation in EMB system. In [2], a genetic algorithm is used to solve an optimization problem for the OED in the EMB with noise, based on the D-optimality criterion.

However, the calculation of the FIM in the OED method is based on system parameters. The main challenge in performing OED is the uncertainty of the system. Due to the lack of knowledge of the true parameter values prior to parameter identification experiments, OED optimization can only be performed by setting an a priori value of the system parameters as a basis for OED. Designing experiments with unknown true parameters reduces the performance of the OED and the information content of the data collected during the experiments [34].

One approach to dealing with system parameter uncertainty is iterative OED: designing dynamic experiments and conducting them repeatedly until the parameter error tolerance is satisfied [34]. Using this approach, an online OED framework for nonlinear dynamic biosystems is developed [35]. The disadvantage of this approach is that, in the worst case, the experiments are repeated too many times and thus not time efficient.

Another problem faced by OED is the risk of violating state and input constraints when system parameters are unknown [4]. In response to this problem, research about robust OED methods have been studied. In robust OED, uncertainties are typically assumed to be deterministic and bounded. The disadvantage of the robust OED is that it can be conservative if the worst case scenario has only a small chance of occurring [4]. A robust experimental design methodology is proposed in [36], which optimizes the expected FIM under parameter uncertainty using stochastic approximation techniques. A robust optimization method is proposed for the design of experiments in nonlinear dynamic processes, addressing the complexity of parameter-dependent objectives and parameter value uncertainty [37].

Bayesian OED has also been proposed to deal with parameter uncertainty. In this approach, the uncertainty in the parameters of the system model is represented as probability distributions, the parameter distributions are updated by Bayesian formulations in combination with the experimental data to obtain more accurate posterior distributions, and then the optimal experimental design is

Design Method	Advantages	Disadvantages
Traditional OED	Simple implementation	Ineffective with parameter uncertainty
Iterative OED	Capable of parameter uncertainty	Requires multiple iterations and time-inefficient
Robust OED	Avoid violating state and input constraint	Possibilities of conservative action
Bayesian OED	Reflect the uncertainty reduction	Limited to linear systems and computationally intensive

Table 2.1: Comparison of different OED approaches

performed by maximizing the information gain, such as the KL divergence between posterior and prior distributions [38].

A Bayesian OED framework integrated with Bernoulli priors is used to optimize sensor placement for accurate model validation and structural damage identification [39], [40]. The disadvantage of Bayesian OED is that it is only applicable to linear systems, and its application to nonlinear systems requires approximation [41]. Laplace approximation of the posterior distributions is a commonly used method [42]. However, the Bayesian approach has the drawback of being computationally intensive and thus is not suitable for parameter estimation tasks with real-time requirements [43]. A comparison of different OED approaches is listed in Table 2.1.

In addition, OED is often used in conjunction with model predictive control (MPC). On the one hand, it can incorporate a measure of the quality of parameter estimates into the MPC formulation to reduce the effect of system uncertainty on control accuracy [44], [45]. In [46], an adaptive MPC framework integrated with OED has been proposed for safe fast charging of batteries with real-time parameter identifiability. On the other hand, MPC can also be used to support OED. In [47], MPC is used to implement the control strategy of OED in a single reaction controller, which transfers the system constraint problem from OED to MPC and reduces the computational difficulty of the OED problem for multi-bioreactors.

In recent years, reinforcement learning has been studied in system input excitation optimization and OED. In the lithium battery model, sensitivity was used as the reward function and Q-learning was used for the input excitation of the lithium battery to improve the accuracy of parameter identification [3]. The disadvantage of this method is that although two system parameters are studied, only one parameter can be optimized per experiment. Additionally, due to the use of Q-learning, the input is limited to a discrete space.

Research on deep reinforcement learning for OED in the biological domain has also been carried out. In [43], the optimal design of experiments for biological environments with unknown system parameters was achieved based on the experimental history data using a technique that combines Twin Delayed Deep Deterministic Policy Gradient (TD3) and recurrent layer, which removes the a priori values of system parameters from the reinforcement learning observation space. The advantage of this study is that multiple system parameters are optimized simultaneously and the action space of the agent is continuous. However, its disadvantages include the assumption that the experiment consists of only

ten time intervals of 2 hours each, which limits the agent's ability to adjust system inputs at a higher frequency. The result is not suitable for systems that require fast responses.

In summary, research has already demonstrated the use of OED in EMB systems. Different OED methods have some limitations. Reinforcement learning has been less applied to OED and has only been explored in the battery and biological fields. In this thesis, the performance of deep reinforcement learning is explored to optimize the input excitation of EMB systems, to improve the robustness of experimental results to the uncertainty of system parameters and fulfil the system constraint.

3 Modeling of the Electromechanical Brake System

In order to conduct research into optimal experimental design, it is essential to create a model of the EMB system. Currently, significant research has been conducted on electromechanical brakes, with a particular focus on disc EMB systems. Compared to disc brakes, drum brakes have not been widely used on vehicles in recent years. There are several reasons for this, including lower heat dissipation performance and the perception that drum brakes are of lower quality and less expensive. However, drum brakes still offer advantages in light of increasingly stringent regulatory requirements, particularly in terms of reduced particulate emissions due to the sealed structure [48]. Consequently, manufacturers such as Bosch have resumed research on electromechanical drum brakes. In this thesis, research is conducted based on an electromechanical drum brake system from Robert Bosch GmbH.

In this chapter, the structure of the EMB system to be modelled is introduced first. Then, different approaches to modelling the EMB system are presented. Next, the key parameters for modelling the EMB from different perspectives are described. Finally, a simplified version of the EMB model is provided for further use.

3.1 Structure of the Electromechanical Drum Brake

As shown in Figure 3.1, the electromechanical drum brake being analysed has the following components: the brake shoes, the brake lining, the return springs and the brake drum. When the ECU receives a braking signal, it activates the electric motor which drives the actuator and pushes the brake shoes and linings against the drum. When the brake is released, the motor reverses and a spring returns the brake shoes and pads to their original position [49], [50].

Due to the structure of the electromechanical drum brake, several forces need to be taken into consideration during the modelling. First of all, the purpose of the brake is to reduce the wheel speed. When the actuator forces the brake shoes to move against the brake drum, the brake lining makes contact with the drum, and clamping force is generated, which is then converted into braking torque. Secondly, the force from the return springs is also a significant component of the model. This force helps to return the brake shoes to their original position when the brake is released. Another major force that needs to be considered is the force from the actuator. This force is responsible for moving the brake shoes to generate the clamping force. The friction torque is also a type of torque that cannot be neglected. It is present throughout the system, from inside the motor to the actuator mechanism. The following sections discuss each of these forces and torques in detail.

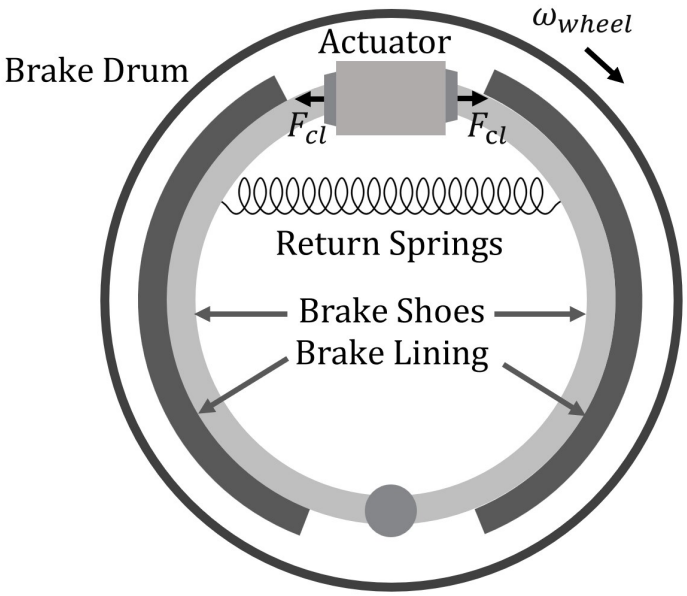

Figure 3.1: Mechanical structure of the electromechanical drum brake to be analyzed [51]

3.2 Lumped Parameter EMB Model

In this section, all the torques mentioned above are carefully analysed. As mentioned above, torques occur throughout the EMB system, which makes modelling challenging. According to research, the lumped parameter EMB model is commonly used [1]. In this thesis, three main torques are considered in the lumped parameter EMB model: the motor torque T_m, the load torque T_l and the friction torque T_f. Considering the dynamic behavior of the motor, the EMB is 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 the friction torque T_f act in opposition, resisting this acceleration. Assuming a linear relationship, the electromagnetic motor torque T_m can be considered 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 total gear ratio γ

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

The total gear ratio γ represents the characteristics of the entire mechanical system, converting the force from the brake shoe side into torque at the motor side. The friction torque T_f can be represented as a function of the motor angle θ and the motor velocity $\dot{\theta}$. The detailed calculations for the clamping force F_{cl} and the load torque T_l are presented in the following subsections.

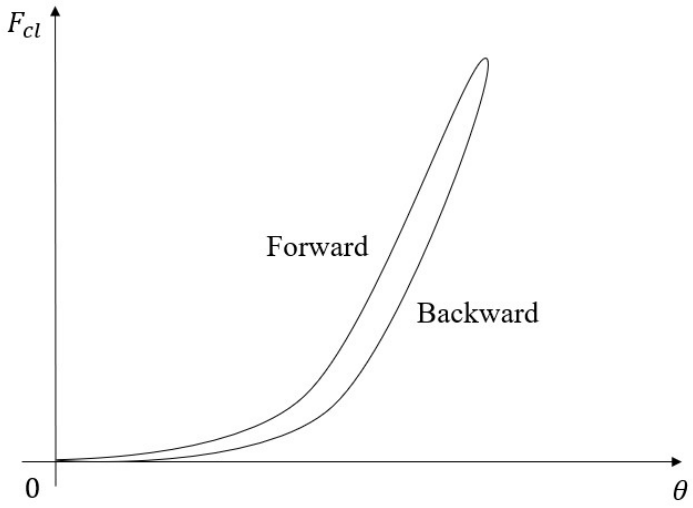

Figure 3.2: Qualitative hysteresis characteristics between clamping force and motor angle

3.2.1 Stiffness Model

The EMB is a complex nonlinear system and its nonlinear characteristics 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 using force sensors positioned on the EMB. As shown in Figure 3.2, the stiffness curve is nonlinear. It is assumed that 0 degrees represents the point of contact between the pad and the drum. The brake force stiffness curve can be divided into two intervals. In the initial range, the brake force increases slowly. However, from the later range, the brake force increases almost exponentially with the angle. The stiffness model of the EMB exhibits hysteresis curve characteristics as described in the referenced literature [52]. Compared to the process where the pads leave the brake drum due to the action of the return spring, when the pads are actuated by the actuator and move towards the brake drum, the curve may reflect a higher stiffness. This could be explained by the actuation phase lag due to internal friction and the existence of gear backlash [52], [53]. In addition, the hysteresis characteristics are also dependent on factors such as speed and temperature. As 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

Friction in physical systems has been studied intensively for a long time. The classical model of friction is a static model, it includes Coulomb friction F_c and viscous friction F_v. By applying this classical model to the EMB system, the frictional torque is equal to the sum of these two frictional torques

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

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

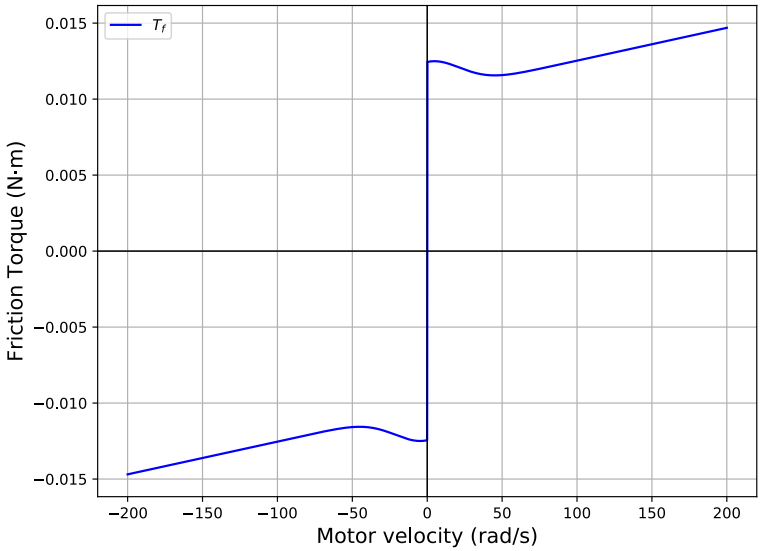

Figure 3.3: Illustration of the Stribeck friction model under $T_c = 1.037 \times 10^{-2}$N·m, $T_s = 1.2444 \times 10^{-2}$N·m, $f_v = 2.16 \times 10^{-5}$ Nm·s/rad, $v_s = 30$rad/s, $\delta_\sigma = 2$

The drawback in equation (3.4) is, the stiction which is higher than the Coulomb friction is not included. Stribeck observed a continuous decrease in the dependence of the friction on the velocity [54]. A commonly used model for the non-linearity with Stribeck behavior considered in EMB is expressed as

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

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

In EMB modeling, an additional friction component 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 above model is detecting when the velocity is zero [55]. Karnopp discovered that this problem can be solved by introducing a small zero speed interval θ into the model and preventing the motor from repeatedly alternating between positive and negative directions of motion [56]. According to the above model, the friction force can be either arbitrary or defined, depending on whether the speed is below the threshold of the low-speed interval. With this theory, the friction of the EMB system could be defined 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 is the external torque and T_f is the load independent static friction torque. If the motor speed is in the zero speed interval and the external torque is less than the static friction torque, the friction torque will be equal to the external torque as the system could be considered as not moving.

Olsson et al.[55] said that many friction phenomena do not show up in constant velocity experiments. Therefore, in addition to static friction modeling systems, dynamic friction modeling has also been studied. Due to the high accuracy of the EMB, the dynamic friction model is more complex and requires detailed modeling of the system to analyse the dynamic properties of the system. Consequently, the dynamic friction model is also considered in the modeling.

There is a dynamic friction model called the LuGre model [57]. 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, the average deflection depends on the speed, becoming smaller at higher speeds. This reduction in friction at higher speeds represents the Stribeck effect. The model also includes speed-dependent behavior, which captures dynamic effects as frictional lag.

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

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

As the equation shows, the LuGre model in static form will be the same as shown in equation (3.5), with the static friction model and the Stribeck effect taken into account. This shows the unification of different ways of modeling friction.

3.3 Simplified Model of the Electromechanical Brake System

The previous section provides a comprehensive overview of the various factors that need to 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. However, in reality, the importance of different levels of accuracy when modeling different parts of hydraulic brake systems has been well described by several studies, with simplified modeling often being applied [58]. Therefore, the most detailed model is not always necessary. In most cases, the model needs to be simplified to facilitate calculation.

This thesis investigates the potential for optimal input excitation using reinforcement learning methods for parameter estimation. In theory, each parameter has an optimal excitation, and overly complex EMB models will contain a greater number of parameters to learn. This complexity will further complicate the problem that reinforcement learning is intended to solve. As this thesis focuses primarily on researching the methodology and subsequently testing it on simulated EMB systems, appropriate simplifications of the models have been made for each of the aforementioned components.

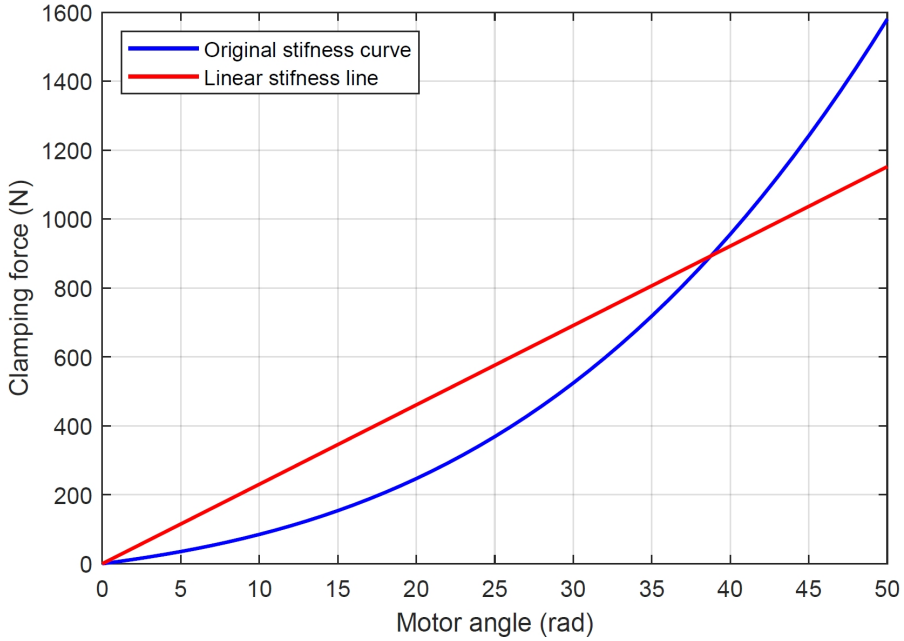

Figure 3.4: Comparison of original and linearised stiffness curves for clamping force

3.3.1 Simplification of the Stiffness Model

For the stiffness model, a polynomial is typically fitted to the experimentally collected data relating motor angle to clamping force, with the fitted result often being a cubic curve that takes the zero angle as the contact point

$$F_{cl} = k_1\theta^3 + k_2\theta^2 + k_3\theta, \quad \text{for } \theta \geq 0 \tag{3.9}$$

Figure 3.4 illustrates the blue 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.

Since the introduction of three stiffness parameters, k_1, k_2, k_3, would greatly increase the dimensional complexity of the reinforcement learning problem, parameter estimation for each of these components is required. To mitigate this problem, a linear interpolation method is used in the modeling process. The linear clamping force is represented by

$$F_{cl} = k_1\theta, \quad \text{for } \theta \geq 0 \tag{3.10}$$

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.

To reduce the dimension level of the stiffness parameter, data from a validated EMB model was imported into *Matlab*, and the *Curve Fitting Toolbox* was used to derive the linear stiffness line (red line) through linear interpolation. The 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.

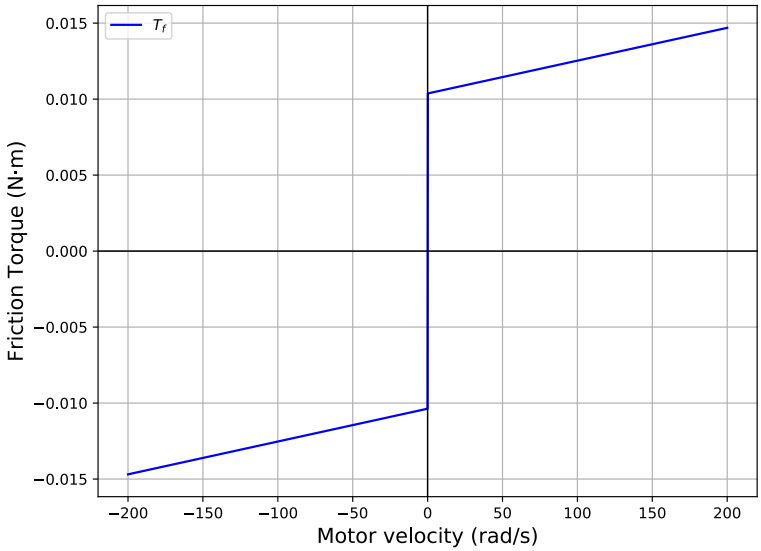

Figure 3.5: Illustration of the simplified friction model

3.3.2 Simplification of the Friction Model

With the discussion of different approaches to modeling friction in the previous text, it can be observed that complex friction models provide high accuracy but also add more parameter dimensions. Taking the Stribeck model as an example, the modeling session introduces more parameters such as the Stribeck velocity. If the load-dependent static friction torque is considered, as shown in equation (3.6), it will not only bring other parameters as T_{s0} and f_l, but also make the friction model relevant to the stiffness model, which will make the analysis of the whole EMB model even more complicated.

In order to limit the dimension of the parameters in the reinforcement learning process, the primary consideration when modeling friction is to ensure the simplicity of the model. Therefore, 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 simplifications of the early texts, a simplified lumped parameter nonlinear EMB model was finally introduced and shown 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 written 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 clamping force, T_f is the friction torque. In this simplified lumped parameter EMB model, the

stiffness characteristics of the clamping force is considered as

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

where k_1 is the stiffness parameter, which is not fully accurate but is easy to calculate. The friction model here only takes into account Coulomb and viscous friction, 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 respectively.

The system output is the observation of 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 the Application of Reinforcement Learning to Optimize Input Excitation in EMB Systems

As demonstrated in Chapter 2, the OED problem has been extensively studied using a variety of OED methods, with additional techniques such as MPC and RL also employed [3], [33]–[35], [39], [40], [43]–[45]. However, these approaches are not without their limitations. In particular, reinforcement learning methods have only been applied in limited contexts and still have significant shortcomings. This chapter has outlined the proposed methodology, introduced a novel reinforcement learning framework for optimal input excitation to facilitate system parameter estimation, and described the theoretical methods used in this framework from beginning to end.

4.1 Formulation the EMB model for the Optimization of Input Excitation

In order to avoid dangerous situations, an important consideration for OED is the system constraints. The input to the EMB model depends on the technical parameters of the motor. It is given that the input current of the motor is limited to a range of -6A to 6A. Secondly, the range of motion of the motor is limited by the physical limitations of the system, so the motor cannot rotate indefinitely. Considering 0 rad as the contact point, the motor's range of rotation is set between -10 rad and 100 rad. An angle less than 0 means that there is no clamping force. In addition, test data obtained from the EMB in the laboratory has shown that the maximum velocity of the motor is around 500 rad/s. Excessive motor velocity not only leads to increased friction and heat generation, but also reduces the control accuracy of the system. As a result, the motor velocity has been limited to a range of -500 rad/s to 500 rad/s.

Since sensors are used in the actual parameter estimation process, the sensor measurements are discrete sequences. Consequently, the continuous state space equations mentioned above must be discretized. There are many methods to discretize the system, including Euler's method, Runge-Kutta method and others [59]. In order to keep the model simple for the reinforcement method, Euler's method is used 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 set to be 0.001s in this model accordingly.

In addition, the sensor noise used to quantify the input and output of the system must be taken into account when constructing the model, as it will directly affect the accuracy of the parameter estimation. According to 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 relatively higher degree of error. The sensor noise is typically modeled as Gaussian noise following a Gaussian distribution $v_o \sim \mathcal{N}(0, R)$ with mean 0 and variance R. For the angular sensor, the variance of the Gaussian noise is $1 \times 10^{-6}\,\text{rad}^2$. For the state observer used to observe the motor velocity, the variance of the Gaussian noise is $1\,(\text{rad/s})^2$. The input signal is assumed to be completely free from noise, which leads to the assumption that $v_i = 0$. After discrediting the system and accounting for the input noise, the EMB model in subsection 3.3.3 becomes

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

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. $u(k)$ represent the input current. 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 results of the simulation model, parameters that have been included in the EMB system have been identified in the table 4.1.

In these parameters, the lumped inertia of the EMB system J and the total gear ratio γ could be considered as fixed and provided by the manufacturer, so they are not included in the parameter estimation. The motor constant k_m and the stiffness parameter k_1 will change with the use of EMB, so there are two parameters to be estimated. The friction parameters are also critical for the EMB system as they directly affect the braking performance as these parameters can be affected by factors such as temperature, wear and contamination which can change the response and reliability of the system. In the end, the estimated EMB parameters θ have 4 dimensions

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

4.2 Optimal Experimental Design and Fisher Information Matrix

This section discussed the formulation of optimal experimental design (OED) in the context of the EMB model, focusing on concepts such as the Fisher Information Matrix (FIM) and sensitivity analysis. In continuous space, the FIM takes the form [60]

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

with R as the variance-covariance matrix of the measurement noise. In the equation (4.4), the part $\frac{\partial \mathbf{h}}{\partial \boldsymbol{\theta}}$ is called sensitivity. It reflects the relationship between system outputs and unknown parameters. Sensitivity analysis determines how changes in the input trajectory affect the system output with respect

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

Table 4.1: Parameters for the Simplified EMB Model

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 that input, suggesting that the input trajectory contains a high level of information about the parameters. Therefore, sensitivity analysis is essential to evaluate the effectiveness of an input in parameter estimation.

In addition, FIM includes the effect of measurement noise. As mentioned earlier, measurement noise was considered to be Gaussian noise with a mean of zero. In multiple-input multiple-output (MIMO) systems, the greater the noise in the output measurements of a given dimension, the less accurate the measurements become. This results in a reduction in the information content of the outputs of that dimension generated in the parameter estimation process. The inverse matrix of the output noise variance matrix R^{-1} is included in the equation (4.4). This form enhances the amount of information contained in the output dimension with measurement variance less than 1 and reduces the amount of information contained in the output dimension with measurement variance greater than 1.

The Fisher Information Matrix 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 the measurement noise is uncorrelated, the inverse of the FIM

provides an approximation to the Cramér-Rao lower bound [60]. This bound represents the theoretical lower bound on the variance of unbiased estimators, which is closely related to the variance-covariance matrix of the estimated parameters. By maximizing some criteria of the FIM, the lower bound is minimized, thus giving the system the possibility to obtain a lower variance of unbiased estimators, which means an improvement of the precision of the parameter estimation. Therefore, when designing the input excitation trajectory, the input excitation criteria are often evaluated using scalar measures of the FIM M and the inverse matrix of the FIM M^{-1}, including [61]:

1) D-optimality: Minimises the determinant of the inverse of the FIM $|M^{-1}|$, ensuring that the volume of the confidence ellipsoid for parameter estimates is minimized. Also, the convex function $log|M^{-1}|$ implies the mutual information between system output and parameters [62].

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

3) E-optimality: Minimises the largest eigenvalue of the inverse of the FIM $\lambda_{max}(M^{-1})$, which minimizes 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 with two parameters that need to be identified, according to Equation (4.4), once the information associated with each parameter is available in the output, FIM M can be expressed as a positive definite symmetric matrix, so the inverse matrix of FIM takes 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 of these three optimality with these parameters in two dimensions. The blue shaded area in the figure represents the area of the confidence ellipse. Since the two parameters form a two-dimensional planar space, the D-optimality is reflected in the minimization of the area of the ellipse. The half-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. A-optimality aims at minimizing the trace, i.e. $m_{11} + m_{22}$, of the FIM inverse matrix. This is equivalent to an overall reduction in uncertainty in all directions.

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

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 FIM into an optimization problem. Assume the existence of a time-discrete EMB

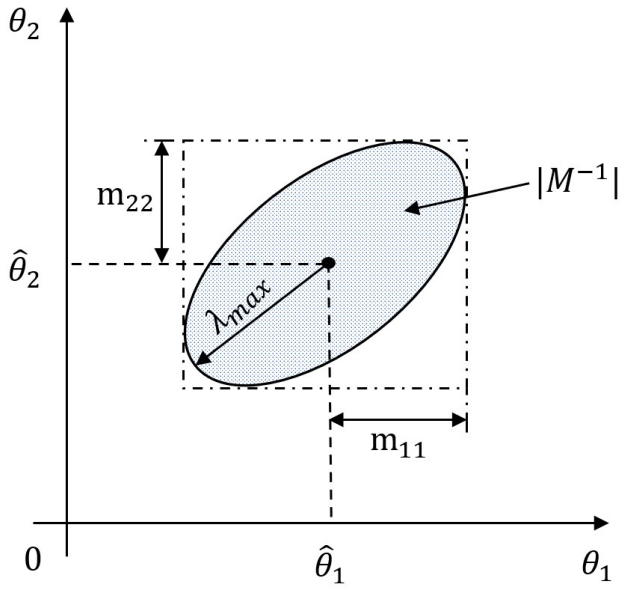

Figure 4.1: Confidence ellipsoid for parameters in two dimensions based on paper [61]

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

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

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 total parameter uncertainty or reducing 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 constraints.

In the optimal experimental design specific to the EMB system, further detailed constraints were considered. According to the requirement of the 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 [63] 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 be combined with reinforcement learning methods. An 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 the computation of the inverse of the FIM, M^{-1}, at each time step. If the dimension of the matrix M is large, this significantly increases the computational complexity and leads to longer computation times. Therefore, the objective function for the optimization problem needs to be modified to avoid matrix inversion operations in practical reinforcement learning applications. The approach to this modification is presented below.

Suppose the FIM is an n-dimensional non-singular matrix with eigenvalues λ satisfying 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 objective 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

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

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

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

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}. From the sensitivity calculation, it can be concluded that the system output is generally more sensitive to parameters with smaller numerical scales. As there can be significant differences in the values of different parameters, it is necessary to normalize the parameters when applying the other two optimality criteria to keep the variance of the different parameters comparable. The normalized 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 change in outputs in response to parameter variations, different output scales in a multi-output system can significantly affect the calculation of FIM and sensitivity. For example, if some parameters in a system only affect certain 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 means that the input excitation will yield a higher information content for the parameters that affect these output dimensions. Therefore, to ensure a balanced sensitivity and an accurate representation of the information content, not only the parameters but also the output scales should be normalized.

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

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

where D_y is a diagonal matrix defined as

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

Furthermore, the variance-covariance matrix of the measurement noise has to be normalized according to the following equation

$$\bar{R} = D_y^{-1} R D_y^{-T} \tag{4.20}$$

This normalization ensures that all outputs are brought to a comparable scale, preventing a 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 to obtain an accurate estimate of the parameters and ensures that the information content of all the output dimensions is represented at the same level.

4.3.4 Calculation of the Fisher Information Matrix for Discrete Systems

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

$$\begin{aligned} x_{d,k+1} &= f_d(x_{d,k}, u_k) \\ y_{m,k} &= h_d(x_{d,k}) + v_{o,k}, \quad k = 1, ..., n \end{aligned} \tag{4.21}$$

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

$$\begin{aligned} M &= \sum_{k=1}^{n} \left(\frac{d\bar{h}_k}{d\bar{\theta}} \right)^T \bar{R}^{-1} \left(\frac{d\bar{h}_k}{d\bar{\theta}} \right) = \sum_{k=1}^{n} (M_{\bar{\theta},k}), \\ \frac{d\bar{h}_k}{d\bar{\theta}} &= \begin{bmatrix} \frac{d\bar{h}_k}{d\theta_1} & \frac{d\bar{h}_k}{d\theta_2} & \cdots & \frac{d\bar{h}_k}{d\theta_n} \end{bmatrix}, \end{aligned} \tag{4.22}$$

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

$$\frac{d\bar{h}_k}{d\bar{\theta}_i} = \frac{\partial \bar{h}_k}{\partial x_d^T} \frac{\partial x_{d,k}}{\partial \bar{\theta}_i} + \frac{\partial \bar{h}_k}{\partial \bar{\theta}_i} \tag{4.23}$$

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]

$$\frac{d}{dt} \frac{\partial x}{\partial \theta}(t) = \frac{\partial f}{\partial x} \frac{\partial x}{\partial \theta}(t) + \frac{\partial f}{\partial \theta} \tag{4.24}$$

with initial condition

$$\frac{\partial x}{\partial \theta}(0) = \frac{\partial x_0}{\partial \theta} \tag{4.25}$$

Transferring this function to the discrete domain it becomes

$$\frac{\partial x_{d,k+1}}{\partial \bar{\theta}_i} = \frac{\partial f_{d,k}}{\partial x_d^T} \frac{\partial x_{d,k}}{\partial \bar{\theta}_i} + \frac{\partial f_{d,k}}{\partial \bar{\theta}_i}, \tag{4.26}$$

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

$$\frac{\partial x_{d,1}}{\partial \theta} = 0 \tag{4.27}$$

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

4.3.5 Implementation of the Calculation of the Fisher Information Matrix in Python

After completing the theoretical formulation of the FIM calculation, the corresponding model is implemented in *Python*. Since the equations in the previous section always calculate the gradient, automatic differentiation tools are differentiation tools are suitable for this task. This implementation uses the *Autograd* feature of *PyTorch* for automatic differentiation [64], which simplifies the calculation 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 shown in equation (4.22).
- *h(x)*, represents the output function that provides system outputs, such as motor position and velocity.
- *jacobian(x, u, theta)*, uses the *Autograd* feature of *PyTorch* to compute the Jacobian of the state function *f(x, u, theta)* in terms of both state variables and system parameters.
- *sensitivity_x()*, calculate the sensitivity of the states and outputs with respect to the system parameters recursively, as described in equation (4.23) and (4.26).
- *fisher_info_matrix()*, computes the FIM by accumulating the sensitivity information over all time steps.

This implementation not only ensures accurate and efficient sensitivity calculation throughout the simulation. Furthermore, as this script is incrementally updated, at each time step not only the FIM, but also the corresponding state variables and system outputs, as well as the sensitivity of each state variable and output with respect to each parameter, can be easily computed. This enables monitoring and evaluation of the evolution of the FIM and sensitivity over 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 shows the evolution of the system input u, the 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, an unexpected behavior was observed when calculating the logarithm of the determinant of the FIM. Figure 4.2(d) shows that the log determinant curve is discontinuous. Further examination of the determinant values from Figure 4.2(b) shows 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 probably due to floating-point errors inherent in the determinant calculation, particularly as the matrix approached 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 in Figure 4.2(f), the conditional number of the FIM unexpectedly spiked at several intervals during the experiment, causing the matrix to become ill-conditioned and leading to numerical instability during the determinant calculation.

In reinforcement learning, where the input trajectory is determined by the agent, it is not possible to anticipate when the FIM will become nearly singular and design the input to avoid this issue. In addition, the determinant of the FIM tends to grow rapidly over time, which can lead to potential overflow problems. To mitigate this problem, a method was developed to scale the FIM matrix to ensure numerical stability and prevent such problems during the determinant calculation.

In addition, a step reward function was designed based on the characteristics of reinforcement learning combined with the D-optimality criterion. The problem that needed to be addressed was that due to the incremental nature of the FIM computation, where the FIM at each time step is equal 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 poses a challenge for the initial computation of the reinforcement learning reward function. Consequently, an initialization method for the FIM has been developed and demonstrated to have no adverse effect on the optimization results.

4.4.1 Initialization of the Fisher Information Matrix

During the calculation of the FIM, an assumption is made

$$M_0 = 0 \tag{4.28}$$

It indicates that no information is obtained at the beginning of the experiment. Since the FIM is updated by a rank-1 update at each time step during training, initializing the FIM as in equation (4.28) poses a problem: at least during the first 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, the

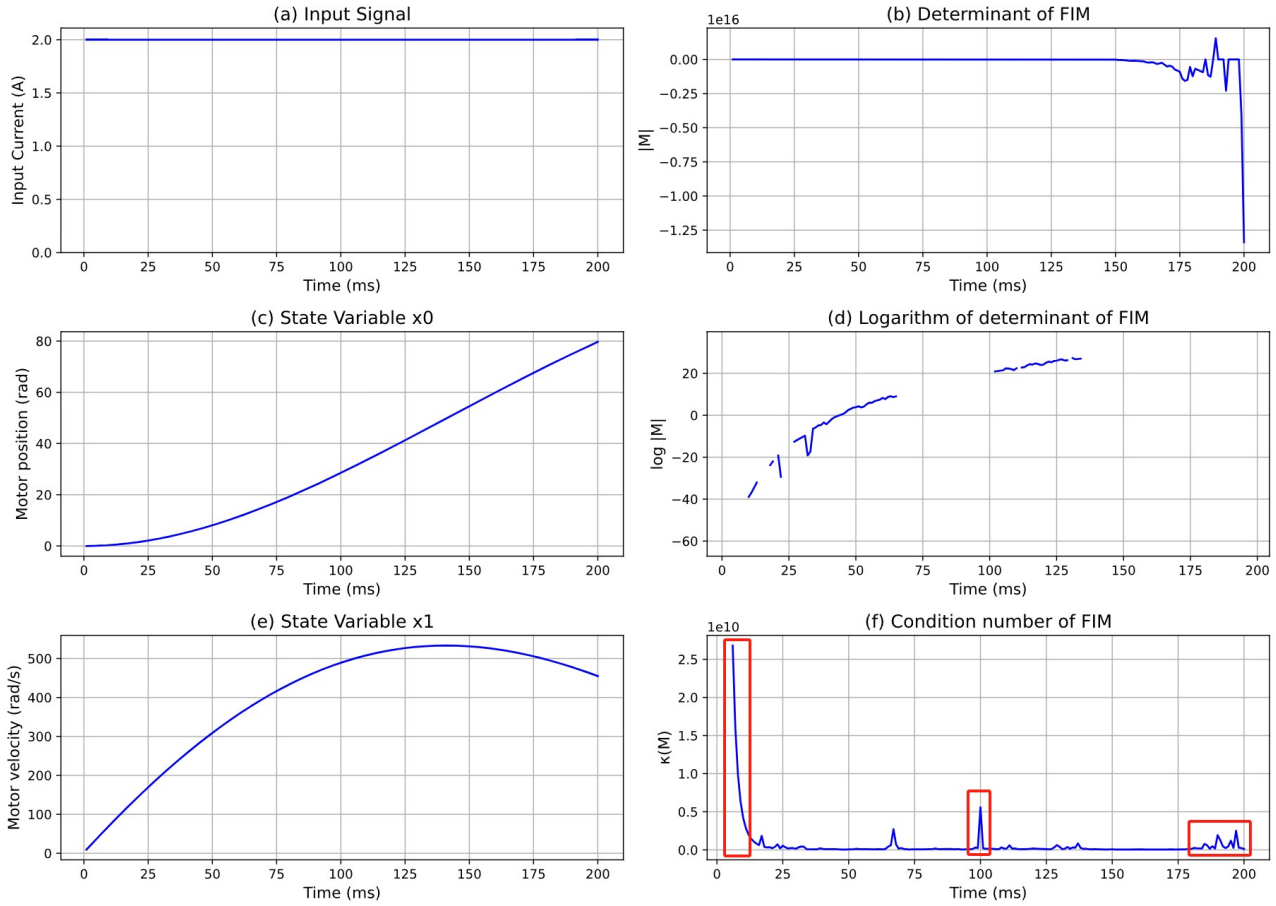

Figure 4.2: System Response Over Time to a Constant Input Trajectory: (a) System input u. (b,c) State parameters x_1 and x_2. (d) Determinant of the FIM $|M|$. (e) Logarithm of the determinant $log(|M|)$. (f) Condition number of the FIM, $\kappa(M)$

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

$$M_{init} = \epsilon I \tag{4.29}$$

Since adding an identity matrix to the FIM will definitely change the FIM, it could be proved that 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

$$\lambda_i(A) + \lambda_1(B) \leq \lambda_i(A+B) \leq \lambda_i(A) + \lambda_n(B) \tag{4.30}$$

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

Theorem 1 (Courant-Fischer Theorem). *[65] 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

$$\lambda_i = \max_{\substack{S\subseteq\mathbb{R}^n \\ \dim(S)=k}} \min_{\substack{\mathbf{x}\in S \\ \mathbf{x}\neq 0}} (\mathbf{x}^T\mathbf{A}\mathbf{x} + \mathbf{x}^T\mathbf{B}\mathbf{x}) \tag{4.31}$$

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

$$\mathbf{x}^T\mathbf{B}\mathbf{x} \geq \lambda_1(\mathbf{B}) \tag{4.32}$$

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

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

Similarly could be proved that

$$\begin{aligned} \lambda_i(\mathbf{A}+\mathbf{B}) &\leq \max_{\substack{\dim(S)=i \\ \mathbf{x}\in S, \|\mathbf{x}\|=1}} \min\left(\mathbf{x}^T\mathbf{A}\mathbf{x} + \lambda_1(\mathbf{B})\right), \\ &= \lambda_i(\mathbf{A}) + \lambda_n(\mathbf{B}) \end{aligned} \tag{4.34}$$

Thus the inequality (4.30) is proved.

When the FIM M_n is initialised with a diagonal matrix M_{init} with sufficiently small diagonal value $\epsilon > 0$, inequality (4.30) becomes

$$\begin{aligned} \lambda_i(M_n) + \lambda_1(M_{init}) \leq \lambda_i(M_n + M_{init}) \leq \lambda_i(M_n) + \lambda_n(M_{init}) \\ \lambda_i(M_n) + \epsilon \leq \lambda_i(M_n + M_{init}) \leq \lambda_i(M_n) + \epsilon \end{aligned} \tag{4.35}$$

Hence, the following equation is proved

$$\lambda_i(M_n + M_{init}) = \lambda_i(M_n) + \epsilon \tag{4.36}$$

Thus, initializing the FIM with $M_{init} = \epsilon I$ 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 normalisation approach with factor α_k is designed

$$\bar{M}_k = \alpha_k M_k \tag{4.37}$$

where $\bar{M}_k$ is the normalised FIM.

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

$$\begin{aligned} r_k &= log|M_k| - log|M_{k-1}| \\ &= log\frac{|M_k|}{|M_{k-1}|} \\ &= log\frac{\alpha_k|M_k|}{\alpha_k|M_{k-1}|} \\ &= log|\bar{M}_k| - log|\frac{\alpha_k}{\alpha_{k-1}}\bar{M}_{k-1}| \end{aligned} \tag{4.38}$$

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 updated by the rank-1 update, the FIM at time step k is

$$M_k = M_{k-1} + M_{\bar{\theta},k} \tag{4.39}$$

Hence, Equation (4.37) changes to

$$\begin{aligned} \bar{M}_k &= \alpha_k M_k \\ &= \alpha_k(M_{k-1} + M_{\bar{\theta},k}) \\ &= \frac{\alpha_k}{\alpha_{k-1}}\bar{M}_{k-1} + \alpha_k M_{\bar{\theta},k} \end{aligned} \tag{4.40}$$

Therefore, the update for the normalized FIM in this approach also does not require the calculation of the original FIM, thus avoiding the occurrence of a potentially large determinant.

A reasonable way to choose the initial normalized FIM is to make it the same as the initial FIM in the 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

$$\alpha_k = \left(\frac{det\left(M_{init}\right)}{det\left(\bar{M}_{k-1}\right)}\right)^{\frac{1}{n}} \tag{4.41}$$

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

4.5 Reinforcement Learning

In this thesis, reinforcement learning methods are used to perform input excitation optimization for EMB parameter estimation. Reinforcement learning is a type of machine learning. 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 autonomously accomplish set

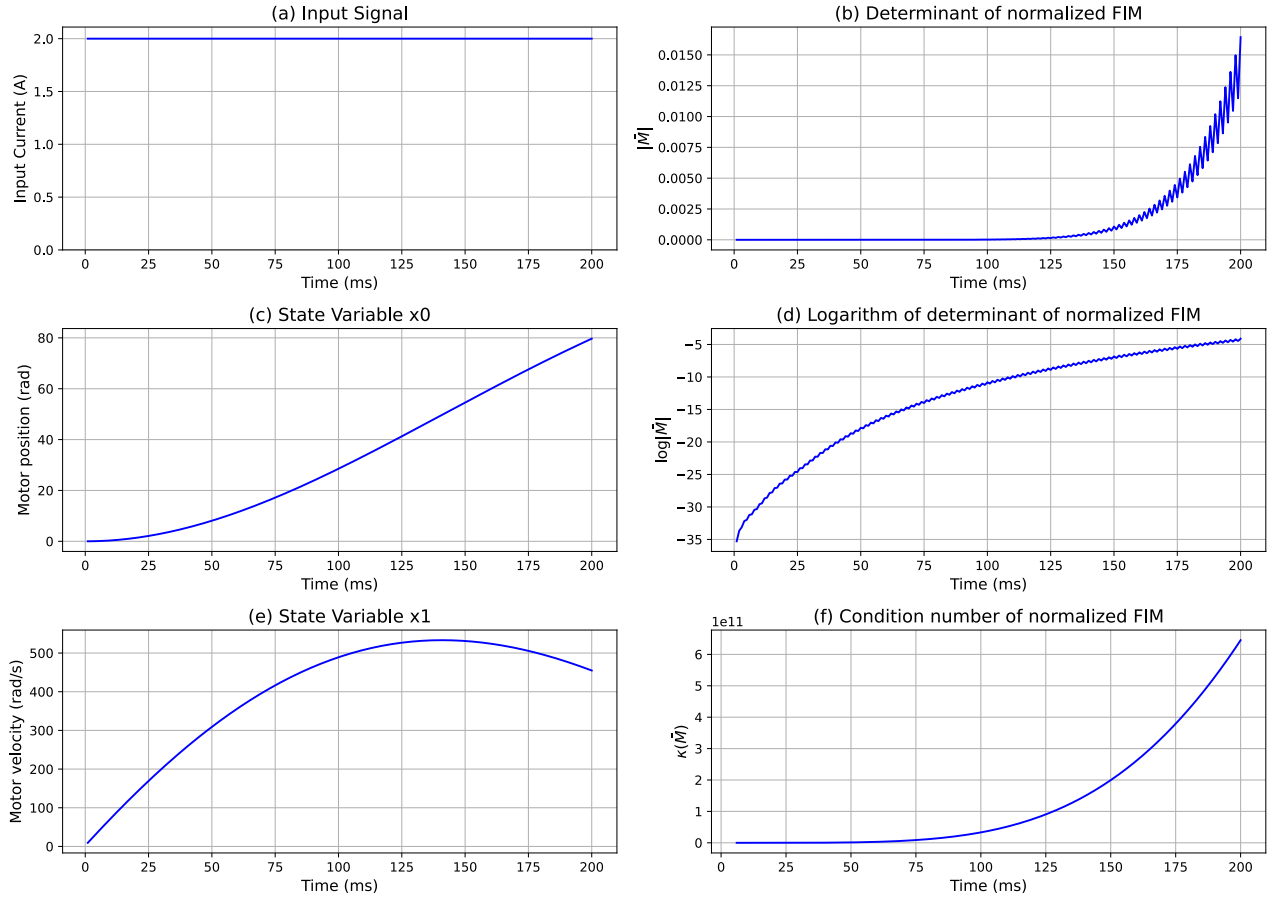

Figure 4.3: System response over time to a constant input trajectory after initialization and normalization of FIM: (a) System input u. (b,c) State parameters x_1 and x_2. (d) Determinant of the FIM $|\bar{M}|$. (e) Logarithm of the determinant $log(|\bar{M}|)$. (f) Condition number of the FIM, $\kappa(\bar{M})$

tasks. In the application of this work, the environment is the simulated EMB system and the agent is responsible for generating the input excitation used for parameter estimation.

Reinforcement learning based on Markov decision processes (MDPs) is an abstraction of the real world. An MDP is defined by a tuple (S, A, P, R, γ), where S is the set of states, A is the set of actions, P is the transition probability between states, R is the reward function, and γ is the discount factor for future rewards [66]. 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 the process continues in the same way. The goal of the agent is to find an optimal policy $\pi(a|s)$ that, based on the state s, could perform the action a and maximize the expected cumulative return.

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \tag{4.42}$$

And the expected cumulative return based on strategy π_θ is

$$J = \mathbb{E}_\pi(G_t) \tag{4.43}$$

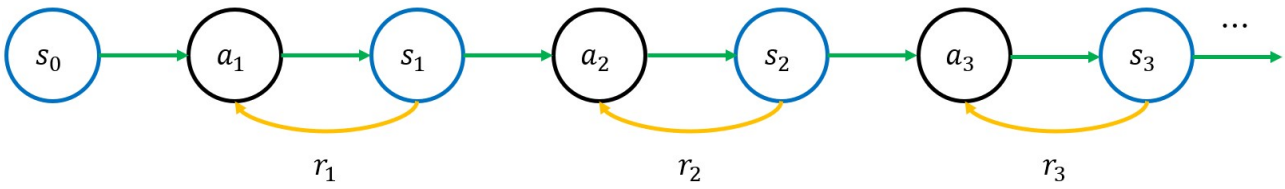

Figure 4.4: Illustration of Markov decision processes

Action-value function and state-value function form the core of 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 policy π

$$\begin{aligned} Q_\pi(s,a) &= \mathbb{E}\left(G_t \middle| s_t = s, a_t = a, \pi\right) \\ &= \mathbb{E}\left(\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \middle| s_t = s, a_t = a, \pi\right) \end{aligned} \tag{4.44}$$

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

$$\begin{aligned} V_\pi(s) &= \mathbb{E}\left(G_t \middle| s_t = s, \pi\right) \\ &= \mathbb{E}\left(\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \middle| s_t = s, \pi\right) \\ &= \sum_{a \in A} \pi(a|s) Q_\pi(s,a) \end{aligned} \tag{4.45}$$

Almost all reinforcement learning algorithms are based on the calculation of action-value function and state-value function. The relationship between these two functions is established by the advantage function A, which 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

$$A(s,a) = Q_\pi(s,a) - V_\pi(s) \tag{4.46}$$

Policy gradient methods are a class of methods that directly optimize the policy π [67]. The basic idea is to increase the probability of actions that can produce positive benefits and decrease the probability of negative actions. The advantage function is employed in this method to compute the gradient of the objective function

$$\nabla J(\pi) = \mathbb{E}\left[\nabla \log \pi(s,a) A(s,a)\right] \tag{4.47}$$

Classical 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 requires the action space to be discrete, and in the case of a huge dimension of the state space, this leads directly to a huge dimension of the Q-table, limiting 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) [68]–[71]. Expect DQN and its variant, all other DRL algorithms are based on actor-critical methods.

Compared to other DRL algorithms, PPO is known for its robustness. Since the application scenario in this thesis is a custom EMB model environment, unlike established reinforcement learning environments that are widely used for benchmarking, the custom environment can be highly sensitive to the parameters of deep reinforcement learning algorithms, making tuning difficult. 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 of PPO is that the clipping mechanism prevents large updates, making it more stable and robust compared to other actor-critical methods, which makes it suitable for the purposes of this thesis.

4.5.1 Actor-Critic Methods

As the name implies, this method contains two main components: the Actor and the Critic. Actor is used to output the action, while Critic is responsible for giving evaluation to those actions. In DRL, these two components are represented by two separate neural networks, a policy network for Actor and a value network for Critic. The policy network uses the policy gradient method combined with the reward function $A(s, a)$ to update the network, as shown in equation (4.47).

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 the mean squared error (MSE) between predicted value and target return. For the one-step TD error, the critic loss takes the form

$$L_V = \frac{1}{2}\left(V(s_t) - G_t\right)^2 \tag{4.48}$$

4.5.2 Proximal Policy Optimization

Proximal Policy Optimization is a reinforcement learning algorithm based 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 [71]. The objective function of TRPO is introduced by [72]

$$J(\theta) = \hat{\mathbb{E}}_t\left[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}\hat{A}_t\right] \tag{4.49}$$

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 there are difficulties in choosing β.

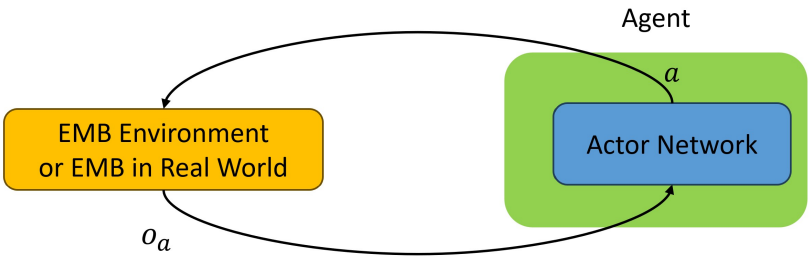

Figure 4.5: Testing Actor with EMB environment after training

The idea of PPO is to set a lower and upper bound for Equation (4.49), to represent the constraint and prevent the policy from making large updates. These bounds are converted into the following clip function with hyperparameter ϵ

$$J^{CLIP}(\theta) = \hat{\mathbb{E}}_t \left[\min \left(\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)} \hat{A}_t, \text{clip}(\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}, 1-\epsilon, 1+\epsilon)\hat{A}_t \right) \right] \tag{4.50}$$

With this clipping mechanism, the calculation of the gradient of the objective function does not require KL divergence, second-order derivatives, or constraint solvers. The policy update is highly constrained, resulting in more robust performance.

Different from other Actor-Critic networks, the memory buffer of PPO is not designed based on episodes, nor does it set a maximum number of episodes during training [73]. Instead, PPO uses a vectorized environment and runs N independent environments in parallel during the training. Each of these parallel environments 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 of 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 performed accurately 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 a decision and give the correct input trajectory. Thus, the gap between the theoretical simulation of optimal experimental design and its practical application must be considered in the design of the framework.

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 shown in Figure 4.5, at the end of the training, when the agent is actually used to 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 remains unknown.

Partial observability is not unique to the EMB system. For example, in robotics, robots are usually first trained in the simulator with all state information available. However, when the robot is deployed after training, it may only have images as input, which also leads to the partial observability problem. To address this problem, partially observable Markov decision processes (POMDPs) have been proposed [74]. A POMDP is an MDP in which the agent is unable to observe the current state. Instead, it makes an observation based on the action and the resulting state, and the goal of maximizing the expected discounted future reward remains the same. POMDPs are described by the tuple $\langle S, A, T, R, \Omega, O\langle$ [74], where S, A, T, R retain the same meaning as in 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 [75], [76]. Most POMDP applications rely on offline algorithms, which have the disadvantage that the training time is too long, also it can not deal with small changes in the environment because the entire offline policy needs to be recalculated. To address these issues, several online POMDP algorithms have been proposed and studied [77]–[79].

The Asymmetric Actor-Critic framework is a variant of the Actor-Critic framework, where the full state in observation space is used to train the critic, while partial observation is used to train the actor [80]. This framework has been applied to image-based robot learning and contextual learning tasks [81]. 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 [82].

Since PPO is also based on Actor and Critic networks, it should be considered 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 research, in this thesis, an asymmetric PPO framework is introduced. As shown in the Algorithm 1, compared to the classical PPO algorithm, this framework employs an asymmetric input strategy for the actor and critic networks. Figure 4.6 illustrates an asymmetric PPO framework running independent environments in parallel during training. 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 velocity, 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 in the observation space for actor and critic are 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 [83].

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

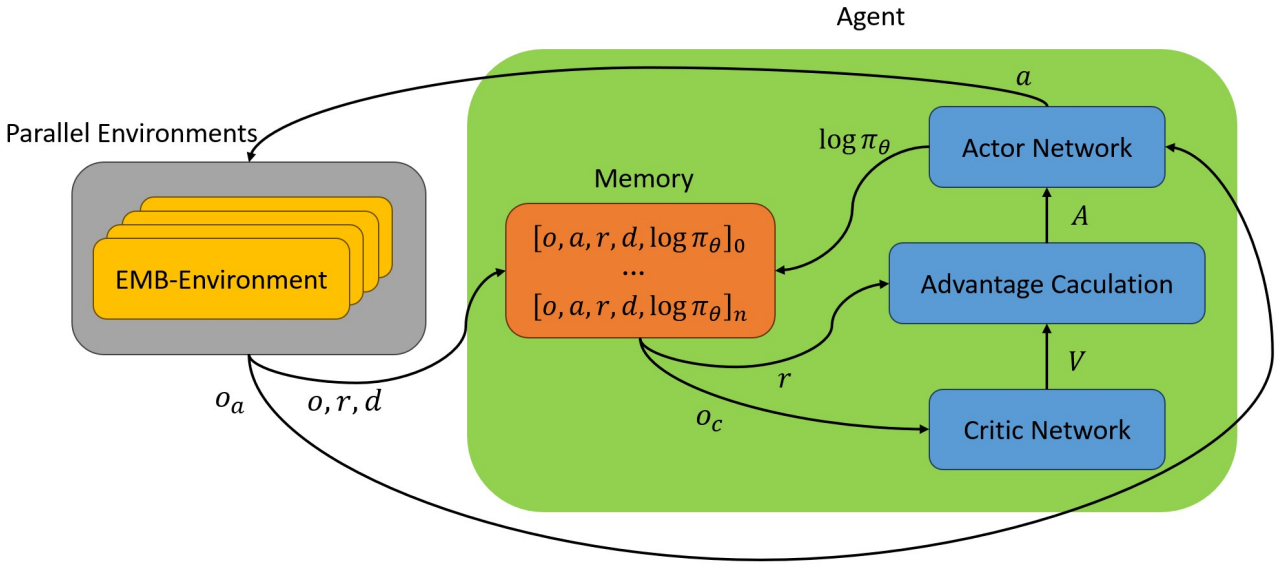

Figure 4.6: Asymmetric PPO framework running independent environments in parallel during training

4.6 EMB Environment Setup for Reinforcement Learning

After solving the computational aspects of the optimal experimental design for the EMB system, a remaining challenge is to 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 [63]. 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.

The environment is the core part of Gymnasium. It is the instantiation of the RL environment and

Algorithm 1 Asymmetric PPO Framework

Initialize Actor-Critic networks
Initialize memory buffer $\mathcal{B}$
Sample initial observation state o_0
for update $= 1, 2, \ldots$ **do**
  for actor $= 1, 2, \ldots, N$ **do**
    for $t = 0, T-1$ **do**
      Get action a_t and action distribution $\log \pi_\theta$ using actor policy $\pi_\theta(o_{a,t})$
      Execute action a_t, receive reward r_t, next observation o_{t+1}, and done flag d_t
      Get value V_t from critic network
      Store $(o_t, a_t, r_t, d_t, V_t)$ in memory buffer $\mathcal{B}$
      if done flag $d_t == 1$ **then**
        Sample initial observation state o_0
      end if
    end for
  end for
  Compute returns R and advantages A with stored trajectories in memory buffer $\mathcal{B}$
  for epoch $= 1, 2, \ldots, K$ **do**
    Shuffle memory buffer $\mathcal{B}$ and divide into mini-batches
    for each minibatch **do**
      Update actor network using $(o_{a,t}, a_t, r_t, A_t)$
      Update critic network using $(o_{c,t}, a_t, r_t, V_t)$
    end for
  end for
  Clear memory buffer $\mathcal{B}$
end for

enables interaction with RL algorithms [63]. 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 are implemented in these components. The following part describes how each component is built in a customized EMB environment.

The input to the EMB system is the motor current, which ranges from -6A to 6A, and thus 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 must be considered in the context of the task requirements and the structure of the RL algorithm. Parameters that can be included in the observation space include the actual and measured values of the motor position and velocity, the prior values of the 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 to the EMB system and the motor position and velocity in the next step are updated. It also calls the function used to compute the FIM from the previous section and updates the states in the observation space. The $step$ function also contains two conditions that automatically trigger the $reset$ function. The first is when the time step reaches 500, which means that the experiment has lasted 500 ms and is over. In addition, the agent must learn to control the motor so that it does not exceed speed

and position limits. If these limits are violated 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 training performance of the agent. 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 different 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 ensures 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 input based on specific parameter values during the training process, but will learn the optimal policy under the parameter distribution based on experience.

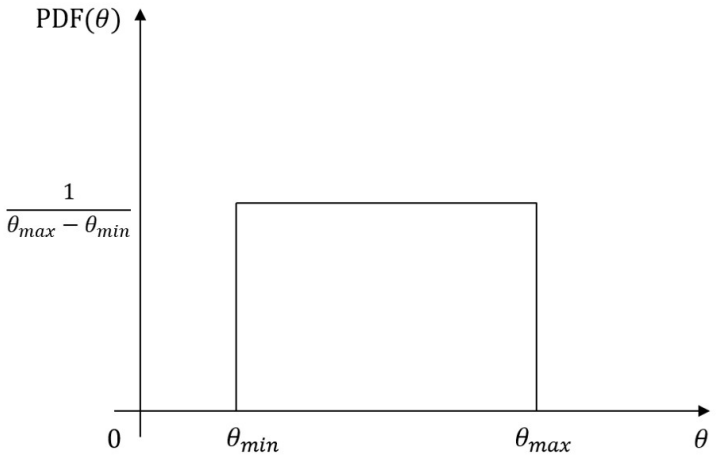

Figure 4.7: Uniform distribution of parameter θ

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 to the Single Parameter Excitation

In this chapter, the previously described reinforcement learning approaches are applied to the EMB system for efficient 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 is chosen as the single parameter to be estimated. An analysis of the behavior 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)

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

Since only a single parameter needs to be estimated in this case, 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 the calculation of the FIM is strongly relevant to 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 (4.26), the sensitivity of each state variable with respect to viscous friction is

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

Note that, according to the EMB model setup, the zero point of the motor position is defined as the contact point, since the clamping force only existed when the brake lining is in contact with the brake drum. Therefore, the term related to the clamping force $-\frac{\gamma k_1}{J}$ in this equation only exists when x_1 is greater than zero; otherwise, this term is zero.

This equation already shows some similarity to some well-known dynamical system. The analysis 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

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

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 (4.23) 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 conclude that based on the goal of obtaining a higher level of information, and also higher FIM element value in this single parameter case, because the matrix is one-dimensional, the approach is to increase 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 speed will result in a greater gain of information.

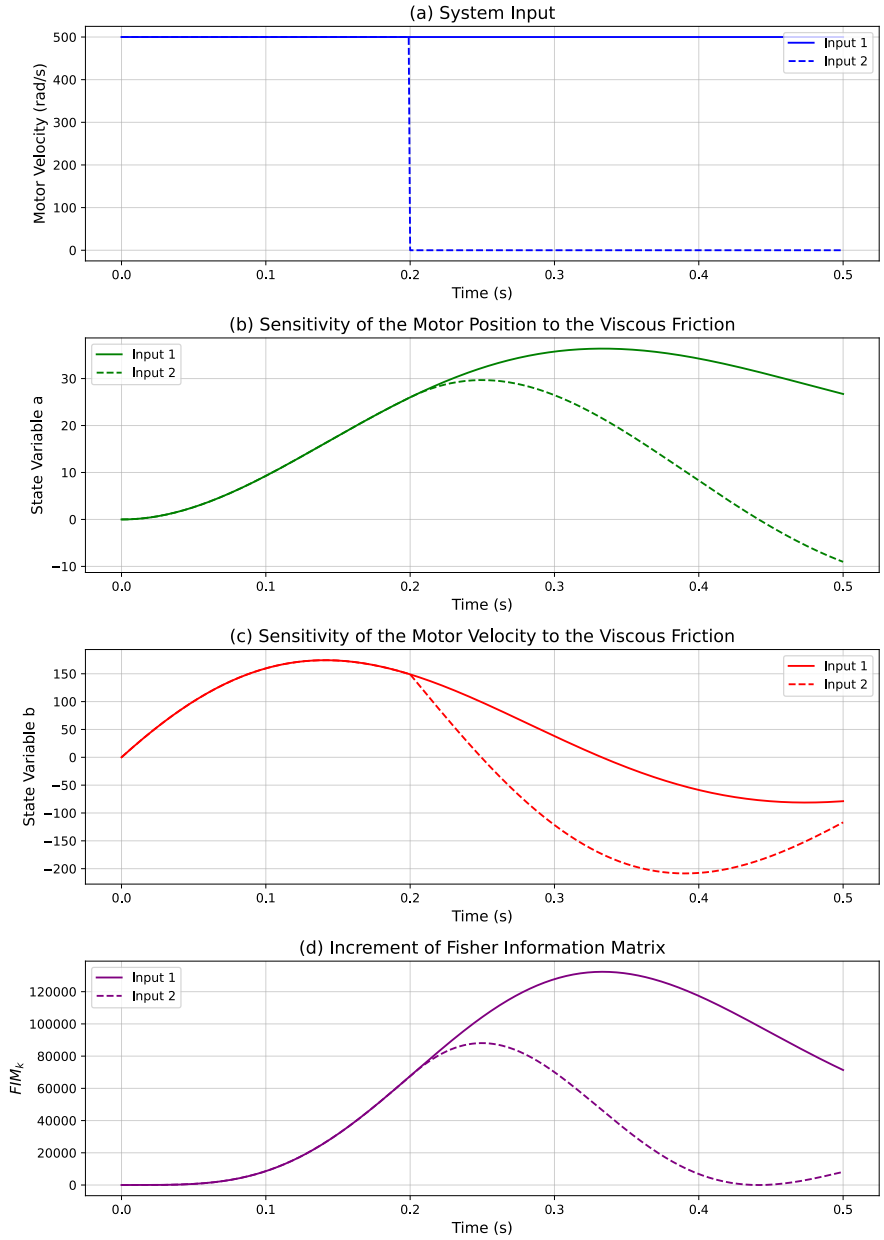

Figure 5.1: Comparison of system dynamics and increment of FIM with different motor velocity trajectory as inputs

The conclusions derived from the above FIM calculation model were validated through empirical 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 provides valuable insights for analyzing subsequent experimental results.

5.2 Comparison of Performance Between Parameter Initialization Methods

The analysis of optimal experiment design begins with testing the effectiveness of the PPO algorithm and comparing the performance of two different parameter initialization methods designed in Section 4.7.1. In this thesis, the PPO algorithm is constructed with a continuous action space, where the action range of the agent is from -6 A to 6 A. The observation space of the reinforcement learning environment includes the following elements: motor position, motor velocity, time step, viscous friction parameter, and the value of the FIM elements.

It is important to note that this section uses an idealized environment in which both the actor and the 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. However, it could reduce the learning difficulty for the agent and focus on the performance comparison of these two parameter initialization methods. Therefore, the asymmetric PPO algorithm is not used in this section.

According to the experiment requirement, the total experiment time should not exceed 500 ms. In addition, the motor position and velocity should return to the origin at the end of an experiment. Since this section is dedicated to the optimal input excitation optimization problem, the return to origin task for the EMB is excluded. Therefore, the experiment duration is reduced to 300 ms. In other words, the goal of the experiment here 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 these two parameter initialization methods 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 (4.22), 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 velocity, 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 current constraints, no specific penalty is added to the reward function because the continuous actions generated by PPO are 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 current output from the agent, ensuring it does not exceed the maximum current that can be input to the EMB motor.

Considering the problem of constraints on the state-space parameters of the system, since the motor moves to an unsafe state will cause the termination of current episode, a very intuitive approach is to penalize the agent in this case. This penalty discourages the agent from entering such state. This approach has been validated in many classical reinforcement learning problems. However, the choice of the penalty value is critical. If the penalty is too small, the agent might ignore the danger of unsafe states. If the penalty is too high compared to the regular step reward, the agent may avoid taking any action to keep safety from the start. After several adjustments based on training, a 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 an unsafe 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 $x_{1,d} = 75$, 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

$$r_k = \begin{cases} -4 \times 10^4, & \text{if in unsafe state,} \\ M_k - M_{k-1} - 200(x_2 - x_{2,d})^2, & \text{if in risky state,} \\ M_k - M_{k-1}, & \text{otherwise.} \end{cases} \tag{5.14}$$

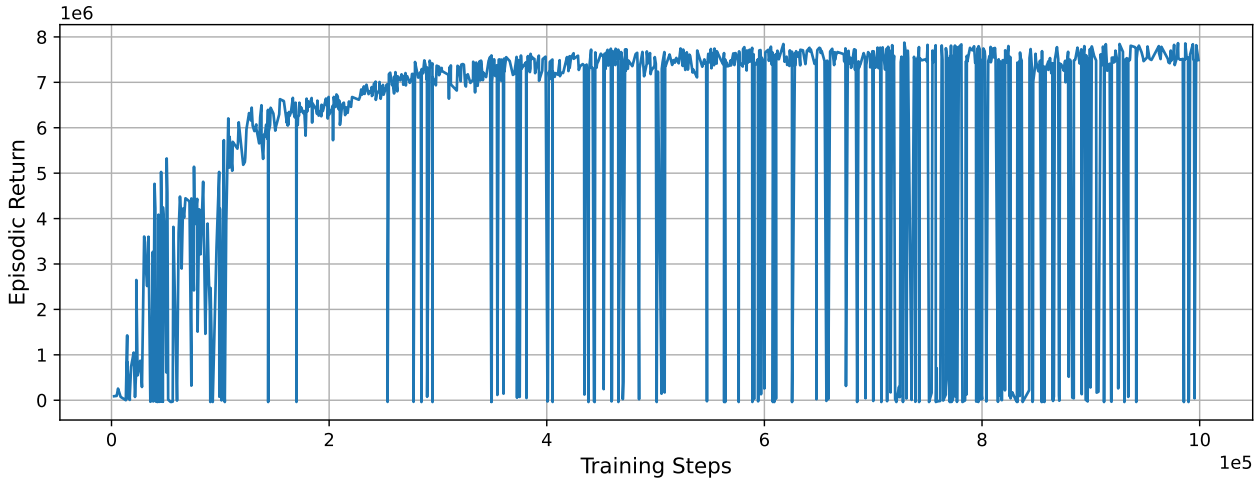

Figure 5.2: Episodic return during training with fixed parameter initialization method

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.

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 was 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 [84]. 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 took 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 current of the EMB to improve the efficiency of the viscous friction parameter identification.

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

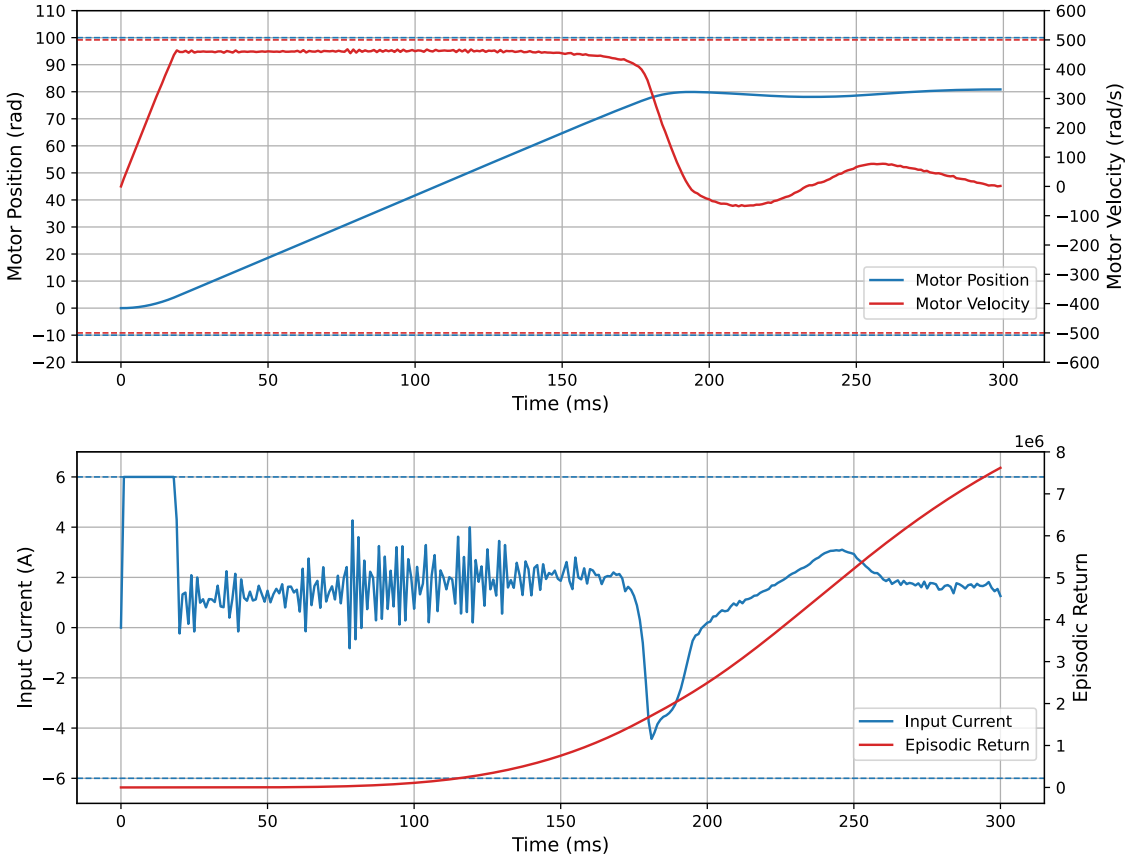

Figure 5.3: Performance of the agent trained with fixed parameter initialization method

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.

The performance of the agent trained with the 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 current, 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 behavior 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 current 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 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 reward that can be obtained in the end. Therefore, the policy of the agent is 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, 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 parameter.

The performance of the trained agent was evaluated in the EMB environment by conducting 5 experiments with parameters selected from uniformly gridded values in the same range used for training, from $(1 \times 10^{-5}, 5 \times 10^{-5})$. 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 behavior. 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 current adjustments, ensuring that the EMB system constraints are not violated.

Another thing to notice in the first 180 ms is that although the motion trajectory of the motor is nearly identical, as the viscous friction value increases, the accumulating reward of the experiment with higher parameter value is also higher. This is because the viscous friction itself is included in the calculation of the sensitivity, as shown in Equation (5.6).

To better compare the performance of different parameter initialization methods, both agents mentioned above 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 agents, 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

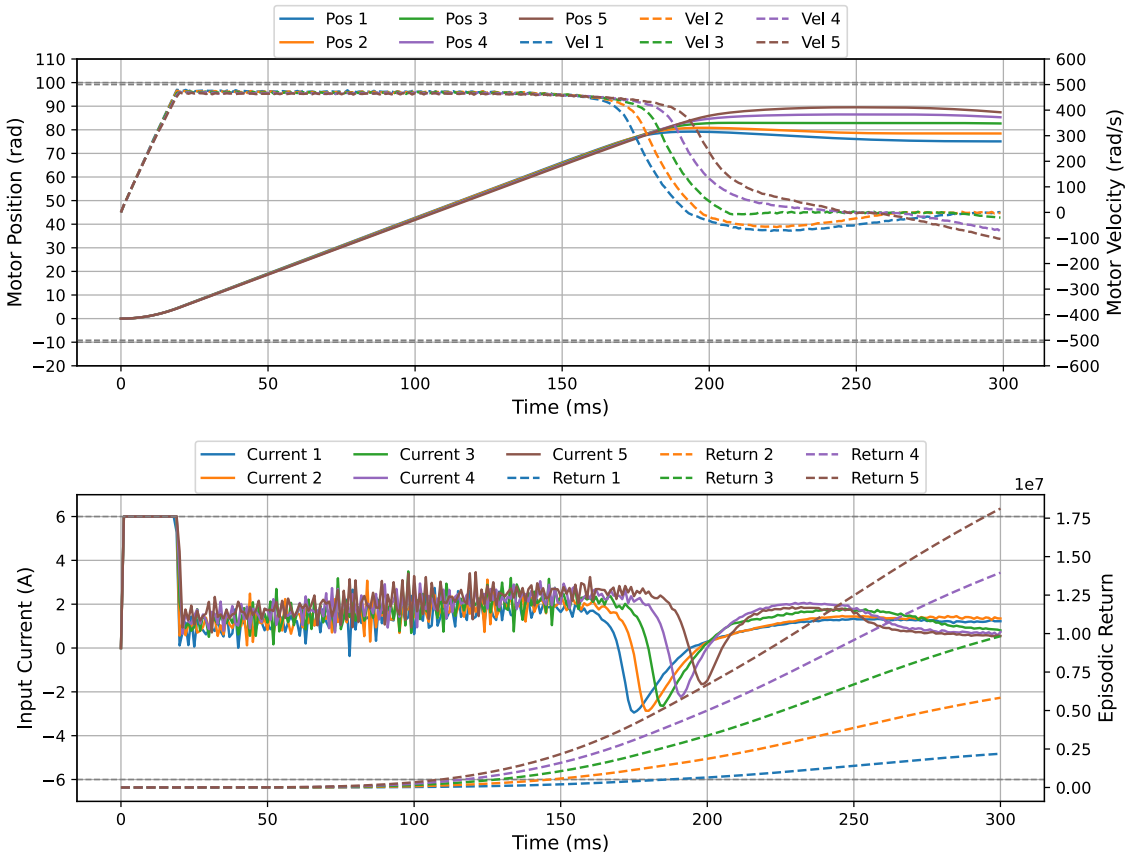

Figure 5.4: Performance of the agent trained with random distribution-based initialization method in 5 experiments with viscous friction coefficient selected from uniformly gridded values (experiment 1: $f_v = 1 \times 10^{-5}$, experiment 5: $f_v = 5 \times 10^{-5}$)

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.

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, as shown in detail in Algorithm 1, which more closely reflects real-world scenarios.

The random distribution-based parameter initialization method is still employed. However, to match actual experimental requirements, in the asymmetry PPO framework, the information available to the

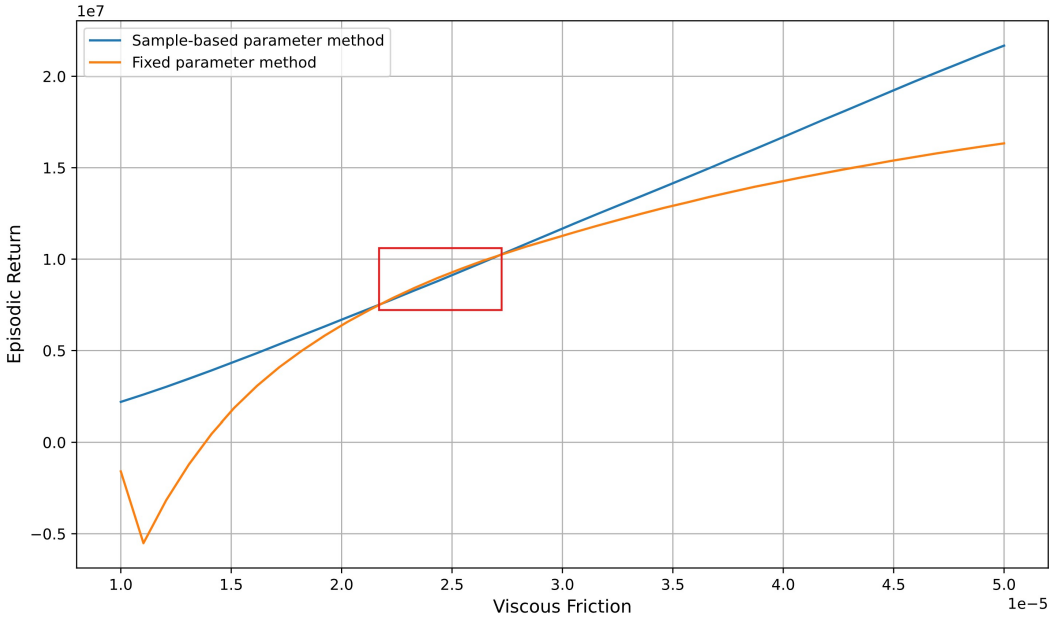

Figure 5.5: Performance comparison of different parameter initialization methods in viscous friction range tests

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 is sampled from the distribution during training. In the meanwhile, the critic network have access to all the information, as shown in Table 4.2. Since the actor network receives less information, this can increase the difficulty of the agent to learn efficiently. In this section, the performance of asymmetric PPO under these conditions is experimentally investigated, focusing on whether the trained agent can reduce its reliance on the a priori parameter values.

Furthermore, additional content related to realistic experimental requirements has been incorporated. According to the 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 in 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 efficiently perform the reset, it cannot optimize the motor trajectory from the perspective of parameter estimation during the reset, resulting in a loss of potential information. Therefore, in this section, the reward function of reinforcement learning is carefully designed to fully utilize the learning capabilities of the reinforcement learning agent. So that the agent is able to optimize parameter estimation while handling complex tasks, and solve the optimization problem according to the requirement in 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, 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 zero, in which the input excitation is also optimized to maximize the information content. Using the asymmetric PPO architecture, two reward function design approaches are 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 agents. 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

$$r_k = \begin{cases} -1 \times 10^7, & \text{if in unsafe state,} \\ \left(1 - \left(\frac{k}{T}\right)^2\right) \Delta M_k - 200(x_1 - x_{1,d})^2 - 0.1 \left(\frac{k}{T}\right)^2 \left((10x_1)^2 + x_2^2\right), & \text{if in risky state and } k > 300, \\ \Delta M_k - 200(x_1 - x_{1,d})^2, & \text{if in risky state and } k \leq 300, \\ \left(1 - \left(\frac{k}{T}\right)^2\right) \Delta M_k - 0.1 \left(\frac{k}{T}\right)^2 \left((10x_1)^2 + x_2^2\right), & \text{if in safe state and } k > 300, \\ \Delta M_k, & \text{otherwise.} \end{cases} \tag{5.15}$$

In this function, $\Delta M_k = M_k - M_{k-1}$ is the incremental change in the Fisher information matrix. T indicates the total number of steps in an episode. Due to the integration of the return to origin task, the running time of the experiment is now set to 500 ms. This allows the agent to get a higher reward before encountering a constraint violation. Hence, the penalty for the unsafe state has been adjusted to -1×10^7.

This 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 are weak, only $\left(\frac{300}{500}\right)^2 = 0.36$ of the the penalties is added to the function. But it increases quadratically as the episode progresses and quickly grows to 1 in the end.

After designing the reward function, training was conducted with random distribution-based parameter initialization method. After training, the agent was tested 5 times in EMB environments with parameters selected from uniformly gridded values, same as the method used previously. The difference in this section is that the selected viscous friction parameter remains unknown to the agent, and the agent's policy is 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

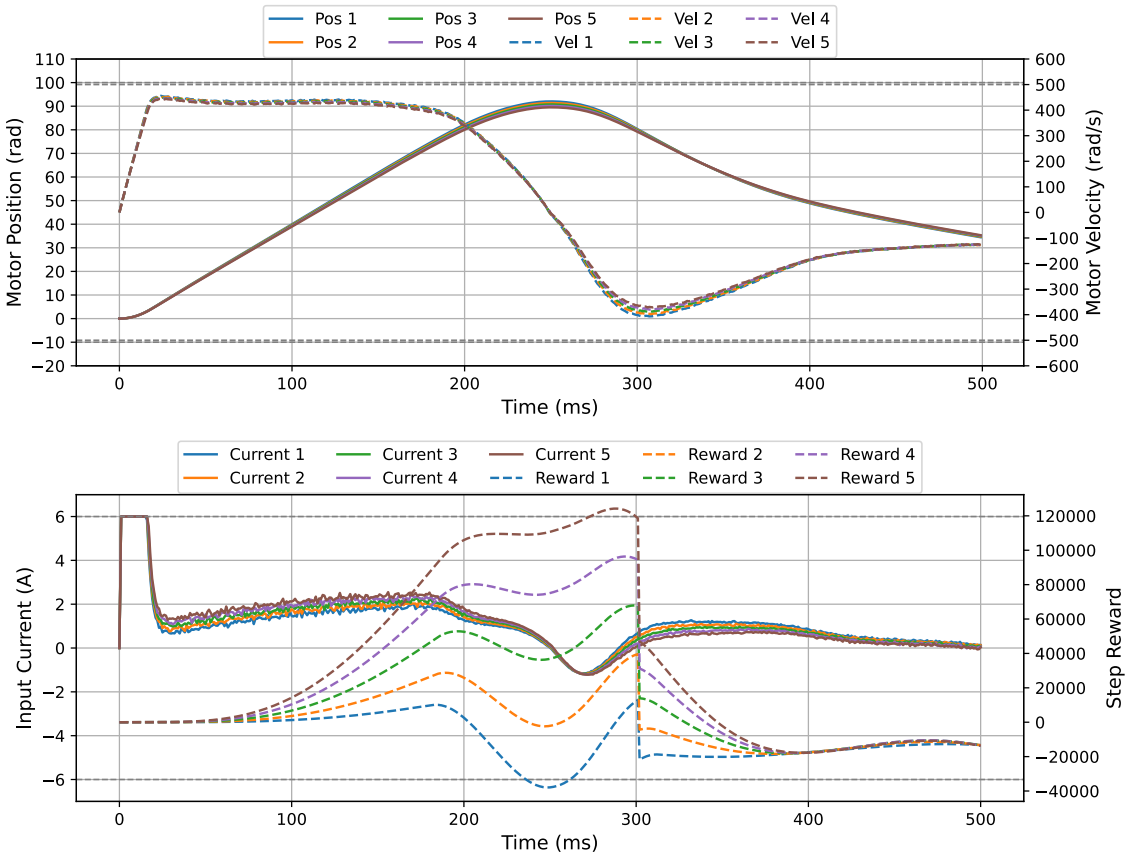

Figure 5.6: Performance of the agent trained with adaptive reward function in 5 experiments with viscous friction coefficient selected from uniformly gridded values (experiment 1: $f_v = 1 \times 10^{-5}$, experiment 5: $f_v = 5 \times 10^{-5}$)

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 reduces 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. This means that the agent could still learn a similar policy under the asymmetric PPO framework, which could maximize the information content for parameter estimation.

Starting from 200 ms, as the motor position enters the risky interval, the agent reduces motor velocity by reducing the current 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 friction values. 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, a conflict between velocity and position in penalty term in the reward function design during the return-to-zero phase could be observed. 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.

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 [85]–[87]. 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. A quintic polynomial trajectory of EMB motor motion can be described as

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

where $\theta(t)$ represents motor angle trajectory and $\dot{\theta}(t)$ represents motor velocity trajectory. These equations can be solved with 6 boundary conditions using the *np.linalg.solve()* function when each time

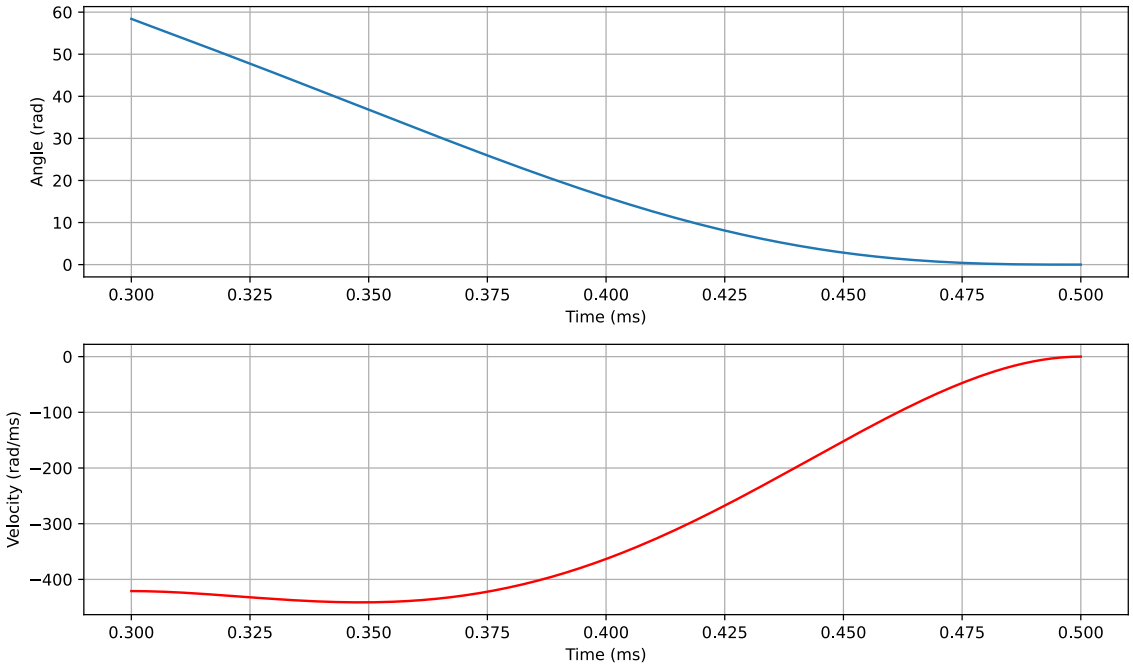

Figure 5.7: Quintic polynomial trajectory for return to origin task

the experiment reaches 300ms. An example of the trajectory generated from this design is illustrated in Figure 5.7.

During the second phase of the experiment, the designed trajectory is not provided to the agent, but used for the reward calculation. Penalties are applied based on the deviation of the EMB motor velocity and position from the trajectory. The increment of the FIM value during this 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 getting higher information content and returning to the origin, where the agent prioritizes following the trajectory to return to the zero point. Only after this task is accomplished does the agent 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 deviates from it to maximize the potential FIM, while still ensuring it can return to the 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 increases 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 training is also conducted with distribution-based parameter initialization method as before. After training, the agent was tested in the same way as previously and the 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 experiment, demonstrating high robustness to system parameters as well as good generalization ability. Starting from 300 ms, the agent's actions guide the motor to follow the quintic polynomial trajectory, eventually bringing the motor's velocity and position to zero regardless of the value of the viscous friction

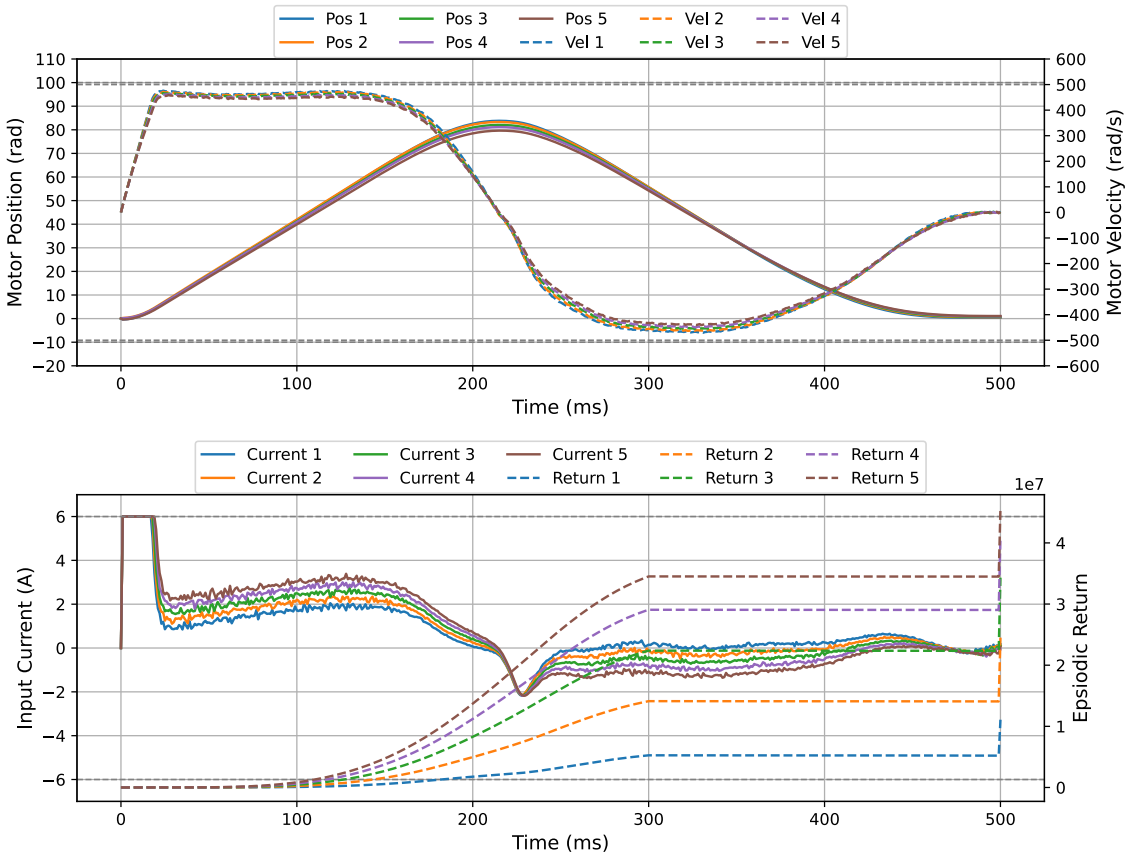

Figure 5.8: Performance of the agent trained with trajectory-based reward function in 5 experiments with viscous friction coefficient selected from uniformly gridded values (experiment 1: $f_v = 1 \times 10^{-5}$, experiment 5: $f_v = 5 \times 10^{-5}$)

parameters. This could be proved by the fact that the episodic return in the second phase did almost not decrease, so the penalty for deviating from the designed trajectory is small. According to the setup of the reward function, the agent is rewarded with the cumulative FIM increment in the second phase if it successfully completes the return-to-zero task. Since the test with the smallest value and the largest value could all accomplish the task, the episodic reward of these 5 experiments is suddenly higher at 500 ms as shown in Figure 5.8.

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 of 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 information content by deviating from the preset trajectory to enhance FIM performance and efficiency of parameter

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

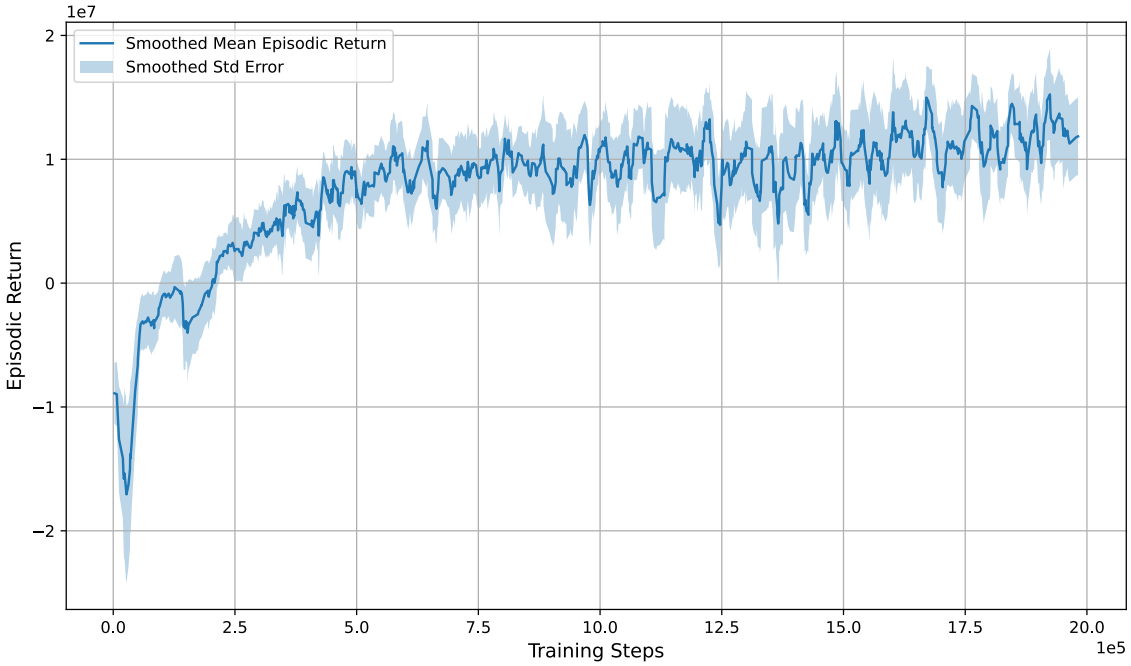

Figure 5.9: Training curve of asymmetric PPO framework with trajectory-based reward function across 5 random seeds

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 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 to 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 extended 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 Analysis of 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 $\bar{\theta} = \left[\bar{f}_v, \bar{k_1}\right]^T$. Thus, in the case of both viscous friction and stiffness characteristics parameters included, Equation 5.6 becomes

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

with term related to Equation (3.13) only present when x_1 is greater than zero.

If $\frac{\partial x_{d,k}}{\partial \theta}$ 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-parameter 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 systems have different inputs: 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-Optimality

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 uniform 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

observed an additional system parameter due to the increased dimension of the FIM. However, since the FIM is a symmetric matrix, it is not necessary to put all 4 elements of the 2-dimensional FIM into the observation space. Only storing the upper-triangular elements already provides the critic network with full information.

6.2.1 Underpeformance 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 lead to a big change in $|M|$, causing 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. This reward function design is not conducive to reinforcement learning, as the step reward in the environment remains very small for most of the time, which may lead to potential numerical issues, also it is not the common way to design reward function in RL.

Therefore, the reward function construction based on D-optimality was reconsidered, and two scaling methods based on Equation (4.13) were proposed. The idea is to see if changing the reward design could be beneficial for the learning of agent.

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

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

while Method 2, moves one step more than Method 1, add an additional parameter $\beta = 2$ to adjust the proportional relationship between the original step reward

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

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 square wave 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 experiment.

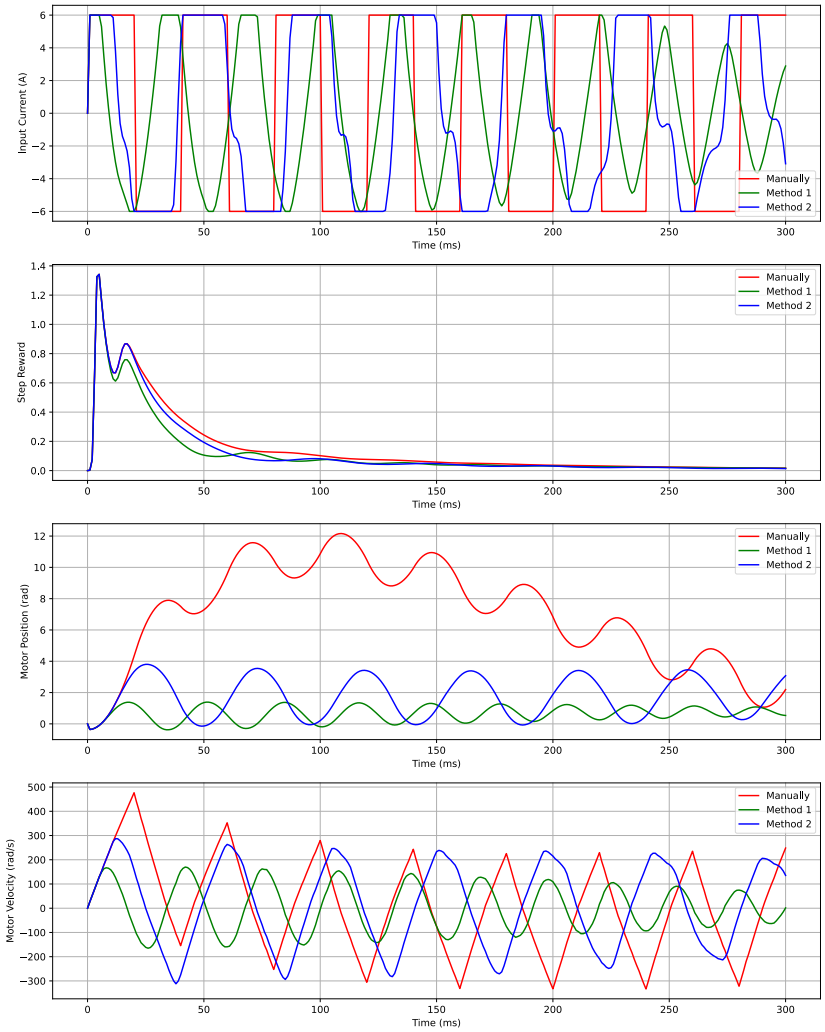

Figure 6.1: Performance comparison of agent based on reward-modified method and manually designed input trajectories

In order to fairly compare the performance of these two methods, the step reward calculations shown in Figure 6.1 are based on the original equation (4.13), without scaling based on these two methods. From 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. The FIM calculation based on the logarithm of the determinant produced a high step reward value at the beginning, and the difference between the two methods was almost negligible at the end. The main difference in the optimization of $|M|$ 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 with Method 1, which avoided accepting high current inputs for longer periods of time, resulting in lower speed and position of the EMB motor compared to Method 2.

Based on the behavior of the agent with both methods, it can be suggested that periodic system inputs might be possibly helpful for efficient 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 A to 6 A was introduced for comparison.

From 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 the D-optimality criterion has limitations in the multi-parameter identification problem of EMB systems, and further research based on other optimality criteria is necessary.

6.3 Performance of the Reward Function Design Based on E-Optimality

The advantage of using the D-optimality criterion to construct the reinforcement learning reward function is that, for the multi-parameter FIM, its minimum eigenvalue starts at 0 at the beginning of the experiment and increases as the parameter identification experiment proceeds. This feature is similar to the FIM characteristics during single parameter estimation. In the last section, the return of an episode in reinforcement learning based on D-optimality is largely dependent on the agent's behavior at the beginning of the experiment. Consequently, the actor's behavior at later stages has a lesser influence. Therefore, using E-optimality to construct the reinforcement learning reward function can effectively mitigate this issue.

6.3.1 Trajectory-based Reward Function Design with E-Optimality

Since the objective function of E-optimality is similar to that of D-optimality, the reward function design for E-optimality follows the previous approach described in Section 5.3.2. The parameter estimation experiment is also divided into two phases: the first phase focuses on exploration by adjusting the action to maximize the FIM objective function, while the second phase imposes the same regression task, requiring the agent to follow a quintuple polynomial trajectory back to the origin. The asymmetric PPO framework and parameter initialization method, which has demonstrated good performance in previous iterations, remain unchanged for using E-optimality.

Considering the increased dimensionality of the FIM matrix, to ensure effective learning of the agent, the total number of training steps was increased to 5 million. The actor network was saved regularly during training and the training process was completed after 3.5 hours. The trained agent was tested in five experiments, with system parameters sampled randomly before each experiment. The results are displayed in Figure 6.2. It can be observed that as training progressed, the agent-driven motor ran at a high speed at the start of the experiment. In the first phase, the agent's strategy was to run the motor at high speed and high angle simultaneously. After 200 ms, the motor position entered a risky range, prompting the agent to reduce the system input and thereby decrease the speed. By Experiment 5, the agent had further reduced the system input compared to other experiments, but the motor position still occasionally exceeded the constraints and caused the penalties. This suggests that while the agent made reasonable adjustments to the system state, the magnitude of these changes was sometimes insufficient to counteract the effects of the wide range of system parameters sampled from distribution.

During the second phase of the experiment, it was observed that the velocity trajectory during motor reset was no longer a smooth curve. The reason for this is that, due to the large initial motor position at 300 ms, the velocity trajectory obtained from solving the quintic polynomial exceeded the 500 rad/s

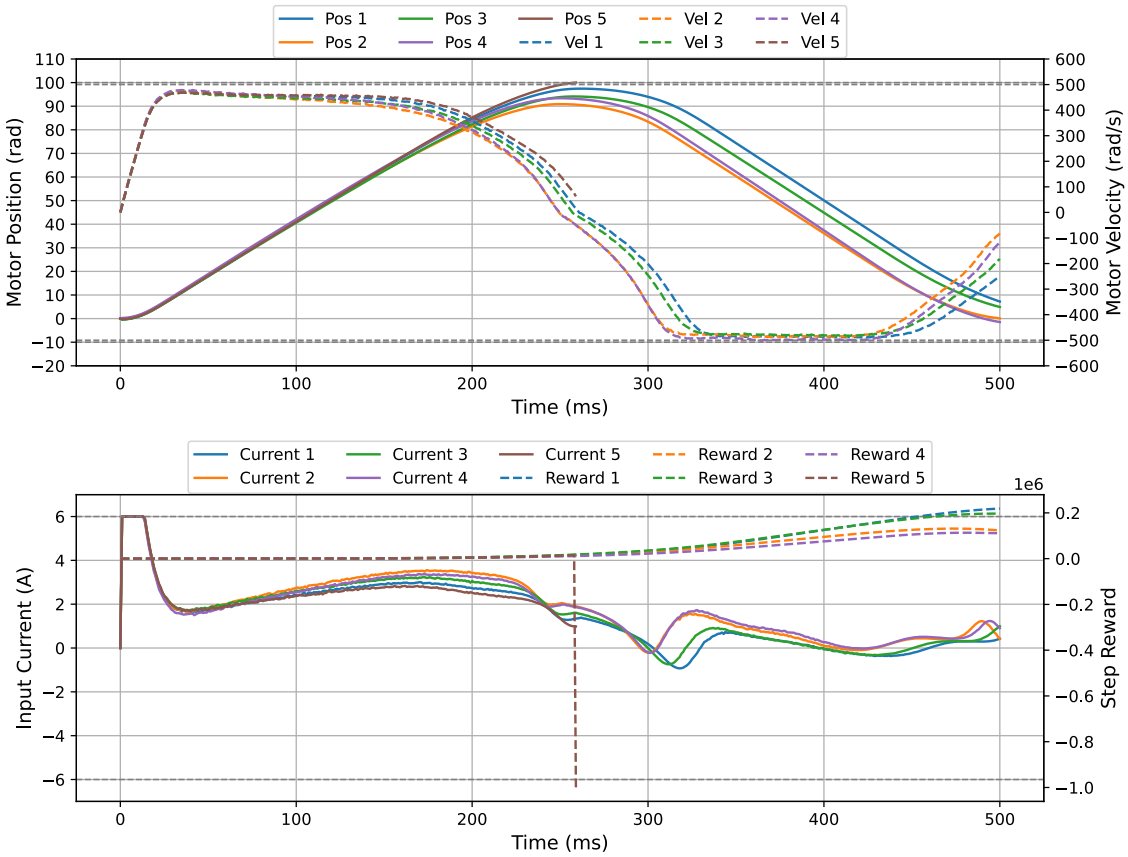

Figure 6.2: Performance of the agent trained with trajectory-based reward function in 5 random experiments

limit. The episode will terminate immediately once the velocity exceeds the boundary condition. As a result, despite the agent's attempt to follow the trajectory during training, it was unable to return to the origin by the end of the experiment. Hence, more aggressive reset strategies need to be considered to ensure better return to origin abilities during the second phase.

6.3.2 Integration of the Trajectory-based and Adaptive Reward Function Design

In this subsection, the method of the adaptive reward function design is integrated with trajectory-based reward function. Given the substantial differences between the motor trajectories driven by the reinforcement learning agent in a multi-parameter estimation environment, the trajectory tracking phase was shortened to 300-450 ms in the new reward function design. Additionally, an additional phase of 50 ms was incorporated to return to origin.

The adaptive reward function design was applied to the final phase. Considering that the ability of the motor to change position in a short period is weaker than the ability to change speed, the reward function in Phase 3 is designed as follows

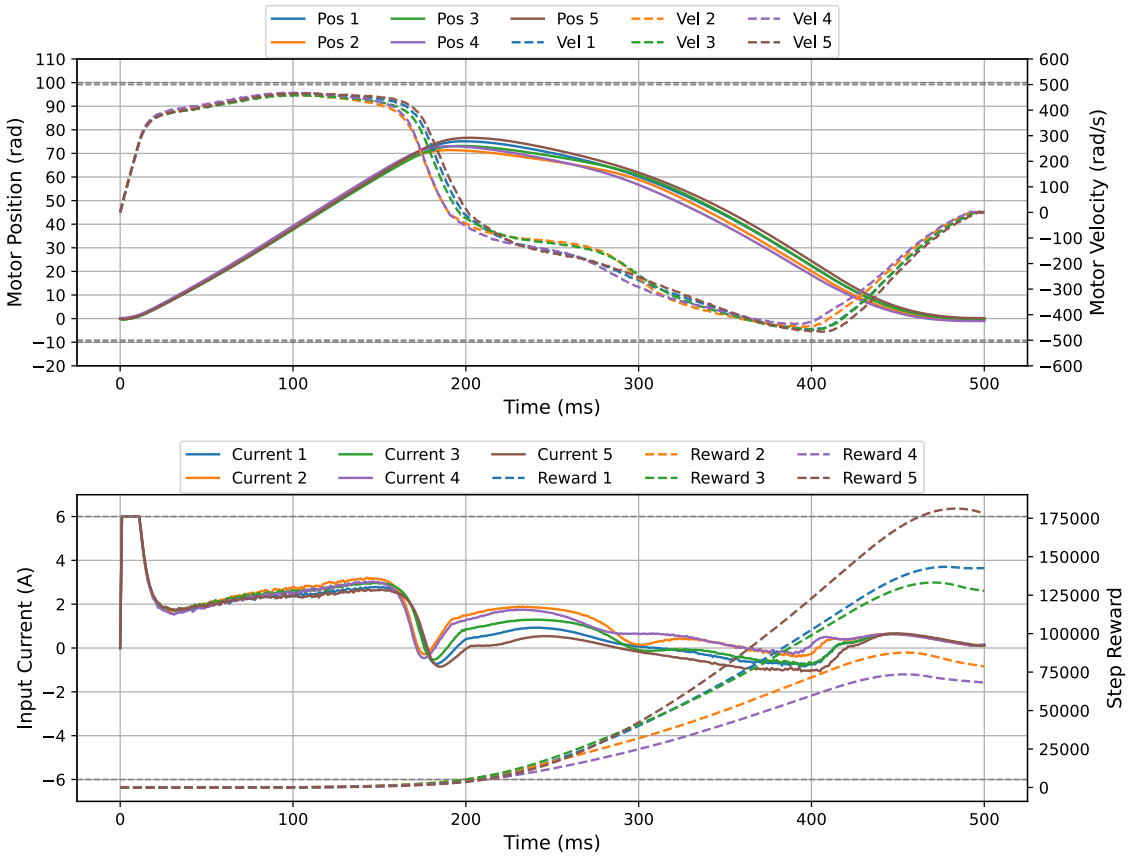

Figure 6.3: Performance of the agent trained with trajectory-based reward function in 5 random experiments

$$r_k = -a\left(1 - \left(\frac{k-450}{50}\right)^2\right)x_1^2 - b\left(\frac{k-450}{50}\right)^2 x_2^2 \tag{6.6}$$

where a and b are parameters to control the weight of the position and velocity penalty terms. The weighting of the position penalty term is larger at the beginning of the third phase and then gradually decreases, while the weighting of the velocity penalty term is set to its opposite.

Similarly, as with the preceding design, the FIM reward information generated by the EMB between 300-500 ms is not immediately provided to the agent, instead it is rewarded once the zero point is reached. It is important to note that, in the case of E-optimality, based on the training experience, the end-step reward needs to be appropriately scaled down. Otherwise, the agent may fail to properly excite the EMB system and become stuck in a local optimum, as it could receive a larger reward by remaining near the zero point for the entire experiment.

After training, five times parameters random sampling experiments were performed, and the results are shown in Figure 6.3. It can be observed that, although the velocity and position of the motors varied at 450 ms in each experiment, they all moved towards zero and successfully returned to zero after 450 ms.

Agent	Constraint Violations	Mean Return	Mean Return (Safe)
Trajectory-Based	56	1.05×10^7	2.40×10^{14}
Hybrid	0	1.53×10^7	1.91×10^{13}

Table 6.2: Performance comparison of agents in 100 random parameter tests

This demonstrates the effectiveness of the integration of trajectory-based and adaptive reward functions in the return to origin task.

To compare the performance of this agent with the one described in the previous subsection in terms of input excitation of parameter identification, additional experiments were conducted. The reward function for the test procedure was computed independently from training, which depended only on the E-optimal criterion, the minimal eigenvalue of FIM, with a penalty of 1×10^6 applied in case of system constraints being violated.

Table 6.2 shows the performance of the two agents in 100 random parameter tests, with the experiments using the same seed to ensure a consistent order of parameter sampling. Compared to the trajectory-based agent, which experienced constraint violations in more than half of the experiments, the hybrid agent experienced no violations and performed more consistently. Thus, the hybrid agent achieved a 50% higher average return because it received no penalty. When these two agents are compared under the parameters that they could both make in the safe case, the trajectory-based agent gets a 26% higher average return. It shows that while the trajectory-based agent learns a more aggressive policy which could get more information in some cases, but due to the uncertainty of the system parameters, this policy could cause damage to the EMB model. Thus, for safety reasons, the hybrid agent demonstrates better performance, contributing to more efficient parameter estimation of the EMB system while ensuring that no constraints are violated under any circumstances.

7 Conclusion and Outlook

7.1 Conclusion

In summary, this thesis focuses on solving the input excitation optimization problem of parameter identification for EMB systems with a reinforcement learning approach. First, a simplified EMB model is constructed to limit the number of parameters in the optimization problem. Then, the optimal experimental design in the EMB system is formulated as an optimization problem in a discretized system. The E-optimal and D-optimal objective functions based on the Fisher information matrix of the optimization problem are improved so that they can be applied to reinforcement learning. To address the instability of the FIM eigenvalues that occurs in the actual computation of the D-optimal theory, an initialization and regularization of the FIM is proposed that ensures numerical stability during computation without affecting the computation of the reward function for reinforcement learning.

To overcome the limitations of traditional optimal experimental design in dealing with system parameter uncertainty, a random distribution-based parameter initialization method is used during reinforcement learning training to generate system parameters. This allows the reinforcement learning-based optimal experimental design (OED) to have a better generalization capability by continuously sampling unknown system parameter values, thereby enhancing the robustness of the trained agent to system parameter variations.

Considering the uncertainty of system parameters in real application scenarios, an asymmetric PPO architecture is proposed and applied to the optimization of input excitation. The main idea of this architecture is to take advantage of the actor-critical nature of the PPO algorithm and exploit the partial observability of the environment when constructing the observation space for reinforcement learning. Specifically, in the EMB system, the actor's observation space includes the motor's position, velocity, and time step, while the critic's observation space includes the motor's true position, velocity, time step, system parameters obtained from sampling, and FIM elements related to reward computation. The advantage of this architecture is that information not available in real experiments can be effectively used in training within the simulation environment. For example, the unknown system parameters can be used to compute the OED objective function, which allows for effective evaluation of the agent's behavior and guidance of policy iterations. In practical applications, the trained agent does not need the critic network, so the unknown system parameters can be ignored. Furthermore, since the practical application is considered as an inference process, the agent makes decisions immediately after acquiring the motor position and velocity, making this approach suitable for systems with high real-time requirements.

Practical system constraints and experimental requirements are also considered in the application of reinforcement learning in EMB systems. For single-parameter input excitation optimization, the RL-trained agent demonstrates robustness to unknown parameters and strong generalization ability. The system input trajectories generated by the agent not only maximize the information content while

ensuring that the system constraints are not violated, but also reliably bring the EMB back to zero even when the system parameters are unknown. Extending this result to multi-parameter experimental design, after improving the optimality criterion, the speed and position of the EMB system remain within constraints while ensuring that the experiment generates higher information content.

Compared to traditional OED methods, the approach proposed in this thesis effectively resolves the dependence of OED on the prior values of system parameters. While deep reinforcement learning requires significant computational power and training time in the range of hours, the trained agent in practical applications requires minimal computational resources and can be deployed on edge devices, making it highly suitable for input excitation optimization for in-vehicle EMB parameter identification.

In contrast to existing research on reinforcement learning for OED in other areas, the method presented in this thesis operates in a continuous action space, allowing finer control over system inputs. In addition, this method allows updating of system inputs at the millisecond level, making it suitable for OED in systems such as EMB, which have strict real-time requirements.

7.2 Outlook

The approach presented in this thesis shows good performance in optimizing parameter estimation inputs for EMB systems, while more related research can be carried out in the future. On the one hand, the EMB model studied in this thesis is highly simplified, and the performance of reinforcement learning in more complex system models needs to be investigated. On the other hand, this thesis demonstrates experimental results in a two-dimensional parameter scenario, and performance in higher-dimensional parameter scenarios remains to be explored.

Additionally, while the input excitation optimization responds to the information content of the experimental data, the parameter identification process has yet to be carried out. A potential future direction is to construct an estimator using an output error or prediction error method to identify EMB system parameters and evaluate the accuracy of the parameters identified using input trajectories generated by the reinforcement learning agent. This work could be conducted in a simulation environment based on a model and also in the real world based on an experimental platform.

It is also important to note that, even though the experiments in this thesis are based on a simplified EMB system, the reinforcement learning-based input excitation optimization method presented here has no theoretical limitations for other applications. The success of the asymmetric PPO architecture and random distribution-based parameter initialization in the EMB system can be generalized to other systems by appropriately modifying the observation space and reward function in the reinforcement learning environment. Furthermore, the development of long short-term memory (LSTM) methods [88] enhances the ability of neural networks to make predictions based on long-term dependencies. For systems with longer experimental times, combining reinforcement learning with LSTM for optimal experimental design could be an interesting direction for future research.

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: 2024-12-02, 2024. [Online]. Available: `https://www.deepl.com/translator`.

[21] OpenAI, *Chatgpt (gpt-4)*, Accessed: 2024-12-01, 2024. [Online]. Available: `https://chat.openai.com/`.

[22] T. Söderström and P. Stoica, "Some properties of the output error method", *Automatica*, vol. 18, no. 1, pp. 93–99, 1982.

[23] P. Kabaila, "On output-error methods for system identification", *IEEE Transactions on Automatic Control*, vol. 28, no. 1, pp. 12–23, 1983.

[24] L. Dugard and I. D. Landau, "Recursive output error identification algorithms theory and evaluation", *Automatica*, vol. 16, no. 5, pp. 443–462, 1980.

[25] G. C. Goodwin and K. S. Sin, *Adaptive filtering prediction and control*. Courier Corporation, 2014.

[26] F. Ding and T. Chen, "Parameter estimation of dual-rate stochastic systems by using an output error method", *IEEE Transactions on Automatic Control*, vol. 50, no. 9, pp. 1436–1441, 2005.

[27] D. Wang, "Least squares-based recursive and iterative estimation for output error moving average systems using data filtering", *IET Control Theory & Applications*, vol. 5, no. 14, pp. 1648–1657, 2011.

[28] M. Gautier, A. Janot, and P.-O. Vandanjon, "A new closed-loop output error method for parameter identification of robot dynamics", *IEEE Transactions on Control Systems Technology*, vol. 21, no. 2, pp. 428–444, 2012.

[29] A. Grkić, D. Mikluc, S. Muždeka, *et al.*, "A model for the estimation of brake interface temperature", *Strojniški vestnik-Journal of Mechanical Engineering*, vol. 61, no. 6, pp. 392–398, 2015.

[30] R. Hoseinnezhad, A. Bab-Hadiashar, and P. Harding, "Calibration of resolver sensors in electromechanical braking systems: A modified recursive weighted least-squares approach", *IEEE Transactions on Industrial Electronics*, vol. 54, no. 2, pp. 1052–1060, 2007.

[31] Q. Lai, H. J. Ahn, Y. Kim, Y. N. Kim, and X. Lin, "New data optimization framework for parameter estimation under uncertainties with application to lithium-ion battery", *Applied Energy*, vol. 295, p. 117 034, 2021.

[32] N. Dirkx, J. van de Wijdeven, and T. Oomen, "Frequency response function identification for multivariable motion control: Optimal experiment design with element-wise constraints", *Mechatronics*, vol. 71, p. 102 440, 2020.

[33] K. Kamali and I. A. Bonev, “Optimal experiment design for elasto-geometrical calibration of industrial robots”, *IEEE/ASME Transactions on Mechatronics*, vol. 24, no. 6, pp. 2733–2744, 2019.

[34] S. Asprey and S. Macchietto, “Designing robust optimal dynamic experiments”, *Journal of Process Control*, vol. 12, no. 4, pp. 545–556, 2002.

[35] M. N. Cruz Bournazou, T. Barz, D. Nickel, *et al.*, “Online optimal experimental re-design in robotic parallel fed-batch cultivation facilities”, *Biotechnology and bioengineering*, vol. 114, no. 3, pp. 610–619, 2017.

[36] L. Pronzato and E. Walter, “Robust experiment design via stochastic approximation”, *Mathematical Biosciences*, vol. 75, no. 1, pp. 103–120, 1985.

[37] S. Körkel*, E. Kostina, H. G. Bock, and J. P. Schlöder, “Numerical methods for optimal control problems in design of robust optimal experiments for nonlinear dynamic processes”, *Optimization Methods and Software*, vol. 19, no. 3-4, pp. 327–338, 2004.

[38] D. V. Lindley, “On a measure of the information provided by an experiment”, *The Annals of Mathematical Statistics*, vol. 27, no. 4, pp. 986–1005, 1956.

[39] M. J. Chen, K. Sivakumar, G. A. Banyay, *et al.*, “Bayesian optimal sensor placement for damage detection in frequency-domain dynamics”, *Journal of Engineering Mechanics*, vol. 148, no. 12, p. 04 022 078, 2022.

[40] R. Hölter, M. Schoen, A. A. Lavasan, and E. Mahmoudi, “Model validation using bayesian optimal experimental design in urban mechanised tunnelling”, in *rd UNCECOMP 2019 ECCOMAS Thematic Conference on Uncertainty Quantification in Computational Sciences and Engineering*.

[41] W. Shen, “Reinforcement learning based sequential and robust bayesian optimal experimental design”, Ph.D. dissertation, 2023.

[42] Q. Long, M. Scavino, R. Tempone, and S. Wang, “Fast estimation of expected information gains for bayesian experimental designs based on laplace approximations”, *Computer Methods in Applied Mechanics and Engineering*, vol. 259, pp. 24–39, 2013.

[43] N. J. Treloar, N. Braniff, B. Ingalls, and C. P. Barnes, “Deep reinforcement learning for optimal experimental design in biology”, *PLOS Computational Biology*, vol. 18, no. 11, e1010695, 2022.

[44] A. Mesbah, “Stochastic model predictive control with active uncertainty learning: A survey on dual control”, *Annual Reviews in Control*, vol. 45, pp. 107–117, 2018.

[45] X. Feng, Y. Jiang, M. E. Villanueva, and B. Houska, “Parallelizable real-time algorithm for integrated experiment design mpc”, *IFAC-PapersOnLine*, vol. 51, no. 18, pp. 518–523, 2018.

[46] J. Espin, Y. Kajiura, and D. Zhang, “Safety-driven battery charging: A fisher information-guided adaptive mpc with real-time parameter identification”, *arXiv preprint arXiv:2406.08626*, 2024.

[47] J. W. Kim, N. Krausch, J. Aizpuru, *et al.*, “Model predictive control guided with optimal experimental design for pulse-based parallel cultivation”, *IFAC-PapersOnLine*, vol. 55, no. 7, pp. 934–939, 2022.

[48] 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.

[49] 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.

[50] 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.

[51] C. Qi, "Kontaktereigniserkennung für ein elektromechanisches system", Master's Thesis, TU Berlin, 2024, p. 8.

[52] 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.

[53] Y. Ki, K. Lee, J. Cheon, and H. Ahn, "Design and implementation of a new clamping force estimator in electro-mechanical brake systems", *International Journal of Automotive Technology*, vol. 14, pp. 739–745, 2013.

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

[55] 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.

[56] 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.

[57] 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.

[58] 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.

[59] W. Gautschi, *Numerical analysis*. Springer Science & Business Media, 2011.

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

[61] 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.

[62] 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.

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

[64] A. Paszke, S. Gross, S. Chintala, *et al.*, "Automatic differentiation in pytorch", 2017.

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

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

[67] 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.

[68] V. Mnih, "Playing atari with deep reinforcement learning", *arXiv preprint arXiv:1312.5602*, 2013.

[69] T. Lillicrap, "Continuous control with deep reinforcement learning", *arXiv preprint arXiv:1509.02971*, 2015.

[70] 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.

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

[72] J. Schulman, "Trust region policy optimization", *arXiv preprint arXiv:1502.05477*, 2015.

[73] 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/`.

[74] 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.

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

[76] 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.

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

[78] 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.

[79] 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.

[80] 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.

[81] 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.

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

[83] 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`.

[84] 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.

[85] 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.

[86] 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.

[87] 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.

[88] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-c. Woo, “Convolutional lstm network: A machine learning approach for precipitation nowcasting”, *Advances in neural information processing systems*, vol. 28, 2015.